Knowledge Transfer from Deep Electronic Networks to Optical Neural Networks

Jinlin Xiang

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2025

Reading Committee:

Eli Shlizerman, Chair

Arka Majumdar

Radha Poovendran

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Knowledge Transfer from Deep Electronic Networks to Optical Neural Networks

Jinlin Xiang

Chair of the Supervisory Committee:
Eli Shlizerman
Electrical and Computer Engineering

Optical Neural Networks (ONNs) offer a promising alternative to electronic networks for artificial intelligence computing by leveraging the speed of light, providing lower power consumption and latency. However, implementing ONNs remains challenging due to high energy cost of nonlinear operations and the precise alignment required for multi-layer optical systems. Previous research introduced hybrid approaches that combine an optical frontend for fast computation with an electronic backend for nonlinear processing. While end-to-end optimization for hybrid ONNs has been demonstrated on specific datasets and optical configurations, these approaches typically lack generalization across tasks and hardware design. This is primarily due to the optical frontend's inability to reliably mimic the feature extraction capabilities of state-of-the-art electronic networks.

In my research, I proposed to transfer knowledge from electronic networks to hybrid electro-photonic convolutional neural networks, enabling the optical frontend to capture features similar to electronic networks while simplifying the model architecture. I trained the hybrid network using a teacher-student transfer learning framework, where a nonlinear electronic teacher network guided the optical frontend to learn features while circumventing nonlinearity. Next, I collaborated with colleagues to compress the convolutional layers of electronic networks (e.g., AlexNet) into a single layer, reducing the need for precise optical alignment and lowering computational costs. Compared with previous works, this approach

reduced latency and power consumption while improving feature alignment via transfer learning.

Furthermore, considering a continual learning setting, I introduced a novel tangent kernel loss as an effective approach for a transfer learning framework. Then, I integrated the approach based on tangent kernel loss into ONNs to form a unified pipeline, Neural Tangent Knowledge Distillation (NTKD). This task-agnostic and hardware-agnostic framework supports image classification and segmentation across diverse optical systems. Experiments on multiple datasets and hardware configurations show that NTKD pipeline consistently enhances accuracy and enables practical deployment in both pre-fabrication simulations and physical implementations.

TABLE OF CONTENTS

Page

List of Figures iii

List of Tables viii

Chapter 1: Introduction 1
1.1 Background 1
1.2 Summary My Research Contributions 3
1.3 Thesis Outline 6

Chapter 2: Fundamental Concepts and Related Works 7
2.1 Optical Neural Networks 7
2.2 Hybrid Optical/Digital ONNs for Computer Vision 10
2.3 Transfer Learning for Hybrid ONNs 11
2.4 Compensation Strategies for Practical ONNs 12

Chapter 3: Transfer Knowledge with Teacher-Student Structure 13
3.1 Knowledge Distillation Loss 13
3.2 Neural Tangent Kernel and Gradient Tangent Kernel Loss 16

Chapter 4: Knowledge Transfer to 4f-based Optical Neural Networks 25
4.1 Motivation 25
4.2 Methods 29
4.3 Experiments and Results 36
4.4 Discussion 41

Chapter 5: Knowledge Transfer to Metasurface-based Optical Neural Networks . . 45
5.1 Motivation 45
5.2 Methods 49

5.3 Experiments and Results 54
5.4 Discussion 57

Chapter 6: Transferable Polychromatic Optical Encoder 65
6.1 Motivation 65
6.2 Methods 68
6.3 Experiments and Results 73
6.4 Discussion 78

Chapter 7: Balanced Neural Tangent Knowledge Distillation for Incremental Learning 82
7.1 Motivation 82
7.2 Methods 84
7.3 Experiments and Results 86

Chapter 8: Neural Tangent Knowledge Distillation for Optical Convolutional Networks 92
8.1 Motivation 92
8.2 Methods 94
8.3 Experiments and Results 99
8.4 Discussion 104

Chapter 9: Conclusion 108

Bibliography 112

Appendix A: Appendix 139
A.1 Connection with Gaussian Processes 139
A.2 Deterministic NTK 141
A.3 Noise Robustness in Wide Finite-Width Networks 144

LIST OF FIGURES

Figure Number Page

1.1 Summary of Research Contributions. This figure presents a unified transfer-learning framework for optical neural networks (ONNs) that I have introduced. The framework is built upon two core algorithmic components: knowledge distillation and tangent kernel based optimization. It demonstrates four applications: one-time optical transfer, meta-optical encoder compression, class incremental learning with GTK loss, and NTK guided optical convolutional networks. 2

2.1 PSF and Convolution illustrated with discrete matrices. (a) Illustrates a point source. (b) Represents an arbitrary PSF which may be obtained by imaging (a) through a realistic lens system. (c) An arbitrary source, or input image, which is an array of point sources. (d) The expected image produced by imaging the arbitrary source (c) through the lens system with PSF shown in (b). (Figure from [JX3]) 9

2.2 Transfer learning is machine learning with an additional source of information apart from the standard training data: knowledge from one or more related tasks. 11

3.1 Illustration of KD training. Student network (green-bottom) parameters Φ are updated according to an interpolation of two losses: the Student loss (cross-entropy loss between data and the student model output) and Temp loss (cross-entropy loss or KL divergence) between the teacher network (blue-top) and student class distribution with a temperature parameter T. (Figure from [JX1]) 14

3.2 Temperature values impact the probability distributions in KD. The plots represent four distinct temperature settings: $T = 1$, $T = 2$, $T = 5$, and $T = \infty$. As T increases, the probability distribution becomes more uniform, and at very high T, the distinction between correct and incorrect classes diminishes. Thus, selecting an appropriate T is crucial for effectively transferring knowledge from teacher to student. 16

3.3 Illustrates the parameters $\boldsymbol{\theta}$ in the LeNet-5 architecture. Input Layer: Grayscale images of size 32×32. Convolution 1: Contains 6 filters, each of size 5×5, with a stride of 1 and valid padding, resulting in an output of size $28 \times 28 \times 6$. Pooling 1: Applies average pooling with a 2×2 filter and stride of 2, reducing the output size to $14 \times 14 \times 6$. Convolution 2: Consists of 16 filters of size 5×5, with a stride of 1 and valid padding, producing an output of $10 \times 10 \times 16$. Pooling 2: Applies average pooling with a 2×2 filter and stride of 2, reducing the output size to $5 \times 5 \times 16$. Dense 1 (Fully Connected Layer): Flattens the previous output and connects it to 120 units, followed by an output of 84 units. Dense 2: Connects to 84 units with an output of 10 units for final classification. 18

3.4 Weight Change Comparison for Different Network Widths: We compare the norm of weight changes during training for three different network widths $m = 10$, $m = 100$, and $m = 1000$. As the width increases, the weight updates become smaller, with the network experiencing the least change at $m = 1000$. 20

3.5 Illustration of NTK evolution for finite-width and wide-width neural networks. Top: In the finite-width case (2-layer network with width m = 10), the NTK grows significantly over time. Bottom: In the wide-width case (2-layer network with width m = 1000), the NTK remains almost constant during training. The training dataset consists of 100 images from MNIST. 21

3.6 Comparison between knowledge distillation processes. Left: In the traditional KD method, the teacher model generates labels based on data, and the student model learns by minimizing the loss between its predictions and the teacher's labels. Right: In our GTK loss approach, both the teacher and student models use their neural tangent kernels to capture gradient information ($G_{teacher}, G_{student}$). The student minimizes the loss by aligning its gradients with those of the teacher, enhancing training efficiency through this gradient-based distillation process. 24

4.1 Nonlinear CNN (top) and the proposed substitute, Spectral CNN Linear Counterpart (SCLC) (bottom). Top: An example of nonlinear CNN with layers that include operations of Convolution, Nonlinear RELU activation, and Max-Pool. Bottom: SCLC of the CNN shown in top row. The convolution corresponds to the elementwise product in the spectral domain. The RELU operation is excluded. The Max-Pool layer is represented by a center-crop operation in the spectral domain. (Figure from [JX1]) 28

4.2 Using a lenslet array and array of phase and amplitude masks, we can implement all the convolutions in a parallel fashion. We can also implement Spectral Pooling in such a structure by placing an amplitude mask in the Fourier plane. Both the lenslet array and convolutional masks can be implemented using meta-optics. (Figure from [JX1]) 34

4.3 Accuracy and Forward propagation time for classification task on High-10 dataset. Left: AlexNet and SCLC accuracy for varying input resolution; Right: Forward propagation runtime of AlexNet, SCLC, and OSCLC for varying input resolution. (Figure from [JX1]) 39

4.4 Examples of object segmentation for the considered benchmarks of (i) Car Segmentation, (ii) Face Recognition, and (iii) VOC2012. Left to right: Ground truth, Our work (SCLC/ OSCLC). (Figure from [JX1]) 40

5.1 Schematic of convolutional neural networks for image classification tasks. (a) All-electronic multi-layered CNN. (b) All-electronic compressed CNN. (c) Hybrid CNN which combines an optical meta-optic front end and electronic backend. (d) The number of multiply-accumulate (MAC) operations of each network configuration, with convolutional MACs in green and fully-connected (FC) MACs in brown. (Figure from [JX3]) 48

5.2 Schematic of the optical system. (a) PSF measurement setup using a monochromatic point light source (left) and optical convolution measurements using a micro-LED display (right). (b) A photograph of the fabricated meta-optics. The meta-optic contains 16 different sub-optics, spatially distributed in a single layer, operating in parallel for classification tasks. (c) Phase maps and SEM images of exemplary sub-optics corresponding to the positive and negative parts of a particular convolutional kernel. (d) The positive and negative parts of an example convolutional kernel (left) and the corresponding PSF simulation (middle) and right (experiment). (e) The simulated electronic output (left) and optical experiment (right) convolved output for the example kernel, for the case of an input "7" from MNIST. (Figure from [JX3]) 52

5.3 Confusion matrices for different network architectures. (a) Classification results for AlexNet-Mod (multiple-layer electronic CNN). (b) Classification results for the all-electronic CNN compressed without using knowledge distillation. (c) Classification results for the all-electronic CNN compressed with knowledge distillation. (d) Classification results for the hybrid optical-electronic CNN. (Figure from [JX3]) 58

5.4 PCA of the hybrid CNN. (a) PCA of the uncalibrated experimental hybrid CNN classification data. (b) PCA of the calibrated experimental data, which has been re-mapped and exhibits clustering behavior similar to that of the compressed electronic CNN data. (c) PCA of the compressed electronic CNN data. (Figure from [JX3]) 59

6.1 Schematic process flows of different image classification methods using original CNN, compressed all-electronic CNN, and hybrid optical/digital CNN. (Figure from [JX4]) 66

6.2 Schematics of the transfer learning process. (Figure from [JX4]) 72

6.3 Optical characterizations of the meta-optic encoder. (a) Photograph of the fabricated optical encoder, consisting of 16 positive and 16 negative convolutional kernels and 5 algnment metalenses. (b) Schematics of the polychromatic PSFs measurement setup. (c) Ground-truth and measured RGB PSFs for a particular polychromatic kernel (positive kernel number 7). (d) Schematics of the meta-optical convolved image measurement setup with a micro-display. A color camera capture convolved convolved images with a single shot. (e) Electrically (above) and optically (below) convolved images of a particular CIFAR-10 image in individual RGB colors. (f) Confusion matrices of CIFAR-10 dataset classification tasks with different network architectures. (Figure from [JX4]) 74

6.4 Principal component analysis for CIFAR-10 image dataset. (a) Original CNN, AlexNet. (b) Compressed all-electronic CNN. (c) Hybrid optical/digital CNN without calibration layer. (d) Hybrid optical/digital CNN with additional calibration layer. (Figure from [JX4]) 77

7.1 Continual Learning Framework: Models are trained sequentially across multiple tasks. Each model is associated with a specific set of tasks, and loss functions are used to update model parameters for the current tasks. As the training progresses, new models incorporate knowledge from previously learned tasks, culminating in a generalized pseudo-task. During inference, the current model is able to make predictions based on the accumulated knowledge. (Figure from [JX5]) 83

7.2 T-SNE visualization (colors indicate classes) of feature representations from the last feature extractor on MNIST with 5 Tasks and 2 new classes introduced in each stage. Representations by GTK (bottom) are clustered more efficiently (supported by Davies-Bouldin Index (DBI - lower better)) than representations that do not use GTK (top). (Figure from [JX5]) 89

8.1 Overview of potential applications for ONNs and our proposed deployment pipeline. (a) ONNs for real-time decision-making in power-constrained scenarios. (b) Our proposed pipeline includes user-driven design, knowledge transfer training, fabrication, and error compensation. (Figure from [JX6]) 93
8.2 Overview of the pipeline. It consists of three steps: (1) Optical Frontend Design based on user-specified inputs, (2) Knowledge Transfer Training using Neural Tangent Kernel (NTK) matching, and (3) Error Compensation for fabricated optical frontends. (Figure from [JX6]) 95
8.3 Optical systems. (a) Compressed Meta ONN on MNIST [1]; (b) Polychromatic Meta ONN on CIFAR-10 [2]; (c) ExtremeMETA for segmentation with dual optical frontends [3]; (d) Customized Polychromatic Meta ONN for segmentation (Ours); (e) Optical measurement setup; (f) Fabricated PSF-engineered meta-optics; (g) Scanning electron microscopy image of the meta-optics. (Figure from [JX6]) 99
8.4 Simulation: T-SNE and confusion matrices in MNIST (a-d) and CIFAR-10 (e-h). (Figure from [JX6]) 101
8.5 Fabrication: T-SNE and confusion matrices on MNIST (a–d) and CIFAR-10 (e–h). (Figure from [JX6]) 103
8.6 NTK compressibility analysis. (a) Cumulative eigenvalue power of the Parameter matrix $g = J^\top J$ for LeNet's convolutional layers, showing that 95% and 99% of power are concentrated in just 108 and 288 parameters. (b) Accuracy of models with varying parameter counts. (Figure from [JX6]) 106

LIST OF TABLES

Table Number Page

2.1 Summary of previous ONN works categorized by task type (classification, segmentation) and implementation level—either simulation only (denoted as **Sim**) or with physical fabrication and experimental validation (denoted as **Fab**). (Table from [JX6]) 10

4.1 Comparison of forward propagation computational complexity between Nonlinear CNN, SCLC implemented on electronic hardware, SCLC implemented on optical hardware. (Table from [JX1]) 33

4.2 Forward propagation accuracy and processing-time of different network variants for classification tasks. AlexNet: nonlinear baseline network (no optical implementation), SCLC: The linear counterpart of AlexNet. (Table from [JX1]) 37

4.3 Accuracy and forward propagation rate (frames per second) for object segmentation task tested on three benchmarks. (Table from [JX1]) 42

4.4 Accuracy of AlexNet SCLC with different teacher models. (Table from [JX1]) 43

5.1 Classification accuracy and computational complexity of different neural network architectures on the MNIST dataset. (Table from [JX3]) 56

5.2 Ablation study of different network configurations, highlighting the contribution of calibration and optical components. (Table from [JX3]) 59

6.1 Classification performance of different network architectures on the CIFAR-10 dataset. (Table from [JX4]) 76

6.2 Transfer learning results on High10 (Table from [JX4]) 78

7.1 Performance comparison between TKIL and other SOTA methods on CIFAR-100 (left-half) and ImageNet-100 (right-half). (Table from [JX5]) 88

7.2 Performance comparison between the TKIL and other state-of-the-art methods on MNIST and SVHN (5 stages, 2 new classes per stage, Memory size = $2k$). (Table from [JX5]) 89

7.3 Performance comparison between TKIL and other SOTA methods on ImageNet-1k. Memory size: $\mathcal{M}$, Upper Bound: Joint Training. (Table from [JX5]) . . . 90

8.1 Performance comparison of ONN methods across different tasks and training strategies. (Table from [JX6]) 101
8.2 Evaluation of ONN error compensation methods on fabricated optical systems across both classification and segmentation tasks. 102
8.3 Random PSF kernel design across different tasks. (Table from [JX6]) 104

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to my advisor, Prof. Eli Shlizerman, for his unwavering guidance and support throughout my doctoral studies. His profound insights and patient mentorship have been instrumental in shaping both my research vision and my academic growth. Beyond technical discussions, his thoughtful advice on broader aspects of life and career has deeply influenced my development as a researcher.

I am also deeply grateful to my supervisory committee members, Prof. Steve Brunton (GSR), Prof. Arka Majumdar, and Prof. Radha Poovendran, for their invaluable feedback, encouragement, and commitment to my success. In particular, I am profoundly appreciative of Prof. Arka Majumdar for his visionary perspective on optical systems and for inspiring my exploration of interdisciplinary research that bridges optics and machine learning. His guidance and enthusiasm have profoundly shaped my understanding of how physical principles can inform intelligent system design.

I am fortunate to have collaborated with many talented colleagues and co-authors, including Anna Wirth-Singh, Minho Choi, Yubo Zhang, Zhihao Zhou, Kun Su, Xiulong Liu, Yang Zheng, Jingyuan Li, Mingfei Chen, Jimin Kim, Trung Le, Rahul Biswas, Yan Jiang, and Tianchen Sun. Working alongside them has been both intellectually stimulating and personally rewarding—their creativity, collaboration, and friendship have made this Ph.D. journey a truly memorable experience.

I extend my deep appreciation to my colleagues at Sanofi, including Bozhao Qi, Marc Cerou, Wei Zhao, Qi Tang, Ruisu Zhang, and Yongjian Yang, for the opportunity to work together on impactful projects. Their expertise, collaboration, and support have significantly broadened my perspective and strengthened the bridge between academic research and

industrial applications.

As I reflect on the past five years at the University of Washington, I recognize that the difficulties, uncertainties, and occasional setbacks were as formative as the successes. These experiences have helped me grow into a more resilient and adaptable researcher. With the confidence gained through this journey, I look forward to embracing the opportunities and challenges that lie ahead.

DEDICATION

I dedicate this work to my parents, Meijuan Chen and Xiaohong Xiang, for their unwavering love, strength, and support throughout every stage of my life. I am deeply grateful to my partner, Dawei Gu, whose love, patience, and encouragement have been a constant source of strength and inspiration. His understanding and companionship have made even the most challenging moments meaningful. I also wish to express my heartfelt gratitude to my dear friends Linxi Zhu, Tianchen Sun, Bohan Zhang, and Yetao Chen.

Chapter 1

INTRODUCTION

1.1 Background

Optical Neural Networks (ONNs) are optical computing systems leveraging the properties of light to perform neural network operations, offering an alternative to AI computations [4]. ONNs have the potential to enhance computing speed and energy efficiency [5]. Compared to electronic networks, which require high power consumption, ONNs consume significantly less power making them well-suited for large-scale computations. Moreover, ONNs enable high-speed processing due to the rapid propagation of light, which is instrumental in reducing latency [6]. Furthermore, the inherent parallelism of optical systems allows ONNs to process large volumes of data simultaneously, enhancing both efficiency and computational capacity over electronic systems [7].

The adoption of ONNs has been limited by challenges such as large device sizes, sensitivity to misalignment, and nonlinearity [7]. During 1980s and 1990s, electronic computing and software technologies advanced rapidly, outperforming optical technologies in practical applications [8, 9, 10]. These limitations of optical technologies, combined with limited research and development of neural networks, slowed progress in optical computing during this period [11, 12, 13]. Recent developments in photonics have begun to address these obstacles, introducing solutions such as miniaturization, tunability, and nonlinear photonic interactions [14, 15]. Meanwhile, the growing limitations of electronic computing, especially in terms of power consumption and latency, have renewed interest in optical computing as a solution to the rising computational demands of AI, particularly in fields such as computational imaging and computer vision.

Recent advances have led to the development of three primary approaches for implement-

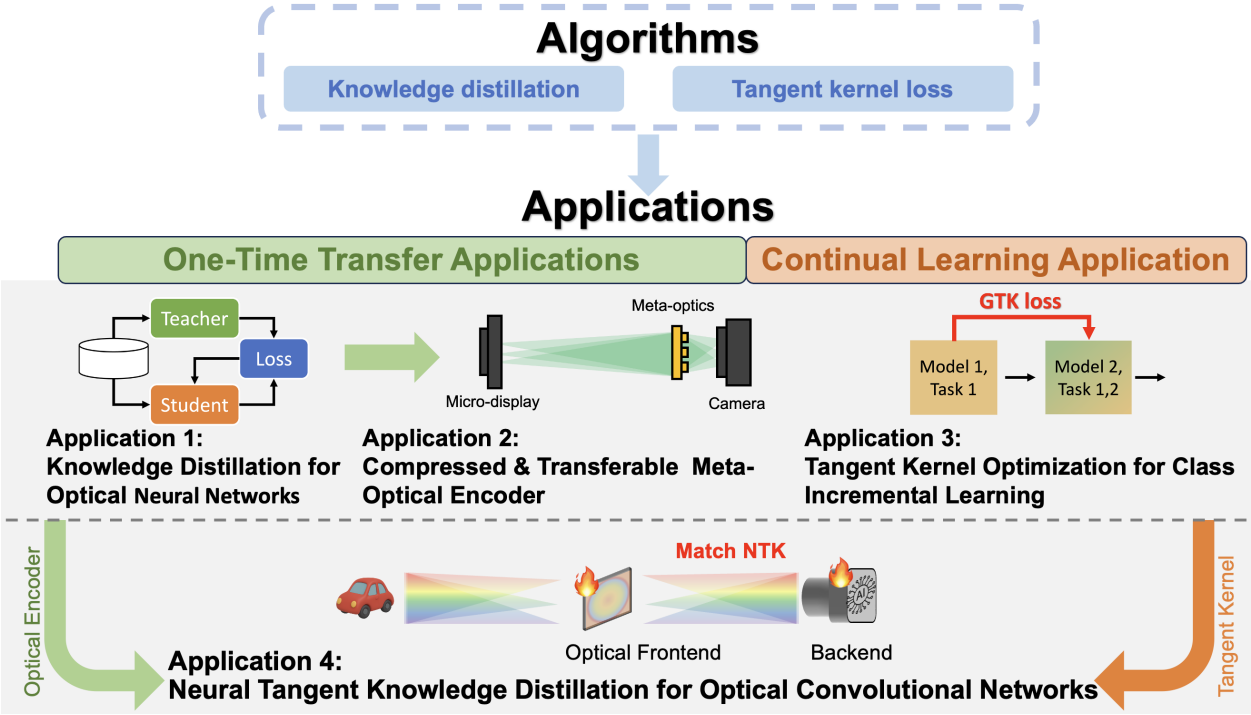


Figure 1.1: Summary of Research Contributions. This figure presents a unified transfer-learning framework for optical neural networks (ONNs) that I have introduced. The framework is built upon two core algorithmic components: knowledge distillation and tangent kernel based optimization. It demonstrates four applications: one-time optical transfer, meta-optical encoder compression, class incremental learning with GTK loss, and NTK guided optical convolutional networks.

ing ONNs: fully optical systems, physically nonlinear ONNs, and hybrid optical-electronic ONNs. Fully optical systems employ multiple layers of diffractive optics and achieve competitive classification accuracy under coherent illumination at terahertz and near-infrared wavelengths [16, 17, 18]. Their operations are linear, as optical components naturally perform linear transformations on light waves. Physically nonlinear ONNs introduce nonlinearity through mechanisms such as atomic vapor cells [19, 20] or image intensifiers [21]. While these methods incorporate nonlinearities, they require repeated conversion between optical and

electronic domains, leading to increased latency and power consumption compared to fully optical systems [22, 23]. Precise alignment between multiple layers is also required since small misalignments can significantly degrade performance [21]. Hybrid optical-electronic architectures have emerged as a promising approach for implementing ONNs [24, 25, 26, 27, 28]. Such approaches leverage a high-speed optical frontend for feature extraction while utilizing an electronic backend to introduce nonlinearity and correct optical misalignment.

Training optical frontends in hybrid networks presents a significant challenge, as linear optical components typically don't effectively mimic electronic neural networks [29]. To address this, compression and model reduction techniques are essential for adapting electronic networks to ONNs while maintaining performance. Various approaches, including pruning and low-rank approximation, have been proposed to enhance efficiency while maintaining accuracy under optical constraints [30, 31, 32]. Pruning, despite effectively removing redundant parameters, is unsuitable for optical frontends as it still relies on multi-layer networks and nonlinear activations. Low-rank approximation does not adequately capture the complexity of datasets, thereby limiting its effectiveness in hybrid ONNs.

Transfer learning offers a promising approach to optimize optical frontends by leveraging knowledge from pre-trained electronic networks, enabling efficient feature learning in optical networks. Among transfer learning techniques, Knowledge Distillation (KD) has been widely used to compress large models that have been validated on various datasets, making it an effective strategy for optimizing optical frontends [33]. By distilling knowledge from an electronic teacher network into an optical student model, KD enables feature learning in optical networks.

1.2 Summary My Research Contributions

In my research, I introduced a knowledge transfer approach to optimize hybrid ONNs, enabling them to better mimic electronic neural networks. Specifically, I demonstrated a hybrid optical-electronic Convolutional Neural Network (CNN) that employed a single optical frontend to mimic the convolutional layers, which was followed by a calibration layer to correct

optical misalignment errors and an electronic backend that introduced nonlinear operations. To enable the optical frontend to learn feature representations similar to those of a nonlinear electronic teacher network, I introduced a teacher-student-based transfer learning approach to enhance network accuracy. This approach leveraged KD loss through a three-step process: (1) pre-training a large electronic teacher model to learn high-quality feature representations, (2) transferring its weights to a smaller optical model to compress convolutional layers and bypass nonlinear activations, and (3) incorporating a calibration layer to mitigate alignment errors and fabrication noise in the optical system [JX1, JX2, JX4]. To validate this approach in experiments, I collaborated with colleagues to implement it in multiple optical architectures. Specifically, we tested the method on $4f$-based ONNs, which used Fourier-based lens systems for spatial filtering and transformation (discussed in Chapter 4), and metasurface-based ONNs, which relied on nanostructure-based planar lenses for wavefront shaping and feature encoding (discussed in Chapter 5– 6).

Furthermore, I propose a novel Gradient Tangent Kernel (GTK) loss as an alternative to KD, which addresses limitations in training hybrid ONNs on large-scale datasets such as ImageNet. In particular, one major limitation of KD is that it encourages the student model to match the teacher's output distribution rather than guiding its parameter updates, leading to suboptimal learning dynamics, especially in hybrid ONNs. To address this challenge, GTK loss explicitly guides the student model's updates to follow the teacher model training trajectory, ensuring that the student learns similar feature representations and converges more consistently. I validated the GTK loss in a Continual Learning (CL) setting, as discussed in Chapter 7. CL allows models to learn new tasks sequentially while retaining prior knowledge, leading to a more comprehensive evaluation framework than a one-time transfer [34]. Demonstrating the effectiveness of GTK loss in CL suggests its potential to improve one-time transfer in hybrid ONNs as well.

Building upon the development of the GTK loss, I further extended the tangent-kernel-based transfer learning to optical systems through the proposed Neural Tangent Knowledge Distillation (NTKD) pipeline. I proposed a task-agnostic and hardware-agnostic pipeline

that supported both image classification and segmentation across diverse optical systems. To assist optical system design before training, the NTKD pipeline designed the metasurface layout according to fabrication constraints. During training, it aligned optical models with electronic teacher networks, thereby narrowing the accuracy gap. After fabrication, the NTKD pipeline further guides fine-tuning of the digital backend to compensate for implementation errors. Experiments across multiple datasets (e.g., MNIST, CIFAR, and Carvana Image Masking) and hardware configurations demonstrated that our pipeline consistently improved ONN performance and enabled practical deployment in both pre-fabrication simulations and physical implementations.

In summary, in Figure 1.1, I illustrate an overview of my research contributions, including transfer learning algorithms and their two primary applications. The key contributions of my current research are as follows:

- I introduced a teacher-student transfer learning framework using knowledge distillation to circumvent the nonlinearity in optical convolutional neural networks. I implemented this framework with $4f$-based optical networks, transferring knowledge from a nonlinear electronic network to a linear optical counterpart [JX1].

- I worked with collaborators to compress the convolutional layers in electronic networks (AlexNet) into a single layer, implemented using a metasurface lens. Through this approach, we demonstrated that knowledge distillation reduced computational operations by approximately 24,000× on CIFAR-10 and 200× on MNIST [JX2, JX3, JX4].

- I proposed a gradient tangent kernel loss as an alternative knowledge transfer loss and validated it in a continual learning scenario [JX5]. Building upon this foundation, I further proposed the Neural Tangent Knowledge Distillation (NTKD) framework, a task-agnostic and hardware-agnostic pipeline for designing and training ONNs, and evaluated its efficacy in both image classification and segmentation tasks [JX6].

1.3 Thesis Outline

This thesis is organized as follows. Chapter 2 introduces the fundamental concepts and related works, including the $4f$ optical system, meta-optical architectures, and transfer learning techniques such as knowledge distillation that form the theoretical foundation of this work. Chapter 3 presents the teacher–student transfer learning framework, beginning with knowledge distillation (KD) and extending it to the proposed tangent kernel loss. Chapters 4 through 6 demonstrate the applications of KD in physical ONN architectures. Specifically, Chapter 4 focuses on knowledge transfer to $4f$-based optical neural networks, Chapter 5 extends this study to metasurface-based ONNs, and Chapter 6 introduces a transferable polychromatic optical encoder for efficient optical processing. Chapters 7 and 8 concentrate on the tangent kernel loss and its extension to hybrid optical–electronic systems. Chapter 7 presents balanced incremental learning under imbalanced data distributions using the tangent kernel loss, while Chapter 8 builds upon this concept to develop the Neural Tangent Knowledge Distillation (NTKD) pipeline, a task-agnostic and hardware-agnostic pipeline that generalizes the tangent kernel based training approach across computer vision tasks, including classification and segmentation, demonstrating robustness and scalability in both simulations and fabricated optical systems. Chapter 9 concludes the thesis by summarizing the key findings and contributions. This thesis includes the material in the author's previous papers published at Nature Communications [JX4], Advanced Photonics Nexus [JX3], Applied Optics [JX1], Conference on Neural Information Processing Systems (NeurIPS) [JX6, JX7], Conference on Lasers and Electro-Optics (CLEO) [JX2], and Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW) [JX5][1].

---

[1]In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of University of Washington's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to `http://www.ieee.org/publications_standards/publications/rights/rights_link.html` to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Chapter 2

FUNDAMENTAL CONCEPTS AND RELATED WORKS

In this chapter, I present the fundamental concepts and related works that form the foundation of this thesis. This material encompasses the principles of Optical Neural Networks (ONNs), including the $4f$ optical system and meta-optical architectures, as well as their integration into hybrid optical–digital pipelines for computer vision tasks. I further review transfer learning and knowledge distillation techniques for optimizing ONN frontends, and discuss compensation strategies addressing fabrication imperfections and physical noise in practical systems. These concepts collectively provide theoretical and methodological basis for the approaches proposed in this thesis.

2.1 Optical Neural Networks

ONNs leverage the parallelism of light to process large-scale data efficiently, making them well-suited for diverse computational tasks. ONNs have been applied to various applications, including image compression and encryption [6], classification [14], and depth sensing [35, 36]. Leveraging the inherent speed of light, these systems enable passive, high-throughput data manipulation [37, 38, 39, 29]. Our work focuses on two primary ONN architectures: the $4f$ optical system and the meta-optical system.

The **4f optical system** is a fundamental optical architecture that utilizes Fourier optics to modulate and filter the spatial frequency components of an image [40]. By leveraging the principles of Fourier optics, this system decomposes an image into its frequency components, facilitating image processing techniques such as filtering and pattern recognition. The $4f$ optical system is employed in various applications, including signal processing and medical imaging. Moreover, these systems can be customized for specific requirements by adjusting

parameters such as lens focal lengths and the types of masks utilized in the Fourier plane, enabling a wide array of tailored solutions for various applications.

The convolution operation can be implemented using the $4f$ optical system by leveraging the **convolution theorem**, which states that convolution in the spatial domain is equivalent to pointwise multiplication in the frequency domain [41]. Let x denote the input (e.g., a signal or image), k denote the convolutional kernel, and $y = x * k$ denote their convolution. In the spatial domain, the convolution is defined as

$$y(i,j) = \sum_m \sum_n x(m,n) \cdot k(i-m, j-n), \tag{2.1}$$

Using the Fourier Transform (the first lens in the $4f$ system), the same operation can be expressed in the frequency domain as

$$Y(u,v) = X(u,v) \cdot K(u,v) = \mathcal{F}(x(i,j)) \cdot \mathcal{F}(k(i,j)), \tag{2.2}$$

where $\mathcal{F}$ denotes the Fourier Transform, and $\cdot$ indicates elementwise multiplication in the frequency domain. (u,v) represent the frequency-domain indices corresponding to the spatial-domain indices (i,j), where u and v denote the horizontal and vertical frequency components, respectively. Then, the result is transformed back to the spatial domain via the inverse Fourier Transform (the second lens in the $4f$ system)

$$y(i,j) = \mathcal{F}^{-1}(Y(u,v)). \tag{2.3}$$

Meta-optics manipulates light at the nanoscale using engineered metasurfaces, offering a compact alternative to traditional bulky optical elements [36]. Unlike traditional optical systems that rely on bulky lenses and mirrors, meta-optics utilizes ultrathin, planar structures made up of subwavelength nanostructures. These metasurfaces, which consist of arrays of engineered nanoparticles, nanorods, or nanoholes, exhibit unique optical properties that allow control over the amplitude, phase, polarization, and direction of light.

The convolution operation can be performed using Meta-optics, where the resultant image corresponds to the input convolved with the system's point spread function (PSF) [42, 43, 44].

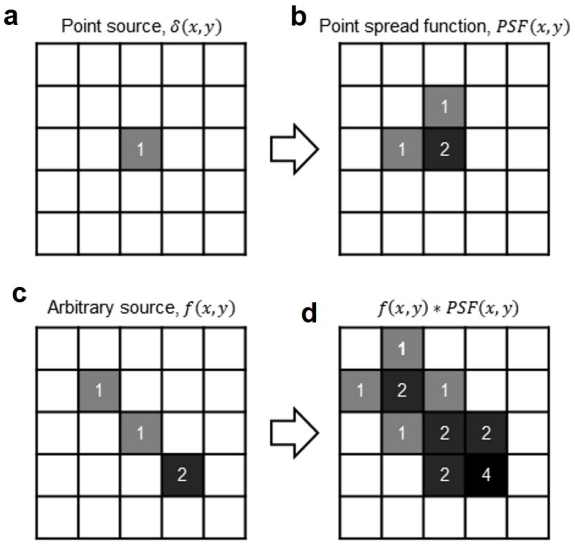


Figure 2.1: PSF and Convolution illustrated with discrete matrices. (a) Illustrates a point source. (b) Represents an arbitrary PSF which may be obtained by imaging (a) through a realistic lens system. (c) An arbitrary source, or input image, which is an array of point sources. (d) The expected image produced by imaging the arbitrary source (c) through the lens system with PSF shown in (b). (Figure from [JX3])

The PSF characterizes the response of an optical system to a point source, determining how the system processes and blurs light. For a discrete input $O[n, m]$ and a discrete point spread function $PSF[n, m]$, the intensity in the image plane $I[n, m]$ is given by

$$I[n,m] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} O[i,j] \cdot PSF[n-i, m-j]. \tag{2.4}$$

The two-dimensional discrete convolution is formally defined as

$$(f * g)[n,m] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} f[i,j] \cdot g[n-i, m-j], \tag{2.5}$$

where $f[i, j]$ represents the input function (e.g., the pixel intensity values of the input image), and $g[i, j]$ represents the kernel function (e.g., the point spread function, PSF, of the optical system).

Table 2.1: Summary of previous ONN works categorized by task type (classification, segmentation) and implementation level—either simulation only (denoted as **Sim**) or with physical fabrication and experimental validation (denoted as **Fab**). (Table from [JX6])

ONN Capability	Works	Classification (Sim)	Classification (Fab)	Segmentation (Sim)	Segmentation (Fab)
Monochromatic	2018–2025: [51, 29, 45, 52, 53, 46, 47, 48, 49, 16, 17, 54, 18, 55, 50, 56, 57, 1][JX2, JX3]	✓	✓	✗	✗
	2023–2025: [14, 58, 59, 54] [JX4]	✓	✓	✗	✗
Polychromatic	ExtremeMETA (2025) [3]	✗	✗	✓	✗
	NTKD [JX6]	✓	✓	✓	✓

By comparing Eq. 2.4 and Eq. 2.5, it is evident that the image produced by an optical system is equivalent to the convolution of the input object with the system's PSF. In Fig. 2.1a, we illustrate an input point source which produces an arbitrary example PSF in Fig. 2.1b. Figure 2.1d demonstrates a discrete convolution between an arbitrary input source (Fig. 2.1c) and optic PSF (2.1b).

2.2 Hybrid Optical/Digital ONNs for Computer Vision

Table 2.1 categorizes ONN applications into two tasks (classification and segmentation) and two optical implementations (monochromatic and polychromatic systems). Most previous work focused on monochromatic ONNs for MNIST image classification, including fully optical systems that performed linear transformations [16, 17], physically nonlinear ONNs that used atomic vapors or intensifiers [45, 46, 47], and hybrid architectures that combined an optical frontend with a digital backend [48, 49, 50]. Previous polychromatic ONNs for classification were limited to small datasets such as CIFAR-10, as ONN architectures faced challenges in scaling to complex benchmarks [14][JX4]. Segmentation tasks are still in the early stages, with a previous study based only on simulation [3]. Our work considers both classification and segmentation tasks. In our pipeline, image reconstruction is implicitly incorporated by encouraging the optical frontend output to align with the simulated result, as the physical output deviates from simulation and requires correction.

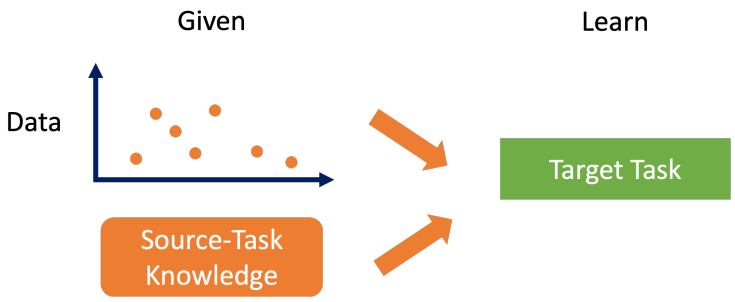


Figure 2.2: Transfer learning is machine learning with an additional source of information apart from the standard training data: knowledge from one or more related tasks.

2.3 Transfer Learning for Hybrid ONNs

Efficient optimization techniques are crucial for overcoming optical hardware limitations in hybrid ONNs, enabling compact and computationally efficient optical frontends that can mimic electronic neural networks. To develop an optical frontend that is both efficient and practical—matching or surpassing the compactness of electronic networks—while reducing computational complexity and memory demands without substantially compromising accuracy, various optimization techniques can be utilized. These include pruning [30, 31], quantization [60], low-rank approximation [61], and transfer learning [62]. However, pruning and low-rank approximation are often less effective for optical applications, as they rely on multi-layer architectures and struggle with the complexity of optical datasets.

Transfer learning improves the accuracy of ONNs by leveraging knowledge from pre-trained models, reducing computational cost and data requirements [63, 64]. Transfer learning is a machine learning technique where a model developed for a specific task is reused as the starting point for a model on a second, related task [62, 65]. Instead of training a model from scratch—which can be computationally expensive and data-intensive—transfer learning leverages knowledge gained from one task to improve learning efficiency on another (also shown in Figure 2.2). Transfer learning encompasses various approaches to facilitate knowledge

transfer [66, 67, 68, 69]. These include instance-based methods (transferring data instances through weighting or sampling), feature-based methods (transforming and aligning latent features across domains), parameter-based methods (leveraging model parameters, such as weights, from large pre-trained teacher models), and relation-based methods (transferring relational or structural knowledge between domains).

Knowledge Distillation (KD) is a specialized form of transfer learning that compresses the knowledge of a large teacher model into a smaller student model, enabling efficient learning for the same task with reduced computational cost [63]. Unlike traditional transfer learning, knowledge distillation aims to compress the knowledge of the teacher model into the student model for the same task. This approach enables the student model to achieve performance close to that of the teacher model while requiring fewer computational resources.

2.4 Compensation Strategies for Practical ONNs

Fabrication imperfections and system noise in physical ONNs often lead to significant performance drops compared to simulations. Some used deep learning to model the system directly in a data-driven manner [70], including ONN auto-learning [71, 72], where ONNs were trained to fit experimental input-output mappings. Other methods introduced physical information via hardware-in-the-loop training. For example, physics-constrained frameworks embedded fabrication-aware models and losses to better align learning with optical behavior [54]. Moreover, some approaches avoided simulation entirely by randomly fabricating optical kernels and training a digital backend to adapt to the fixed frontend structure [73, 74, 75, 76, 77]. This simplified fabrication but placed the learning burden entirely on the backend. It is also not clear if such random surfaces perform better than an ordinary lens. In contrast, our work identifies sources of physical errors and introduces an NTK alignment strategy for effective compensation.

Chapter 3

TRANSFER KNOWLEDGE WITH TEACHER-STUDENT STRUCTURE

3.1 Knowledge Distillation Loss

Knowledge Distillation (KD) is a machine learning method introduced for compression of neural networks [78, 79, 80]. In particular, it defines the transfer of knowledge from a large neural network model, called the *teacher*, to a small model, called the *student* [81, 32]. Figure 3.1 shows the schematic flow of KD. KD assumes that the teacher model is already trained on the given task with high test accuracy. Then the student model is trained with both the approach of minimization of the loss between the model prediction and the training data (*Student loss*) and, in addition, compares the outcome of the student model with the teacher model through a temperature loss (*Temp loss*) using the KL divergence [82]. The optimization of both losses is performed through the back-propagation, which adjusts the parameters of the student model using Stochastic Gradient Descent, or ADAM optimization [83].

It was shown that student models trained with KD approach can achieve significantly higher accuracy than the same model trained on the data alone without a teacher network. The advantage of KD stems from the proposal to compute the temperature loss which treats the prediction of the teacher model as "soft labels" that inform the student model. In particular, conventional learning considers exclusively the labels in the training data as "hard labels" and for tasks such as classification computes the probabilities distribution vector p^{hl}, where each element p_i^{hl} in the vector corresponds to the probability of the student input belonging to the class i. The softmax function is used to compute probabilities

$$p_i^{hl} = exp(z_i) / \sum_j exp(z_j), \tag{3.1}$$

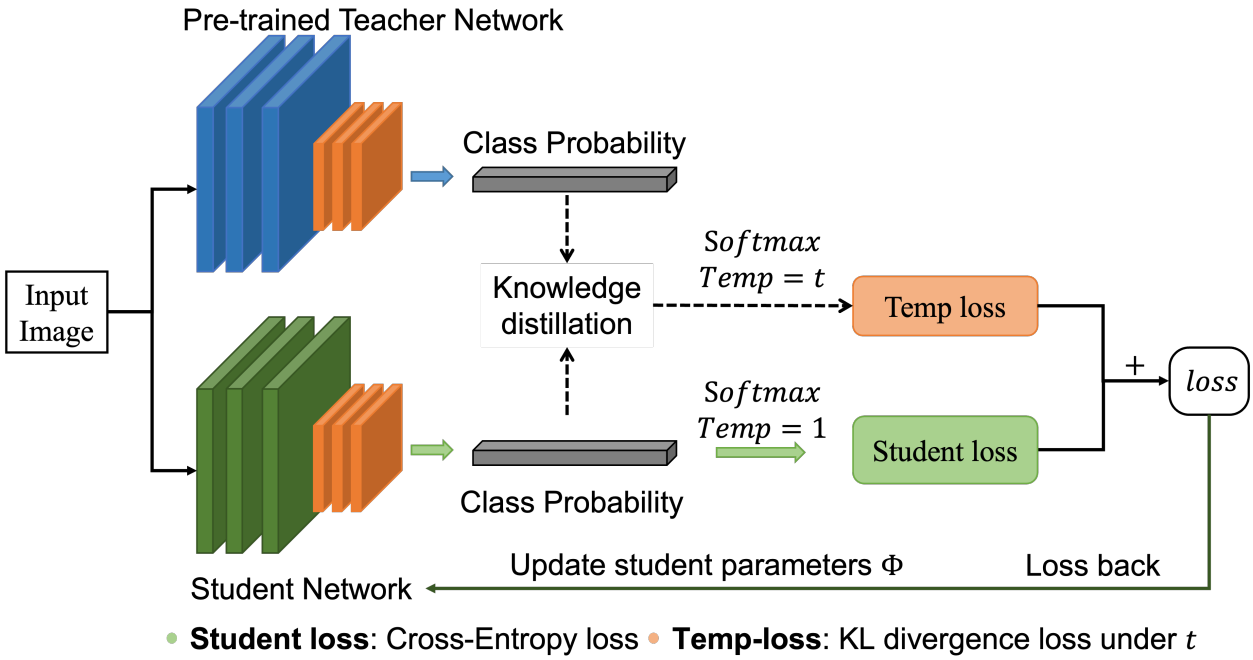


Figure 3.1: Illustration of KD training. Student network (green-bottom) parameters Φ are updated according to an interpolation of two losses: the Student loss (cross-entropy loss between data and the student model output) and Temp loss (cross-entropy loss or KL divergence) between the teacher network (blue-top) and student class distribution with a temperature parameter T. (Figure from [JX1])

where z_i are the student logits after the last fully connected layer [63]. Training with hard labels is a sensitive process, especially for compact networks, such as the student model. This typically results in inefficient networks and in convergence to poor local optima. To overcome this limitation, KD proposes to add soft labels generated by the teacher model. These labels are the probabilities that the teacher model generates. In particular, for each input, at the same time of computing p^{hl} with the student model, KD computes the soft probabilities vector, p^{sl}, with the teacher model according to

$$p_i^{sl} = exp(y_i/T)/\sum_j exp(y_j/T), \tag{3.2}$$

where, y_i are the logits of the teacher after the last fully connected layer [63]. Through p^{sl} the teacher model contributes the probability estimates to the student model. This knowledge is unavailable from the data alone and is helpful for the student since provides extra information of the similarities between classes. The similarities are important since these indicate the effective knowledge of the teacher model and subsequently assists in achieving a similar knowledge and performance in the student model. The two probability distributions p^{sl} and p^{hl} are taken into account (as an interpolation) to compute the overall loss used in training of the student model.

The probabilities contributed by the teacher are defined as soft since the softmax function has a softening parameter, T, named as the distillation temperature. The success of KD depends on the choice of T. In a well-trained teacher model, the correct class has a much higher probability than other classes. For a low value of T, the probability of the correct class will approach 1 while probabilities of other classes will be negligible and will not influence the training, due to p^{sl} being similar to p^{hl}. On the other hand, when T is too high, p_i^{sl} will approach a uniformly distributed vector and will lose the distinction between the correct and incorrect classes. It is therefore important to chose T in between these two extreme cases such that p^{sl} would pass the similarities detected by the teacher to the student. We also present an example in Figure 3.2. In this case, we have five classes: homework (0.997), cake (0.00), book (0.002), school (0.001), and car (0.00). These numbers in parentheses represent the output probabilities from the teacher model for each class. The soft labels not only indicate that "homework" is the correct prediction but also guide the student model by conveying that "homework" is more similar to "book" and "school" than to the other classes. When the temperature is set to 1, the probabilities of the other classes (car, book, school, cake) are small, having minimal influence on the training process. On the other hand, when the temperature is high or approaches infinity, the probabilities of all classes converge to approximately 0.2, removing any distinction between them. By setting the temperature to 5, the subtle similarities identified by the teacher model can be meaningfully preserved and transferred to the student.

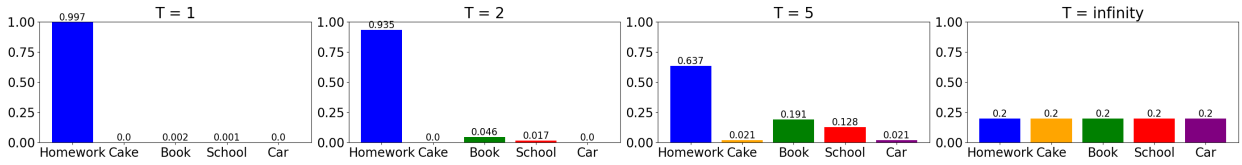


Figure 3.2: Temperature values impact the probability distributions in KD. The plots represent four distinct temperature settings: $T = 1$, $T = 2$, $T = 5$, and $T = \infty$. As T increases, the probability distribution becomes more uniform, and at very high T, the distinction between correct and incorrect classes diminishes. Thus, selecting an appropriate T is crucial for effectively transferring knowledge from teacher to student.

In practice, the distillation of knowledge from the teacher to the student is particularly effective when the differences in the overall architecture between the two networks are minimal [84, 85]. In network compression, it is typically the case that the architectural blocks are kept the same and the only change that is implemented is the reduction of the number of neurons in each block. This leads us to the proposition of the implementation of KD between nonlinear and linear CNNs, where besides exclusion of nonlinear components, the linear network will be kept as similar as possible to the nonlinear one.

3.2 Neural Tangent Kernel and Gradient Tangent Kernel Loss

NTK offers a framework for analyzing the training dynamics of neural networks under gradient descent [86]. This framework becomes particularly insightful as the network's width—defined as the number of channels in convolutional layers and the number of neurons in fully connected layers—approaches infinity. These kernels offer key insights into why sufficiently wide neural networks reliably converge to a global minimum while minimizing empirical loss.

In this section, we adopt the formal definition of the NTK from previous works, followed by an extension of NTK theory to include *finite-width* neural networks. We then introduce the Gradient Tangent Kernel (GTK) loss, designed specifically for the *finite-width* neural

networks, functioning under a teacher-student framework similar to knowledge distillation, to transfer knowledge effectively. This new GTK loss is further tailored for linear optical neural networks, providing a novel approach to applying NTK-based methods in cutting-edge neural network architectures.

3.2.1 Neural Tangent Kernel

We adopt the definition of the NTK based on previous works [86, 87, 88]. Let $F(\boldsymbol{\theta}, \boldsymbol{x})$ represent the output of a neural network, where $\boldsymbol{\theta}$ denotes the network parameters, $\boldsymbol{x}$ is the input and $\boldsymbol{y}$ is the ground truth. Given a training dataset $(\boldsymbol{x}, \boldsymbol{y}) = \{x_i, y_i\}_{i=1}^n$, we consider training the neural network by minimizing the squared loss over the training data. The loss ℓ is defined as

$$\ell(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(F(\boldsymbol{\theta}, x_i) - y_i)^2. \tag{3.3}$$

The parameters $\boldsymbol{\theta}$ evolve according to the differential equation

$$\frac{d\boldsymbol{\theta}(\tau)}{d\tau} = -\nabla\ell(\boldsymbol{\theta}(\tau)) = -\sum_{i=1}^{n}\left(F(\boldsymbol{\theta}(\tau), x_i) - y_i\right)\frac{\partial F(\boldsymbol{\theta}(\tau), x_i)}{\partial \boldsymbol{\theta}}, \tag{3.4}$$

where $\tau \geq 0$ is a continuous time index (iterations). Under Eq. 3.4, the evolution of the network output $F(\boldsymbol{\theta}(\tau), x_i)$ can be written as

$$\frac{dF(\boldsymbol{\theta}(\tau), x_i)}{d\tau} = -\sum_{j=1}^{n}(F(\boldsymbol{\theta}(\tau), x_j) - y_j)\left\langle \frac{\partial F(\boldsymbol{\theta}(\tau), x_i)}{\partial \boldsymbol{\theta}}, \frac{\partial F(\boldsymbol{\theta}(\tau), x_j)}{\partial \boldsymbol{\theta}} \right\rangle, \qquad \forall i \in [n]. \tag{3.5}$$

Since $u(\tau) = (F(\boldsymbol{\theta}(\tau), x_i))_{i\in[n]} \in \mathbb{R}^n$ is the network output on all x_i's at time t, and $y = (y_i)_{i\in[n]}$ is the desired output, Eq. 3.5 can be written more compactly as

$$\frac{d\boldsymbol{u}(\tau)}{d\tau} = -\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})(\boldsymbol{u}(\tau) - \boldsymbol{y}), \tag{3.6}$$

where the neural tangent kernel is

$$\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})_{i,j} = \left\langle \frac{\partial F(\boldsymbol{\theta}(\tau), \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}}, \frac{\partial F(\boldsymbol{\theta}(\tau), \boldsymbol{x}_j)}{\partial \boldsymbol{\theta}} \right\rangle. \tag{3.7}$$

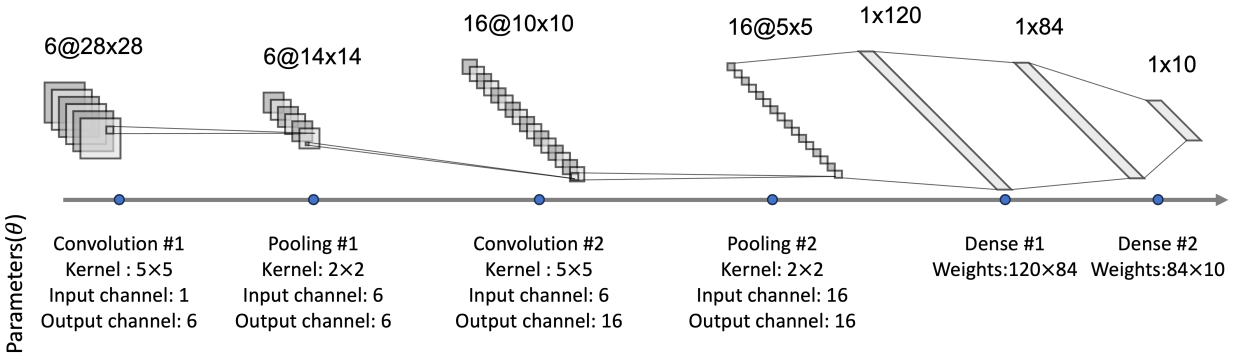


Figure 3.3: Illustrates the parameters $\boldsymbol{\theta}$ in the LeNet-5 architecture. Input Layer: Grayscale images of size 32×32. Convolution 1: Contains 6 filters, each of size 5×5, with a stride of 1 and valid padding, resulting in an output of size $28 \times 28 \times 6$. Pooling 1: Applies average pooling with a 2×2 filter and stride of 2, reducing the output size to $14 \times 14 \times 6$. Convolution 2: Consists of 16 filters of size 5×5, with a stride of 1 and valid padding, producing an output of $10 \times 10 \times 16$. Pooling 2: Applies average pooling with a 2×2 filter and stride of 2, reducing the output size to $5 \times 5 \times 16$. Dense 1 (Fully Connected Layer): Flattens the previous output and connects it to 120 units, followed by an output of 84 units. Dense 2: Connects to 84 units with an output of 10 units for final classification.

Using LeNet as an example (as illustrated in Figure 3.3) to calculate the NTK, the NTK $\mathcal{K}_\tau(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is computed using the Jacobian matrices $\frac{\partial F(\boldsymbol{\theta}(\tau), \boldsymbol{x}_1)}{\partial \boldsymbol{\theta}}$ and $\frac{\partial F(\boldsymbol{\theta}(\tau), \boldsymbol{x}_2)}{\partial \boldsymbol{\theta}}$ for two input sets $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Each Jacobian has dimensions corresponding to the number of inputs and the total number of network parameters. For N inputs in $\boldsymbol{x}_1$ and M inputs in $\boldsymbol{x}_2$, the NTK matrix size is $N \times M$, and the number of parameters is P. In LeNet, the total number of parameters, including those from both convolutional and fully connected layers, is $P = 9760$, while d represents the output size. The first Jacobian has dimensions $N \times P \times d$, and the second Jacobian has dimensions $M \times P \times d$. Consequently, the NTK matrix has a size of $N \times M$, where each entry corresponds to the pairwise inner product of the respective rows in the Jacobians.

3.2.2 Neural Tangent Kernel Property

We summarize key theorems of the NTK from previous theoretical works [89, 86]. As the layer widths approach *infinity*, the NTK exhibits the following properties. At initialization, the NTK becomes deterministic, meaning it is independent of the specific initialization values and is determined solely by the network architecture [86, 88]. Moreover, the parameters of the neural network and the NTK change very little throughout the training process [86, 88]. We also offer the proof in Appendix. The mathematical formulation of the NTK Theorem is as follows

For a network of depth l at initialization and in the limit as the layer widths m, and $m_1, \ldots, m_l \to \infty$, the NTK $\mathcal{K}$ converges in probability to a deterministic limiting kernel and this kernel remains constant throughout the training process. The detailed proof of the deterministic nature of the NTK is also provided in Appendix A.2.

$$\mathcal{K}_{\tau=0}(\boldsymbol{x}, \boldsymbol{x}) \to \mathcal{K}_{\tau=\infty}(\boldsymbol{x}, \boldsymbol{x}). \tag{3.8}$$

To verify the NTK theorem, we implement a neural network for regression tasks. In this example, we demonstrate linear fitting using two data points: $(-3, 2)$ and $(0.5, -1)$. This task demonstrates how different network widths affect weight updates during training. The network is a fully connected neural network with two hidden layers and a ReLU activation function. We denote the network as $f(\text{width} = m, \text{bias} = b, \text{hidden\_layers} = 2)$. The width m of the network is varied across three configurations: $m = 10$, $m = 100$, and $m = 1000$, allowing us to observe how increasing the network's capacity influences the magnitude of weight updates.

In Figure 3.4, the experimental results confirm the theoretical prediction that as the width of the network increases, the norm of the weight changes decreases. For $m = 10$, we observe substantial changes in the weights during training. As the width increases to $m = 100$, the

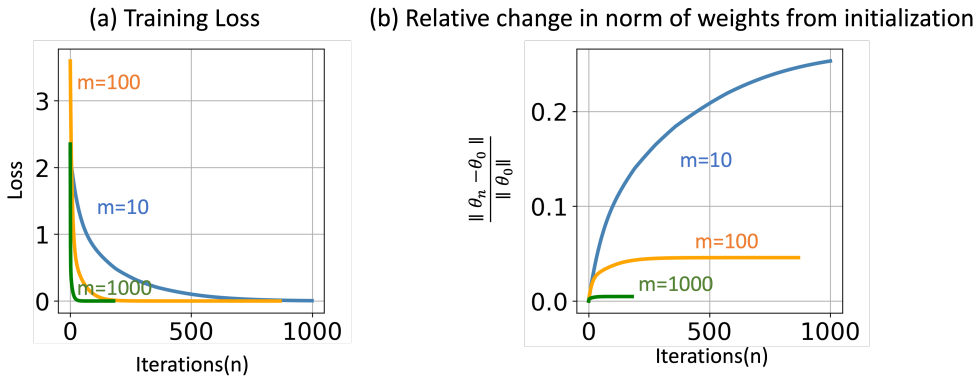


Figure 3.4: Weight Change Comparison for Different Network Widths: We compare the norm of weight changes during training for three different network widths $m = 10$, $m = 100$, and $m = 1000$. As the width increases, the weight updates become smaller, with the network experiencing the least change at $m = 1000$.

norm of the weight change decreases, but noticeable adjustments are still present. However, at $m = 1000$, the weight changes are minimal, and the norm of the weight updates is significantly lower than in the smaller configurations. These results provide evidence that, as the network width approaches infinity, the weights remain nearly static, validating the hypothesis derived from NTK theory.

3.2.3 Neural Tangent Kernel for Finite-Width Neural Networks

While NTK theory provides theoretical support for the *infinite-width* limit, practical neural networks operate in *finite-width* conditions. Under these real-world constraints, the property that the NTK remains constant during training no longer holds.

In the *finite-width* networks, the NTK evolves and grows over time, as illustrated in Figure 3.5. This shift from static to evolving NTKs introduces the need for deeper investigation into their behavior in finite-width settings. NTK theory still offers valuable insights, demonstrating that *infinite-width* neural networks essentially perform kernel regression, where

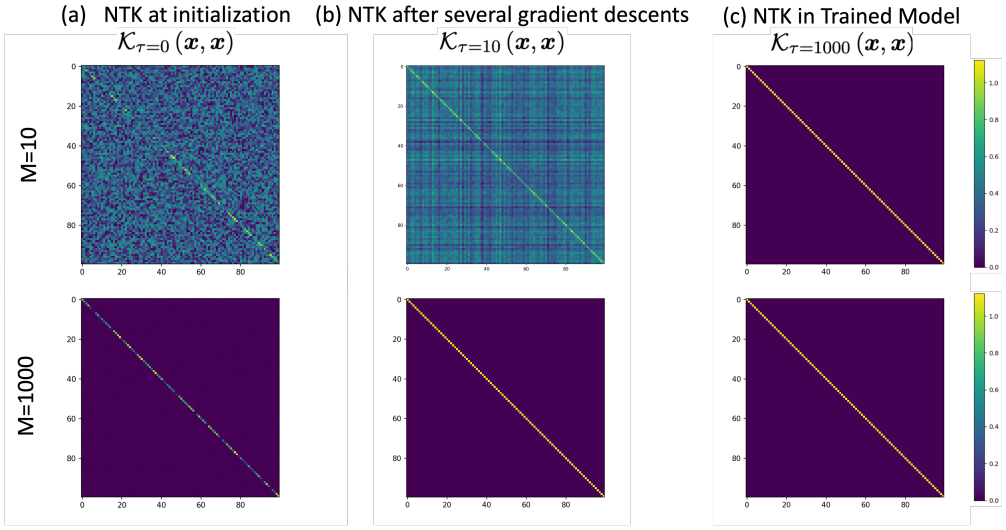


Figure 3.5: Illustration of NTK evolution for finite-width and wide-width neural networks. Top: In the finite-width case (2-layer network with width m = 10), the NTK grows significantly over time. Bottom: In the wide-width case (2-layer network with width m = 1000), the NTK remains almost constant during training. The training dataset consists of 100 images from MNIST.

the NTK serves as the kernel. We investigate this evolution and provide an investigation of the training dynamics in finite-width neural networks.

Infinite-width condition: We begin by considering the dynamics in Eq. 3.6, which are identical to those of kernel regression under gradient flow (assuming $\boldsymbol{u}(0) = 0$) [90]. The NTK is almost unchanged in the limit of infinite width, i.e., $\mathcal{K}_0 = \mathcal{K}_\infty$. Thus, we can directly use the closed-form solution for this network $F(\boldsymbol{x})_\infty$ with kernel regression for any testing input $\boldsymbol{x}'$

$$F(\boldsymbol{x})_\infty = \mathcal{K}_0(\boldsymbol{x},\boldsymbol{x}')\mathcal{K}_0^{-1}(\boldsymbol{x},\boldsymbol{x})\boldsymbol{y}, \textit{ where } \theta_0 = \theta_\infty, \textit{ under infinite-width limits,} \tag{3.9}$$

where $\mathcal{K}_0(\boldsymbol{x},\boldsymbol{x}')$ represents the kernel matrix between the testing input $\boldsymbol{x}'$ and the training inputs $\boldsymbol{x}$, and $\boldsymbol{y}$ is the vector of training labels. Then, we obtain training predictions when

we set the inputs to be the training data points, i.e., $\boldsymbol{x}' = \boldsymbol{x}$,

$$F(\boldsymbol{x})_\infty = \mathcal{K}_0(\boldsymbol{x}, \boldsymbol{x})\mathcal{K}_0^{-1}(\boldsymbol{x}, \boldsymbol{x})\boldsymbol{y}, \textit{where } \theta_0 = \theta_\infty, \textit{ under infinite-width limits.} \tag{3.10}$$

Finite-width condition: For a **trained** finite-width neural network, the output at time τ for any testing input $\boldsymbol{x}'$ can be expressed in a similar manner. Assuming the parameters θ_τ at time τ have converged and no longer change ($\theta_\tau \to \theta_{\tau\to\infty}$), the NTK in this finite-width neural network becomes constant. Under this assumption, the prediction can also be obtained using kernel regression. Thus, the output of a **trained** neural network at time τ for any testing input $\boldsymbol{x}'$ is

$$F(\boldsymbol{\theta}(\tau), \boldsymbol{x}') = \mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x}')^T\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})^{-1}\boldsymbol{y}, \textit{where } \theta_\tau = \theta_\infty, \textit{ under finite-width limits.} \tag{3.11}$$

From Eq. 3.11, we obtain training predictions when we set the inputs to be the training data points, i.e., $\boldsymbol{x}' = \boldsymbol{x}$,

$$F(\boldsymbol{\theta}(\tau), \boldsymbol{x}) = \mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})^T\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})^{-1}\boldsymbol{y}, \textit{where } \theta_\tau = \theta_\infty, \textit{ under finite-width limits.} \tag{3.12}$$

From Eqs. 3.6 and 3.12, we observe that the only variable in training predictions is the kernel function $\mathcal{K}_\tau(\boldsymbol{x}, \boldsymbol{x})$ determined by the Jacobian matrix $\frac{\partial F(\boldsymbol{\theta}(\tau), x)}{\partial \boldsymbol{\theta}}$. Therefore, the Jacobian matrix solely determines the training output $\boldsymbol{u}(\tau)$. This motivates us to employ the Jacobian matrix from the teacher model to assist with training a student model. Since the teacher model does not need to be retrained, its NTK and the Jacobian matrix are both **deterministic**. In this case, the teacher labels are the teacher model's NTK and student predictions are the student model's NTK. Based on this observation, we introduce a novel loss for teacher-student structure transfer learning.

3.2.4 Tangent Kernel Loss

We transfer the trained teacher model $F_{teacher}$ by regulating the neural tangent kernel in the student model $F_{student}$ with GTK loss. When training the student model with a trained

teacher model, we minimize the difference between their outputs to transfer knowledge from $F_{teacher}$ to $F_{student}$. This is described as

$$\min_{\boldsymbol{\theta}_{student}} \left(F_{teacher}(\boldsymbol{\theta}_{teacher}, \boldsymbol{x}), F_{student}(\boldsymbol{\theta}_{student}, \boldsymbol{x})\right). \tag{3.13}$$

Then we plug Eq. 3.12 into Eq. 3.13.

$$\min_{\boldsymbol{\theta}_{student}} \left(\mathcal{K}_{teacher}(\boldsymbol{x}, \boldsymbol{x})^T \mathcal{K}_{teacher}(\boldsymbol{x}, \boldsymbol{x})^{-1} \boldsymbol{y}, \mathcal{K}_{student}(\boldsymbol{x}, \boldsymbol{x})^T \mathcal{K}_{student}(\boldsymbol{x}, \boldsymbol{x})^{-1} \boldsymbol{y}\right), \tag{3.14}$$

where the neural tangent kernel $\mathcal{K}_{teacher}$ and the Jacobian matrix $\frac{\partial F_{teacher}(\boldsymbol{x}, \boldsymbol{\theta}_{teacher})}{\partial \boldsymbol{\theta}}$ from the teacher model are predetermined. The only variable is the NTK function $\mathcal{K}_{student}(\boldsymbol{x}, \boldsymbol{x})$ from the student model. To further simply Eq. 3.14, We note that the Jacobian matrix $\frac{\partial F_{student}(\boldsymbol{x}, \boldsymbol{\theta}_{student})}{\partial \boldsymbol{\theta}}$ solely determines the NTK $\mathcal{K}_{student}(\boldsymbol{x}, \boldsymbol{x})$ and obtain Eq. 3.16, i.e., minimization of the cosine distance between two Jacobian matrices $\frac{\partial F_{teacher}(\boldsymbol{x}, \boldsymbol{\theta}_{teacher})}{\partial \boldsymbol{\theta}}$ and $\frac{\partial F_{student}(\boldsymbol{x}, \boldsymbol{\theta}_{student})}{\partial \boldsymbol{\theta}}$.

$$\min_{\boldsymbol{\theta}_{student}} \left(\mathcal{K}_{teacher}(\boldsymbol{x}, \boldsymbol{x}), \mathcal{K}_{student}(\boldsymbol{x}, \boldsymbol{x})\right) \tag{3.15}$$

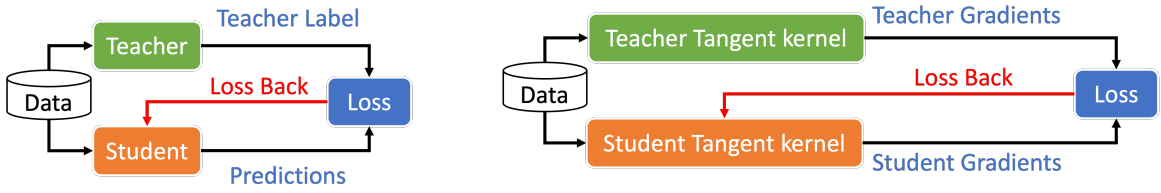
$$\Rightarrow \min_{\boldsymbol{\theta}_{student}} \left(\frac{\partial F_{teacher}(\boldsymbol{\theta}_{teacher}, \boldsymbol{x})}{\partial \boldsymbol{\theta}}, \frac{\partial F_{student}(\boldsymbol{\theta}_{student}, \boldsymbol{x})}{\partial \boldsymbol{\theta}}\right), \tag{3.16}$$

Here, we select the **cosine distance loss** to minimize differences between NTKs, as recent research demonstrates that cosine distance loss effectively captures the distance between two NTKs and reduces discrepancies [91]. Therefore, the kernel-based objective function is

$$\mathcal{L}_{GTK} = \min_{\boldsymbol{\theta}_{student}} \left(1 - \frac{\langle G_{teacher}, G_{student} \rangle}{\|G_{teacher}\| \|G_{student}\|}\right), \tag{3.17}$$

Where $G_{teacher} = \frac{\partial F_{teacher}(\boldsymbol{\theta}_{teacher}, \boldsymbol{x})}{\partial \boldsymbol{\theta}}$ and $G_{student} = \frac{\partial F_{student}(\boldsymbol{\theta}_{student}, \boldsymbol{x})}{\partial \boldsymbol{\theta}}$.

We also show the comparison between KD and GTK loss in Figure 3.6. In the traditional KD method, the teacher model generates soft labels from the data, and the student model learns by minimizing the loss between its own predictions and the teacher's labels. This focuses on matching output but does not account for the underlying training dynamics. On the other hand, in our GTK loss approach, both the teacher and student models leverage their respective neural tangent kernels to capture gradient information. Rather than matching



(a) Knowledge Distillation Temperature Loss

(b) Gradient Tangent Kernel Loss for Knowledge Transfer Learning

Figure 3.6: Comparison between knowledge distillation processes. Left: In the traditional KD method, the teacher model generates labels based on data, and the student model learns by minimizing the loss between its predictions and the teacher's labels. Right: In our GTK loss approach, both the teacher and student models use their neural tangent kernels to capture gradient information ($G_{teacher}, G_{student}$). The student minimizes the loss by aligning its gradients with those of the teacher, enhancing training efficiency through this gradient-based distillation process.

predictions, the student model minimizes the loss by aligning its gradients with those of the teacher, as described in Eq. 3.17. This gradient-based distillation enhances training efficiency by better aligning the dynamics of the student with those of the teacher.

Chapter 4

KNOWLEDGE TRANSFER TO 4F-BASED OPTICAL NEURAL NETWORKS

4.1 Motivation

Convolutional Neural Network (CNN) architectures are well known for their ability to compute visual tasks [92, 93, 94, 33, 95]. Many of these tasks require fast processing of real-time signals. In autonomous navigation, for example, a network must be capable of identify obstacles with different textures and lighting conditions in real time. While CNNs are instrumental in providing high accuracy for such tasks, the time that it takes for the input to propagate through the trained network (forward propagation time), is large and precludes real-time operation. The main reason for such inefficiency is the computational complexity of CNNs, which is $\mathcal{O}(HWk^2)$, where H and W are the height and width of an image frame and $k^2 = k \times k$ is the size of the convolutional kernel. The challenge of enhancing the computational performance has driven significant development of electronic hardware that is dedicated to computing convolutions, with graphics processing units (GPUs) and tensor processing units (TPUs), that deliver an order of magnitude acceleration. Even with such dedicated hardware, however, effective computation times are still sub-optimal and become too large for many applications, especially when high-resolution inputs are processed. For example, when applied to ResNet-18 for autonomous navigation, the Jetson Nano, NVIDIA hardware development kit, achieves at most 5 frames per second (fps) rate for images of dimensions 1000×1000 [96].

Since convolution is the most time-consuming operation in CNNs, a possible gain in computational efficiency can be achieved by implementing the CNN in the Fourier domain (spectral domain) [97, 98]. The Fast Fourier Transform (FFT) operation has the complexity

of $\mathcal{O}(HWlog(HW)$ for the images and the kernels. In the Fourier domain, the convolution is transformed into an elementwise product with only $\mathcal{O}(HW)$ operations. While promising, the approach does not boost the forward propagation time in practice. Due to nonlinearities that follow most of the layers in a CNN, the spectral approach ends up including a large number of costly forward and inverse Fourier transforms between successive layers. Optimizations of network architectures to be compatible with spectral operations were proposed, such as FCNN and Clebsch–Gordan Nets [99, 100]. These architectures, however, appear to suffer from a reduction in classification accuracy for complex visual tasks. For example, FCNN reaches less than half of state-of-the-art accuracy on CIFAR-10. Another branch of research developed in parallel is the introduction of spectral pooling layers for the purpose of adapting the spatially applied Max-Pooling operation to the spectral domain [101, 102, 103]. These networks still entail nonlinear activations that require conversions between the spatial and spectral domains. Removing these nonlinear functions dramatically reduces the computational complexity of spectral CNNs, which comes at the cost of reduced performance.

Another promising approach for accelerating computation is the development of optical neural networks (ONNs) that could replace electronic hardware [104]. ONNs utilize the inherent parallelism of light, which enables passive manipulation of massive amounts of two-dimensional data at the speed of light [37, 38, 39, 29]. Thus, ONNs can offer time of computation that is nearly instantaneous in comparison to those provided by the best electronic hardware available. Specifically, a lens can perform a Fourier transform with $\mathcal{O}(1)$ complexity [105]. In recent years, this promise has resulted in various ONNs for MNIST classification based on a sequence of diffractive masks in the Terahertz regime [39], vector-matrix multiplication using integrated photonics [106, 107], and hybrid optical-electronic networks leveraging the inherent Fourier transform property of lenses exploited in a $4f$ architecture [29]. While impressive in their own right, the demonstrated ONNs to date have been limited in terms of the complexity of the scenes on which they operate, which have mostly been limited to low-resolution images with simple features like MNIST digits. When applied to more complex scenes (e.g., CIFAR-10), these implementations exhibit low

classification accuracy.

A major contributing factor to these limitations is that these networks lack the optical implementation of a nonlinear activation and were mostly constrained to linear operations in the optical domain [29]. Achieving an optical nonlinearity requires very large optical power, a high-quality factor resonator, exotic materials, or a combination thereof. While some nonlinear functions are readily available, such as the square operation imparted by a detector, it is unclear how effective it is for achieving comparable performance to that of the ReLU nonlinearity, which is the most common nonlinear activation for CNNs [108, 46]. If the nonlinearity is implemented electronically, as a subsequent layer, the electronic signal needs to be converted back to the optical domain to enable additional optical processing. Such repeated signal transductions significantly increase the power consumption and latency, thus obviating the benefits of an ONN versus a more traditional electronic implementation [29].

To compensate for the lack of nonlinearity in an ONN, as well as to find an optimal level of performance for a fully linear spectral network, here we develop a Knowledge Distillation (KD) training methodology to transfer the information from a nonlinear network (teacher) to a SCLC (student). Originally, KD training was introduced for pruning of networks, i.e., knowledge from a large teacher network is transferred to a less complex student model [63, 109]. Trained with the KD approach, the student network typically converges faster and obtains better performance than it would achieve without the KD training. A common example for successful KD training is object classification in images. In this problem, the teacher classifies images and provides "soft labels" to the student during training along with the actual labels. A Kullback-Leibler (KL) divergence loss between the soft labels and the student model predictions is then optimized to take into account the teacher's predictions [78]. KD training is a generic approach and was applied to a variety of problems such as semantic segmentation in which soft labels are used for classification of each pixel [110, 111].

In this chapter, we adapted the KD training framework to circumvent the need for nonlinearity by "distilling the nonlinearity" from the nonlinear, teacher CNN, to the linear counterpart SCLC, which can be easily implemented using free-space optics. We find that

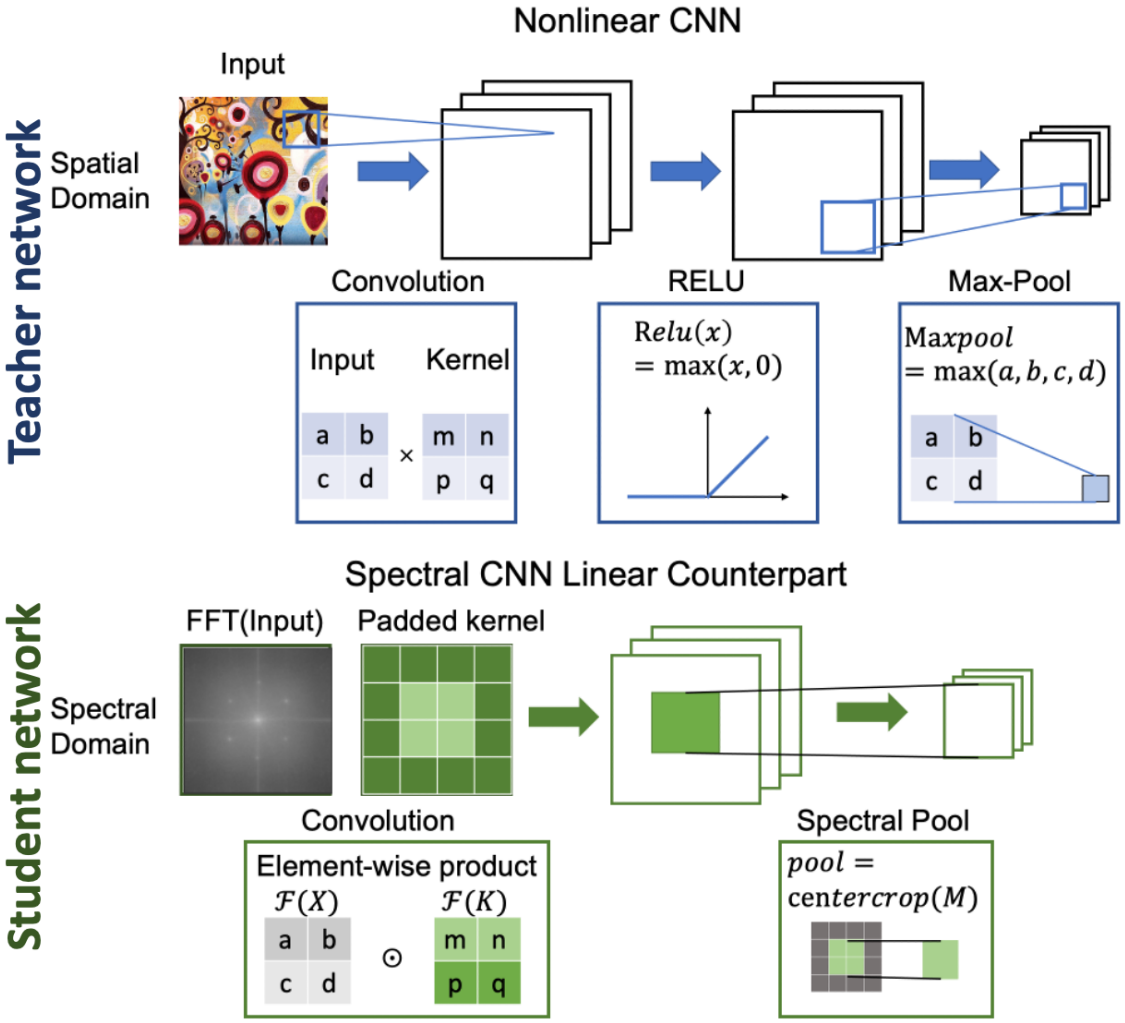


Figure 4.1: Nonlinear CNN (top) and the proposed substitute, Spectral CNN Linear Counterpart (SCLC) (bottom). Top: An example of nonlinear CNN with layers that include operations of Convolution, Nonlinear RELU activation, and Max-Pool. Bottom: SCLC of the CNN shown in top row. The convolution corresponds to the elementwise product in the spectral domain. The RELU operation is excluded. The Max-Pool layer is represented by a center-crop operation in the spectral domain. (Figure from [JX1])

for boosting student accuracy the teacher and the student networks are required to be as architecturally similar as possible. The KD approach allows the student network to

achieve state-of-art performance, exceeding that of previous training methods or networks in the spectral domain, and is also easily amenable to optical implementation because light propagation can be naturally described in Fourier space. The adapted KD training enables us to design a hybrid optical-electronic architecture for the SCLC network, where the optics serve as a linear frontend processing unit connected to an electronic backend that typically includes the last layer that corresponds to a specific task. We train this student network with KD training and demonstrate the performance of such a network on two common problems in which CNNs are leading computational methods: object classification and object segmentation. We show that the KD-trained SCLC can achieve performance easily surpassing that of a linear network trained with a standard training approach and nearing the performance of the nonlinear network.

4.2 Methods

We propose to construct a Spectral CNN Linear Counterpart (SCLC), the "Student Network", from a spectral nonlinear CNN, the "Teacher Network", based on a free-space optical $4f$ architecture. The "Teacher Network" takes the shape of a common CNN designed for generic tasks, for example, image classification or object segmentation, as demonstrated in Fig. 4.1.

In the case of object segmentation, we consider a CNN of a "U" shape [112], where the input passes through similar operations of Convolution, ReLU, and Max-Pool, called convolution layers, and in addition, the output of each Convolutional layer contracts the input dimension to a "bottle-neck" layer, called skip connections, from which the representation is expanded with a set of inverse operations, such as transposed Convolution, ReLU, and Max-Pool, altogether called transposed convolution layers. To measure the performance of the proposed architectures, we concatenate them with a backend that corresponds to either a classification task (softmax fully connected backend layer) or a segmentation task (sequence of transposed Convolution backend layers).

To obtain a network model that is readily realizable with optical components, the network requires a conversion to the spectral domain, such that the input into the model is the

Fourier transform of the input image and the operations such as convolution and pooling are implemented in the spectral domain. Furthermore, the model should not include nonlinearities since computing those will require an inverse transform and transition from optical to electronic components. Moreover, as explained the nonlinear activations will be difficult to implement optically.

4.2.1 Convolution in SCLC

The convolution is performed as the elementwise product between the input images represented in the spectral domain and the kernels padded to the same dimensions as the inputs. The input into the convolution is a four-dimensional tensor (S: batch size, C: number of input channel, H: height of the image, W: width of the image), and the 2D convolution operation with a stride of 1 is calculated as

$$y(i,j) = x * k = \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} x(h,w) \times k(i-h, j-w) \tag{4.1}$$

where x is the input image, k is the kernel and $*$ indicates the convolution operation. We implement the convolution in SCLC in the spectral domain by using an FFT and an elementwise product

$$Y = \mathcal{F}(x * k) = \mathcal{F}(x) \odot \mathcal{F}(k) \tag{4.2}$$

The function $\mathcal{F}$ denotes the FFT operation, and $\odot$ indicates the elementwise product, which requires the input and the kernels to have the same dimension $H \times W$. During training, the spectral convolution kernel is updated according to

$$\begin{aligned} \sigma_X &= \nabla_X \mathcal{L}|_{X=X_0} = \sigma_Y \odot K_0 \\ \delta_k &= \nabla_K \mathcal{L}|_{K=\mathcal{F}(0)} = \sigma_Y \odot X_0 \\ k_1 &= k_0 + \lambda[\mathcal{F}^{-1}(\delta_K)], \end{aligned} \tag{4.3}$$

where $\sigma_X(\sigma_Y)$ is the error from the previous (next) layer; X_0 and K_0 are, respectively, the input and kernel in the forward propagation; $\nabla_K(\nabla_X)$ is the gradient operator w.r.t. the

kernel (input); and $\mathcal{L}$ is the loss function. The updates to the elements in the kernel are computed by applying an inverse FFT (i-FFT) and multiplying by the learning rate λ.

Once the network has been trained, the implementation with spectral convolution components has significant benefits for inference latency for a given input (forward propagation). Indeed, assuming an input image size of (H, W) and square kernel size of (k, k), the complexity of one spatial convolution in the image domain is $\mathcal{O}(HWk^2)$. On the other hand, the spectral convolution composed of elementwise products is of complexity $\mathcal{O}(HW)$. In the optical setup, elementwise operations can be performed in parallel and thus the runtime, being independent of H and W, is further reduced to $\mathcal{O}(1)$. In addition, the computational complexity of an FFT is $\mathcal{O}(HWlog(HW))$, which brings the overall complexity for the spectral convolution to be $\mathcal{O}(HWlog(HW))$. As the proposed network is in the spectral domain, the FFT and i-FFT transforms are needed to be applied at the beginning and at the end of the network for images only, such that the complexity of the intermediate layers will only apply FFT to kennels, which greatly reduces the transformation times. Additionally, significant acceleration occurs in the optical implementation since the spectral transforms can be achieved through phase transforming components at the speed of light. The combination of convolution via forward and inverse transforms in optical domain would correspond to instantaneous running time of $\mathcal{O}(1)$, when compare to electronic running time for all layers of the network.

4.2.2 Pooling in SCLC

Pooling in the spectral domain needs to take into consideration both mimicking the effect of the standard Max-Pool used in CNNs and to be practically implementable with optical components. In a nonlinear CNN, common pooling operations are Average-Pooling or Max-Pooling, which select the average of the elements or the maximum element, respectively, from the region of the feature map covered by the pooling filter shown in Fig. 4.1. The purpose of the pooling layers is to reduce the dimensions of the feature maps and to find the representative elements to be transferred forward from the feature map. We propose to substitute the Max-Pool with a different pooling function, called “Spectral pool”, which will

enable similar functionality in the spectral domain. In particular, we propose to replace the pooling layer with a linear, low-pass filter in the spectral domain. The forward and backward propagation in the layer are defined as

$$y = \mathcal{F}^{-1}(CROP(\mathcal{F}(x), (H', W'))) \tag{4.4}$$

$$z = \mathcal{F}^{-1}(PAD(\mathcal{F}(y/x^*), (H, W))) \tag{4.5}$$

where the $CROP$ operation in forward propagating keeps the center part of the spectral feature map and reduce the total size from (H, W) to (H', W'). The PAD operation in backward propagation matches the dimension of gradients outputs from (H', W') to (H, W) by padding zeros instead of cropped elements. The $\mathcal{F}$ and $\mathcal{F}^{-1}$are applied if the networks are in spatial domain. Those operations are similar to max pooling since the idea of Max-Pool is to find the representative element in each kernel, while it goes over the feature map and effectively selects the most descriptive features. Spectral-Pool drops the high frequencies in the Fourier domain such that it achieves a less noisy output and captures the dominant features, similar to the effect of the conventional Max-Pool operation. It was shown that training with spectral pooling has better convergence properties compared to Max-Pooling. Furthermore, a spectral pooling is of lower computational complexity and is inherently amenable to optical implementation with $\mathcal{O}(1)$ complexity, see Table 4.1.

Typically, an activation function, such as ReLU, tanh, or σ is applied to extract the features after convolution to avoid extremities in neural units values and maintain network stability. These nonlinearities, however, are nearly impossible to achieve in a practical optical implementation. Thus, the SCLC skips the nonlinear activation function to make the network structure amenable to optical implementation. The lack of nonlinearity will be circumvented by the application of KD to maximize the performance achieved using linear operations only.

4.2.3 Knowledge Distillation in SCLC

KD training requires a teacher and student model both performing the same task and exhibiting similarity in their architectures. In previous applications, KD training was applied

Table 4.1: Comparison of forward propagation computational complexity between Nonlinear CNN, SCLC implemented on electronic hardware, SCLC implemented on optical hardware. (Table from [JX1])

Structure	Nonlinear CNN	SCLC (Computational)	OSCLC (Optical)
Convolution layers	$\mathcal{O}(HWk^2)$	Spatial: $\mathcal{O}(HW\log(HW))$ Spectral: $\mathcal{O}(HW\log(HW))$	$\mathcal{O}(1)$
Pooling layers	$\mathcal{O}(HWk^2)$	Spatial: $\mathcal{O}(k^2\log(k^2))$ Spectral: $\mathcal{O}(1)$	$\mathcal{O}(1)$

to model pruning, where the teacher and the student models have exactly the same structure with the student having a fraction of the units of the teacher. Here, we extend KD application and consider the teacher and the student models with a similar number of units; however, the student is an adapted version of the teacher operations without the nonlinear activation and has a revised pooling operator. In particular, the teacher model is a pre-trained nonlinear CNN, which is a state-of-the-art system for that particular task. The student network is the SCLC that corresponds to that system with the architectural changes described in the previous sections. The objective of KD training is to optimize the total loss subject to updating the weights, Φ, of the SCLC student model only. The total loss is a linear combination of the Temperature loss and the Student loss. The weights are updated according to implementation of back-propagation that optimizes the total loss.

We select the Temperature loss to be the KL function between the soft labels ($p^{sl,t}$) from the pre-trained nonlinear CNN model (*teacher*) and predictions ($p^{sl,s}$) from SCLC (*student*) both distilled with the same temperature T. We chose KL loss over cross-entropy because it includes an extra penalty on the direction of the loss, which facilitates convergence. The student loss is the standard cross-entropy loss between the data labels and SCLC probabilities

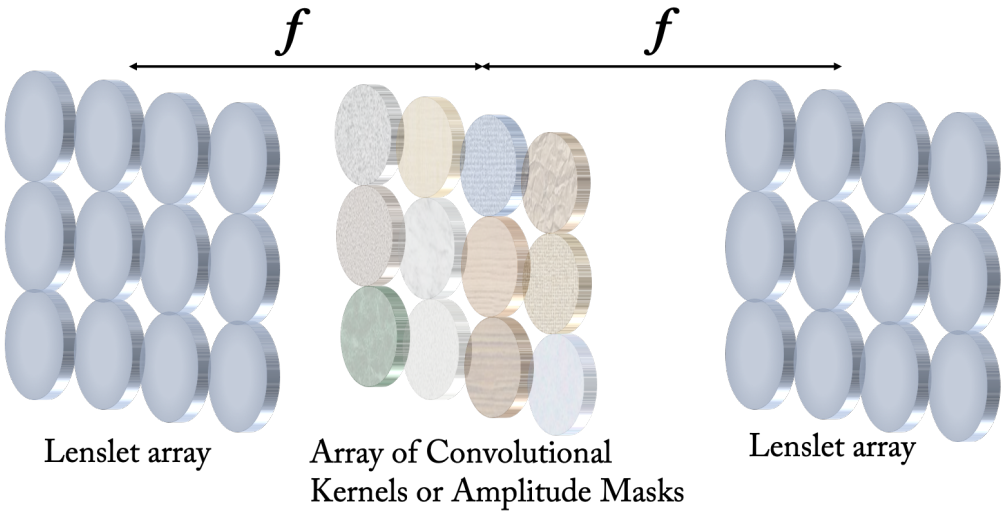


Figure 4.2: Using a lenslet array and array of phase and amplitude masks, we can implement all the convolutions in a parallel fashion. We can also implement Spectral Pooling in such a structure by placing an amplitude mask in the Fourier plane. Both the lenslet array and convolutional masks can be implemented using meta-optics. (Figure from [JX1])

(p^{hl}). The total loss is then calculated as a weighted summation of the two losses

$$Loss(x, \Phi) = \alpha \mathcal{L}_C(y, p^{hl}) + (1 - \alpha)\mathcal{L}_K((p^{sl,t}; T = \tau), (p^{sl,s}; T = \tau)), \tag{4.6}$$

where x corresponds to the input, y is the training data, Φ are student model weights, $\mathcal{L}_C$ is the cross-entropy loss function, $\mathcal{L}_K$ is the KL divergence loss function, p^{hl} corresponds to the student model hard predictions, $p^{sl,s}$ corresponds to the student predictions under given teacher model probabilities $p^{sl,t}$, and α is the weighting parameter.

4.2.4 Optical Implementation of SCLC (OSCLC)

In this section, we explore how the proposed SCLC network can potentially be implemented using free-space optics, i.e. Optical SCLC (OSCLC). The choice of free-space optics is motivated by the large number of information channels available to us in such an implemen-

tation [46]. There are three components of the OSCLC that need to be considered for optical implementation: convolutional layers, spectral pooling, and summation of different channels.

Each convolutional layer can be implemented by a $4f$ correlator architecture to further accelerate the forward propagation speed (Figure 4.2) [37, 113, 114, 115]. A typical $4f$ correlator architecture comprises two lenses of equal focal length spaced apart at $2f$ distance and with input and output planes located in the front and the back focal planes of the first and second lenses respectively. The first lens produces a Fourier transform of the input scene at the focal plane. A complex-values phase-mask placed in that focal plane provides the point-wise multiplication implementing the convolution and the second lens performs the inverse Fourier transform. Therefore, a $4f$ correlator architecture is able to function as an equivalent architecture for a single channel of a linear spectral CNN counterpart. We note that using coherent light, and phase-only spatial light modulators, we can handle the negative weights. By creating a lenslet array and an array of convolutional phase-masks we can parallelize all the convolutional operations. Thus the computational complexity decreases to $\mathcal{O}(1)$ for any operations in spectral domain and will not grow exponentially with the input image resolution/ pixels. Based on our simulation, with $1mm$ center-to-distance between $3mm$ focal length lenses in an array, we have negligible cross-talk between the channels[29]. The spectral pooling in the OSCLC can also be implemented by a low-pass filter, where we block high frequency components in the Fourier plane. Essentially, we can use a $4f$ correlator, with an amplitude mask in the Fourier plane. The highest frequency components in the image correspond to the largest wavevectors, which in the Fourier plane are distributed further from the optical axis. By placing a circular aperture in the Fourier plane, we will block frequency components that reside outside of the central aperture, enabling us to perform spectral pooling optically. This approach is also amenable to variants of our defined low pass spectral pooling operator. For example, we could implement a high-pass filter by inverting the trasmittance of the Fourier plane mask to be opaque at the center and transparent outside this region. Alternatively, if frequency components from the whole range at the Fourier plane are desired, we could instead use a mask based on concentric transparent annuli, which

corresponds to a spectral pooling operator that passes frequency components only at certain frequency intervals. An additional consideration that must be made with this architecture is the field of view of the lenses in the lenslet array. In order to limit crosstalk between the different convolution channels, the field of view will be constrained so that the convolutions at the output plane will be non-overlapping; this is achievable using an array of field stops positioned at the input plane.

The summation of different channels is also basic principle in deep neural networks and is widely used in object classification and segmentation. If there are no summations after elementwise products, the channels will grow exponentially, requiring a massive number of kernels after only a few convolutional layers, which makes it impossible for real optical implementations. Although fewer kernels in the convolution layers can alleviate this challenge, the overall accuracy, especially in complex scenarios, would suffer. Such a summation can be implemented using various techniques used for coherent beam combining [116, 117]. We note that, while free-space optics tends to be bulky and prone to misalignment, recent demonstrations of meta-optics and volume optics [118, 119] exhibit complicated free-space optics in a compact form factor, possibly in a monolithic fashion mitigating any misalignment. We note that, in the current paper, for simulation as assumed perfect alignment and aberration-free optics, which are ideal conditions, and future works will explore analysis of the proposed architecture with realistic optics and their robustness against experimental imperfections.

4.3 Experiments and Results

In the case of classification, we consider CNNs that are structured with multiple layers of repeating operations of Convolution, Nonlinearity (ReLU), and Max-Pool (selecting the maximum value). In the case of object segmentation, we consider a CNN of a "U" shape, where the input passes through similar operations of Convolution, ReLU, and Max-Pool, called convolution layers, and in addition, the output of each Convolutional layer contracts the input dimension to a "bottle-neck" layer, called skip connections, from which the representation is expanded with a set of inverse operations, such as transposed Convolution, ReLU, and

Max-Pool, altogether called transposed convolution layers To measure the performance of the proposed architectures, we concatenate them with a backend that corresponds to either a classification task (softmax fully connected backend layer) or a segmentation task (sequence of transposed Convolution backend layers).

Table 4.2: Forward propagation accuracy and processing-time of different network variants for classification tasks. AlexNet: nonlinear baseline network (no optical implementation), SCLC: The linear counterpart of AlexNet. (Table from [JX1])

Dataset	Model(input:224 × 224)	Accuracy	Runtime (ms/img)	Optical Runtime (ms/img)
Cats vs. Dogs	AlexNet	96.10%	350.66	-
	SCLC	79.50%	11.05	0.61
	SCLC + KD	90.60% (+11.10%)	11.05	0.61
	SQ-Nonlinear	51.70%	12.04	0.61
	SQ-Nonlinear+ KD	51.72%	12.04	0.61
Cifar-10	AlexNet	85.09%	350.66	-
	SCLC	65.45%	11.05	0.61
	SCLC+KD	80.80% (+15.35%)	11.05	0.61
High-10	AlexNet	93.95%	350.66	-
	SCLC	70.12%	11.05	0.61
	SCLC + KD	81.40% (+11.28%)	11.05	0.61

4.3.1 Object Classification Task

We evaluate the proposed SCLC on different classification datasets to estimate the accuracy that SCLC can achieve when trained with KD. We compare the accuracy to that of the teacher network, AlexNet. The first dataset that we consider is the Kaggle Cats and Dogs Challenge [120] which consists of 125000 images of dimensions 96×96 associated with two classes: a cat or a dog. Additional dataset that we consider is Cifar-10 [121] which consists of 60000 images of dimension of 32×32 including 10 classes. The third dataset that we consider

is High-10 which is a subset of ImageNet [122] and consists of approximately 10000 annotated images, equally distributed in 10 classes with images of resolution 500×300. During training of AlexNet all images are resized or cropped to dimensions of 224×224 to match AlexNet's input size. To evaluate the performance of SCLC in classification we add a single fully connected backend layer to it. The backend layer will be implemented in electronics for the OSCLC since the last layer will include nonlinearity. We employ the KD approach to train SCLC with AlexNet being the teacher network and when the training converges, we test SCLC variants against AlexNet.

In particular, we compare the SCLC trained with and without the KD approach and a variation of AlexNet with a square nonlinearity that could be realized optically using detectors. We show the results of the comparison for the three benchmarks in Table 4.2. We observe that for all benchmarks, KD training significantly enhances the accuracy of the classification achieved by the SCLC (by 12.5% on average). Indeed, KD contribution appears to be essential in generating an SCLC network with robust accuracy. On Kaggle's Cats and Dogs classification SCLC achieves 90.60% (6% below the accuracy of AlexNet), on Cifar-10 classification it achieves 80.80% (5% below the accuracy of AlexNet) and on HIGH-10 classification it achieves 81.4% ($\approx 12\%$ below the accuracy of AlexNet). These results are encouraging since SCLC trained with standard training has a much bigger gap of $16\%, 20\%, 23\%$ between its accuracy and the accuracy of AlexNet. The KD approach appears to close this gap by more than half for all benchmarks. Furthermore, when the RELU nonlinearity is modified to square nonlinearity (SQ-AlexNet) both KD and standard training do not result with sufficiently accurate network. This is because the square nonlinearity will magnify the parameters of the model. This observation indicates that the KD approach is effective in regimes where the network architecture of the student is kept as close to the teacher as possible.

Notably, to match AlexNet input dimensions, all experiments were implemented with the same input image resolution of 224×224. We therefore explore with HIGH-10 dataset (that includes higher resolution images) how accuracy varies if the resolution of the input is

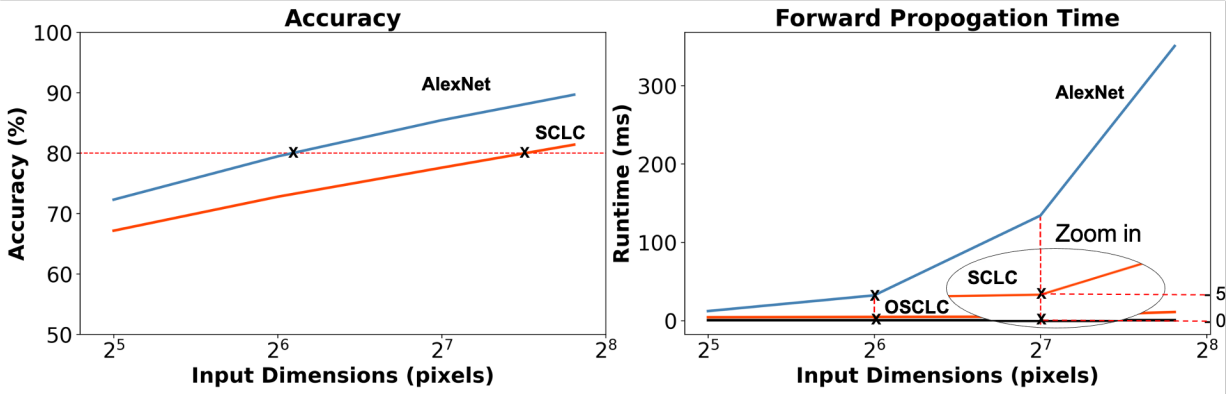


Figure 4.3: Accuracy and Forward propagation time for classification task on High-10 dataset. Left: AlexNet and SCLC accuracy for varying input resolution; Right: Forward propagation runtime of AlexNet, SCLC, and OSCLC for varying input resolution. (Figure from [JX1])

increased. For each input image resolution down sampling to 224, the teacher (AlexNet) is first trained and then the student (SCLC/OSCLC) is trained under the supervision of the teacher model. We show in Fig. 4.3 that increasing the resolution of the input increases the accuracy of both the teacher and student models (with a linear rate). SCLC trained with higher resolution inputs can surpass the accuracy of the teacher network trained with lower resolution dataset, e.g., SCLC with input of $2^7 \times 2^7$ dimensions performs similarly ($\approx 80\%$) to the teacher with input of $2^6 \times 2^6$ dimensions. While the accuracy of the two is similar, the runtime of SCLC is expected to be more favorable since teacher's runtime grows exponentially. It is impossible to continually increase the input dimension for nonlinear models. Higher resolution inputs require higher computational power and longer processing time, which makes it not capable for mobile GPUs or any instant usages. However, SCLC overcomes those by processing the networks in optical components. The computational efficiency and computational power increase at much lower rates or even constant when increasing the input dimension. Indeed, we show that for 80% accuracy both SCLC and OSCLC are at least 5 times faster than ($\approx 5ms$ and $\approx 1ms$) than AlexNet ($\approx 25ms$).

We further demonstrate the efficient runtime of SCLC compared to AlexNet in Table 4.2

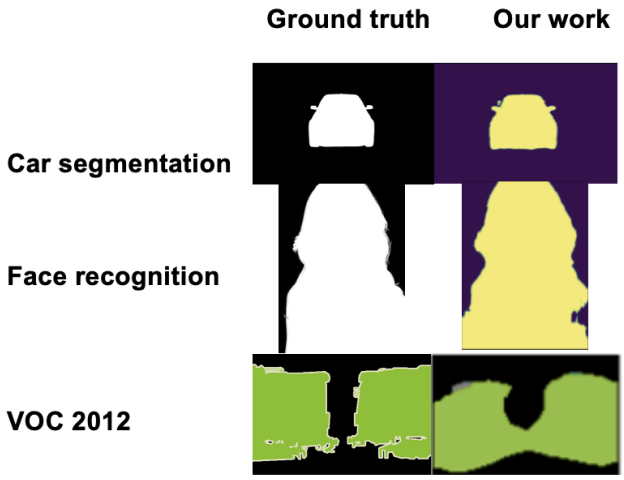


Figure 4.4: Examples of object segmentation for the considered benchmarks of (i) Car Segmentation, (ii) Face Recognition, and (iii) VOC2012. Left to right: Ground truth, Our work (SCLC/ OSCLC). (Figure from [JX1])

columns 3 and 4 for input size of 224×224. While AlexNet runtime for a single image is $350ms$, SCLC runtime drops by an order of magnitude to $11ms$ for a single image. Simulated runtime of OSCLC drops by another order of magnitude to $0.6ms$. The estimation of OSCLC forward propagation runtime includes three parts: 1) running time of the optical structure, 2) transduction time between optics and electronics, and 3) running time of the electronic backend. Since light propagation is very short, $\approx ps$, the main contribution comes from the signal transduction ($\approx 0.32ms$ for 100kb images via USB 3.0 protocol at a rate of 2500Mbit/s) and the backend propagation time ($\approx 0.28ms$ for a single image on a GPU (Tesla P100).

4.3.2 Object Segmentation Task

For object segmentation, a standard CNN is of a U-shape with a sequence of convolutions, transposed convolutions and skip connections. The teacher network is U-Net [112]. KD training corresponds to pixel-level loss for both soft predictions and soft labels. The frontend for this task is the sequence of contracting convolutions while the backend corresponds to a

sequence of transposed convolutions.

Kaggle's Carvana Image Masking Challenge that consists of 5088 cars with an original resolution of 1920×1280. The dataset is randomly split into 4580 and 508 images for training and testing, respectively. Face Recognition dataset consists of 2000 images with 1700 for training and 300 for testing. VOC2012 consists of 2913 images of original resolution 500×375, which includes 20 classes and one background class. The images in all datasets are down-sampled to 960×640 or smaller due to limitations of GPU memory.

The results are shown in Table 4.3. For both Car and Face datasets Iobserve that the SCLC performs relatively well and obtains accuracy that falls from that of the teacher U-Net by only a few percent. Iobserve that consideration of higher resolution inputs corresponds to enhanced accuracy in classification experiments. For the VOC2012 dataset, which includes lower resolution images and has more segmentation classes, both the teacher and SCLC perform with rather low accuracy below 80%. This example demonstrates that in such problems, it is crucial to consider the fully available image resolution. Icompare the computational efficiency of the SCLC against the teacher in terms of frames per second rate and find that the SCLC is approximately $2\times$ faster than U-Net. While such a speedup is more modest than in the image classification task, the rate is closer to a real-time operation rate (30 fps) or alternatively allows for consideration of larger input images that may correspond to enhanced segmentation with the same frame rate as that of the teacher. The main contribution to SCLC runtime is the electronic backend that consists of transposed convolution layers, which have a computational complexity of $\mathcal{O}(HWk^2)$ (compared to image classification backend, which is a fully connected single layer).

4.4 Discussion

The SCLC network is a multi-layered model in spectral space, where each layer consists of a matrix multiplication and a spectral pooling (prior to the backend). Theoretically, the convolution layers and their corresponding operation in the spectral space can be implemented as a single matrix that implements an elementwise product operation with the input. In

Table 4.3: Accuracy and forward propagation rate (frames per second) for object segmentation task tested on three benchmarks. (Table from [JX1])

Dataset	Network	Accuracy	Rate (higher is better)
Car Segmentation	U-Net	98.02%	7.36 fps
	OSCLC	97.1%	12.7 fps
Face Recognition	U-Net	95.58%	9.38fps
	OSCLC	91.39%	15.6fps
VOC2012	U-Net	75.51%	7.36fps
	OSCLC	62.21%	12.7fps

particular, the convolutional kernels in the spectral domain correspond to an elementwise product between the Fourier transform of the input and the Fourier transform of the kernel padded to the same size of the input image. The elementwise product yields a matrix for each layer and the product of the sequence of matrices generates a single matrix reflecting elementwise product with the input.

$$O = x \odot k_1 \odot ... \odot k_i \odot ... \odot k_n, i = 1, 2, ...n \tag{4.7}$$

Where O is the output, x is the input image and the k_i is the padded parameter in each layer. The spectral pooling is also reduced to a single operation where it crops the center part of this matrix.

$$O_{CROP} = CROP(O) = CROP(x \odot k_1 \odot ... \odot k_i \odot ... \odot k_n), i = 1, 2, ...n \tag{4.8}$$

Such a setup could be instrumental in implementation of the optical network (OSCLC) for operation on given inputs (i.e., forward inference). Notably, such a structure would not be applicable during training. The reason stems from the fact that each element in the outcome matrix incorporates multiple parameters that originally correspond to multiple layers and being efficiently trained using the KD approach that we propose in this work.

Table 4.4: Accuracy of AlexNet SCLC with different teacher models. (Table from [JX1])

Teacher Model	SCLC Accuracy
ResNet18	78.50%
VGG16	79.80%
AlexNet	81.40%

As our experiments indicate, for effective approximation of the performance of the baseline AlexNet by KD, the SCLC needs to have as similar structure as possible to the original nonlinear network structure (i.e., same number of layers). Therefore, an approach such as KD is not directly applicable to train the much smaller number of elements in the outcome single matrix.

We also studied deeper and more complex teacher networks than AlexNet, such as VGG16 and ResNet18, to examine whether the KD could distill "additional knowledge" from these networks and improve the performance of the SCLC counterpart of AlexNet. Table 4.4 shows the results of training the SCLC with three different teacher networks: AlexNet, VGG16 and ResNet18 on the High-10 dataset. As can be observed, even when the SCLC student is trained with a better performing teacher, SCLC still has the best performance when the teacher network is the network from which it was originated (i.e. AlexNet). These results reaffirm the need for the student and the teacher networks to be as architecturally similar as possible to boost student accuracy.

While we have shown that SCLC counterpart reaches accuracy and potential speedup through OSCLC in forward inference, these results are for networks that are not too deep, such as AlexNet. When the number of layers increases to a much deeper network, such performance is not guaranteed to persist. The reason for this limitation is that it is unclear how to perform the normalization or skip connections operations in the optical setting. Notably, since optical setup would be mostly applicable to real-time processing situations,

operations such as batch-normalization may not be required since the batch size for these applications will be of size 1. However, as deep learning literature indicates, incorporation of normalization is critical for improving the overall performance of the network.

Chapter 5

KNOWLEDGE TRANSFER TO METASURFACE-BASED OPTICAL NEURAL NETWORKS

5.1 Motivation

Convolutional neural networks (CNNs) represent a significant milestone in image classification, recognition, and tracking [122]. CNNs, for example, AlexNet, are composed of several convolutional layers that adaptively learn spatial representations from input images. While powerful, the convolution operation is computationally expensive, leading to high latency and power consumption. In fact, it has been estimated that about 80% of the total runtime of CNNs is used in performing convolution operations [123]. Reducing this latency and as a result power consumption has become an active area of research, with multiple works proposing free-space optical systems as a solution [25, 124, 22, 19, 125, 126]. Beyond latency reduction and power consumption, optical information processing features qualities including high bandwidth, spatial parallelism, and low-loss transmission which have led to a surge of interest in the field [25].

For decades, it has been known that a $4f$ lens system can be used to perform convolutions optically by placing an appropriate filter at the Fourier plane of the lens [127, 128, 23, 124]. This was demonstrated in 2018 [124] using a diffractive optical element as the filtering element and traditional refractive lenses composing the $4f$ system. Spatial light modulators [19] and digital micromirror devices [129] can also be used as the filtering element. However, one drawback of the Fourier-based $4f$ approach is that it requires three elements (two lenses and a spatial filter), resulting in a bulky optical system with greater propensity for misalignments than single-element optical systems. Such misalignments from each optical convolutional layer cannot be ignored even when weights are trained with noisy inputs. In addition, the

filtering optics must be contained within a compact area at the focal plane in the $4f$ system, which limits parallel processing ability unless creative measures are taken such as utilizing naturally present diffraction orders [129] or lenslet arrays [22].

Advantageously, the convolution operation can also be performed using free-space optics and requires only a single element. The resultant image produced by any optics is the input convolved with the point spread function (PSF) of the optics [42, 43, 44]. Therefore, by engineering optics to produce a particular PSF, convolution can be performed optically simply via passing light through the optics. Further, by passing the input through several of these optics in parallel, multiple convolution operations can be performed simultaneously at the speed of light [22, 130]. This approach leverages the inherent parallelism of light enabling the passive processing of a vast amount of data without increasing computation time [25, 129, 22]. This unique optical capability circumvents scalability issues when handling high-resolution images in traditional electronic-based CNN systems.

However, a challenge to all optically-implemented CNN approaches is that nonlinear layers are interspersed with the linear layers. For example, AlexNet consists of five convolutional layers followed by three fully connected layers [122]. Specifically, each convolutional layer in the architecture utilizes the rectified linear unit (ReLu) as its non-linear activation function, followed by the local response normalization and Max Pooling layer, ensuring an effective mechanism for spatial hierarchy extraction. Therefore, nonlinearity is consistently applied across all layers, serving as a foundational element of the network's design to enhance its learning capability. Without the particular nonlinear layers that are effective in CNNs (e.g., ReLu), the classification accuracy of the CNN drops by about 20% [JX1]. The nonlinear layers cannot be implemented using simple lens-like optics; to implement them optically, some physical nonlinearity must be introduced, for instance by using an atomic vapor cell [19, 20] or image intensifier [21]. Hybrid approaches involving repeated transduction of the signal to perform linear operations in optics and nonlinear operations in electronics provide little benefit due to large latency and power consumption in signal transduction [22, 21, 23]. Implementing only one of many required convolution operations does not provide much

benefit in terms of speed and latency. Alternatively, there have been recent breakthroughs in using end-to-end designs for physical and hybrid networks designs which perform image classification or other tasks without explicitly using convolution [24, 25, 26, 27, 28]. Such an approach can effectively implement multiple linear layers in one optical frontend. While novel, these end-to-end neural networks are computationally expensive to train and are applicable only to the physical system for which they were specifically designed. In another approach, all-optical classifiers composed of several layers of diffractive optics have achieved reasonable classification accuracy using coherent illumination at terahertz [16, 17] and near-infrared [18] wavelengths. Further, a metasurface-based on-chip diffractive neural network has also been demonstrated [131]. However, these all-optical approaches are limited to implementing only linear operations.

In this chapter, we experimentally demonstrate a hybrid optical-electronic CNN consisting of a single optical convolution layer with an electronic single fully connected layer to achieve similar accuracy as AlexNet on hand-written digit classification tasks. To overcome the limitation from the absence of optical nonlinearity, we apply knowledge distillation to remove the nonlinear layers, and compress multiple layers into a single linear layer [JX1]. Knowledge distillation (more details in Methods) circumvents the need for nonlinearity without a significant reduction in the performance by transferring knowledge from a larger, pre-trained network (the "teacher" network) to a more compact network (the "student" network). Here, we use a modified AlexNet, denoted AlexNet-Mod, as the teacher network and a single convolutional layer coupled with a single fully connected layer as the student network. We use this architecture to demonstrate a hybrid meta-optical platform, wherein an optical frontend based on a single meta-optic performs the linear convolution operation, followed by an electronic backend which contains a linear calibration layer and a fully connected layer. In such a way, the most computationally expensive operation is performed optically to leverage the benefits of optical computing, namely high spatial bandwidth and low power consumption. The use of a single meta-optic layer drastically simplifies the experimental setup and provides a compact geometry.

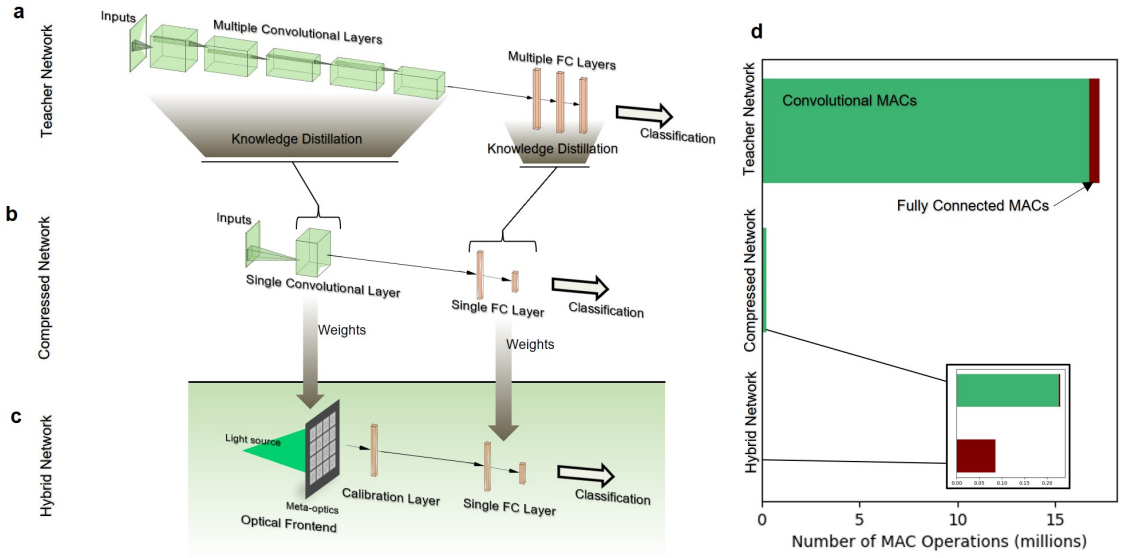


Figure 5.1: Schematic of convolutional neural networks for image classification tasks. (a) All-electronic multi-layered CNN. (b) All-electronic compressed CNN. (c) Hybrid CNN which combines an optical meta-optic front end and electronic backend. (d) The number of multiply-accumulate (MAC) operations of each network configuration, with convolutional MACs in green and fully-connected (FC) MACs in brown. (Figure from [JX3])

The optical frontend of this network is realized using inverse-designed meta-optics. The meta-optics are arrays of sub-wavelength scatterers which act as phase masks, imparting spatially-coded phase shifts to incident light. Here, we design meta-optics to realize a phase mask which performs the desired convolutional steps of the CNN by engineering the PSF. We fabricate and experimentally validate the performance of the designed optics using incoherent green light illumination from a light emitting diode, centered at 525 nm. Further, we experimentally demonstrate the classification accuracy of the entire hybrid CNN on the MNIST dataset. The hybrid CNN is described in Figure 5.1, where we compare the architectures of multi-layer electronic CNNs, compressed electronic CNNs (a linear single layer CNN), and our hybrid system. The number of multiply-accumulate (MAC) operations in

the entire network is reduced by over two orders of magnitude through compressing multiple convolutional layers into a single layer and implementing them optically. The classification accuracy of the compressed hybrid CNN is reduced by only 5% from AlexNet-Mod (98% accuracy) to achieve 93% classification accuracy on the MNIST dataset.

5.2 Methods

The hybrid CNN is described in Figure 5.1, where we compare the architectures of multi-layer electronic CNNs, compressed electronic CNNs (a linear single layer CNN), and our hybrid system. The number of multiply-accumulate (MAC) operations in the entire network is reduced by over two orders of magnitude through compressing multiple convolutional layers into a single layer and implementing them optically. The classification accuracy of the compressed hybrid CNN is reduced by only 5% from AlexNet-Mod (98% accuracy) to achieve 93% classification accuracy on the MNIST dataset.

5.2.1 Transfer Knowledge to Linear Networks

The knowledge distillation (KD) algorithm is designed to compress neural networks. KD accomplishes this by transferring knowledge from a larger, pre-trained network (referred to as the "teacher model") to a more compact network (referred to as the "student model"). In our implementation, we use AlexNet-Mod (a modified version of AlexNet) as the teacher network and a linear electronic network as the student network. The student network only comprises a single CNN coupled with a single fully connected layer. Straightforwardly training this linear network tends to easily converge to sub optimal saddle points; however, with the knowledge distillation approach, the student network converges faster and obtains better performance than it would achieve without the knowledge distillation training.

The KD algorithm optimizes the linear student network by combining two types of losses: temperature loss and student loss. Therefore, the total loss is then calculated as a weighted

summation of the two losses

$$\mathcal{L}(x, \Phi) = \alpha \mathcal{L}_C(y, p^{hl}) + (1 - \alpha)\mathcal{L}_k((p^{sl,t}; T = \tau), (p^{sl,s}; T = \tau)) \tag{5.1}$$

where x corresponds to the input, y is the training data, Φ are the student model weights, $\mathcal{L}_C$ is the cross-entropy loss function, $\mathcal{L}_k$ is the Kullback-Leibler (KL) Divergence Loss function [132], p^{hl} corresponds to the student model hard predictions, $p^{sl,s}$ corresponds to the student predictions under given teacher model probabilities $p^{sl,t}$, and α is the weighting parameter.

5.2.2 Calibrating Optical Experiment Results with Limited Data

The calibration function is designed to remap the optical convolution outputs to align with those of the previously trained backend. This addition addresses a variety of differences which may occur between the optical and electronic counterparts, including scaling, translation, rotation, and optical noise. With the addition of the calibration layer, the weights of the fully connected layer are preserved, and the optical frontend can be integrated into the existing network framework without retraining the backend or fine-tuning the optical alignment.

Specifically, the backend includes a calibration layer and the original backend layer used in the "compressed CNN", which are both fully connected layers. The input of the original backend layer is 288, and the output is 10, where 288 is the flattened image size after the optical convolutional layer. The input and output of the calibration layer are both 288 to align with the optical frontend and the original backend. The network's performance could be further enhanced by adding additional calibration layers, but incorporating more fully connected layers might risk gradient vanishing or explosion. The loss function used in this process is defined as

$$\mathcal{L} = min(f_{calibrate}(ON), EN), \tag{5.2}$$

where ON is the network with the optical experiment results and EN is the all-electronic network. This approach aims to refine the experiment output to align more closely with the pre-designed electronic network. To prevent overfitting, we strategically limit our training

to only 10% of the available data, ensuring that our model remains efficient [JX5, JX10]. The calibration layer addresses diverse types of noise encountered in the optical system, thus ensuring a more robust hybrid network.

To obtain the accuracy, the model generates scores for each input image These scores are output of the last FC layer (denoted as logits, size 10 by 1). We use the following equation to convert the logits to accuracy

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\left(\arg\max(\text{logits}_i) == \text{label}_i\right), \tag{5.3}$$

where N is the is the total number of samples. We convert these logits into predicted class indices by selecting the class with the highest score for each data point. Next, we check if the predicted class for each data point matches the true class. Finally, we sum the number of matches to determine how many predictions were correct.

5.2.3 Meta-optics Design

The meta-optics are designed with our experiment setup in mind. Due to the sensitivity of our camera (GT-1930C) and available light sources, we design the optics specifically for 525 nm illumination. For all electromagnetic simulations, we use a simulation grid size of 586 nm to be both comparable to the wavelength of the light and evenly divisible by the size of the camera pixels (5.86 μm per pixel). Each sub-optic is square, 800x800 simulation pixels in size, which provide compact footprint and reasonable computation time. While it is not necessary to propagate the electric fields on a sub-wavelength grid, it is necessary to design the meta-optic scatterers with sub-wavelength periodicity. Therefore, we divide each meta-optic pixel into a 2x2 block of square meta-optic scatterers each with a period of 293 nm. With 750 nm SiN (n = 2.06) pillars, we select a set which provides 0 to 2π phase shift at the desired wavelength. The scatterer unit cells were simulated using S4 RCWA [133]. More details on the meta-optic design are available in the Supplement.

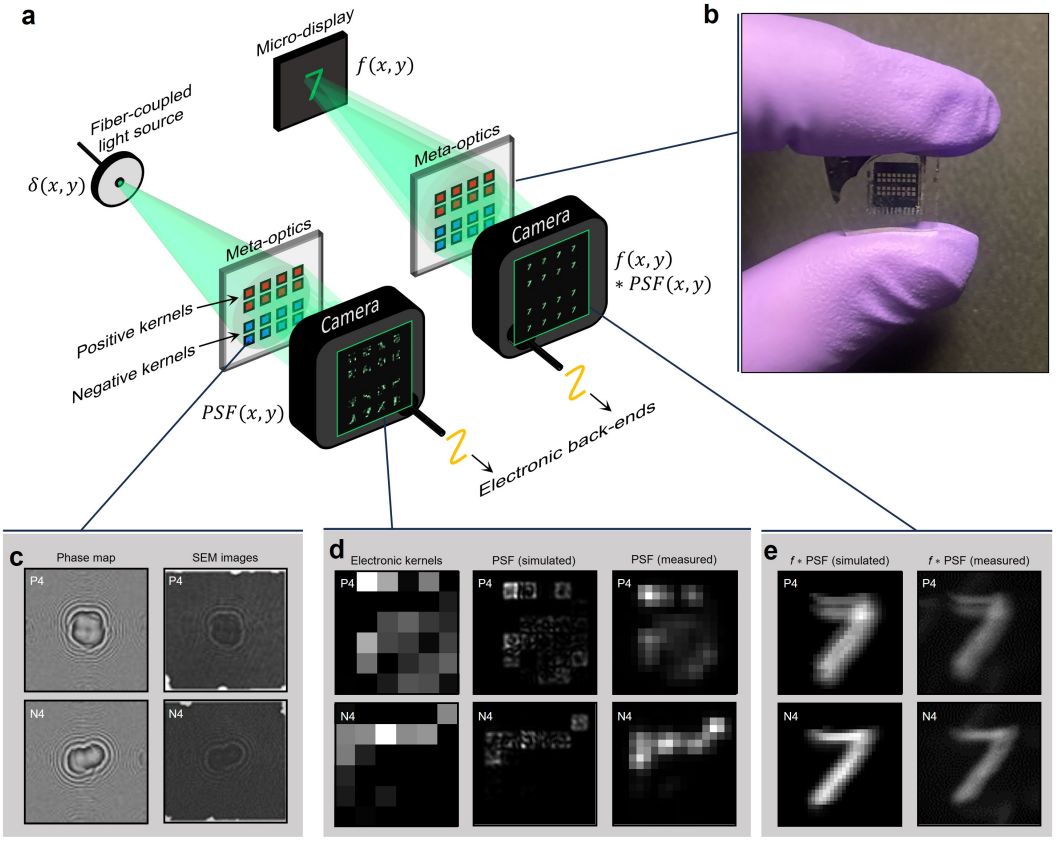


Figure 5.2: Schematic of the optical system. (a) PSF measurement setup using a monochromatic point light source (left) and optical convolution measurements using a micro-LED display (right). (b) A photograph of the fabricated meta-optics. The meta-optic contains 16 different sub-optics, spatially distributed in a single layer, operating in parallel for classification tasks. (c) Phase maps and SEM images of exemplary sub-optics corresponding to the positive and negative parts of a particular convolutional kernel. (d) The positive and negative parts of an example convolutional kernel (left) and the corresponding PSF simulation (middle) and right (experiment). (e) The simulated electronic output (left) and optical experiment (right) convolved output for the example kernel, for the case of an input "7" from MNIST. (Figure from [JX3])

5.2.4 Meta-optics Fabrication

The convolutional meta-optics are fabricated on a silicon nitride layer on a quartz substrate. We first deposited silicon nitride on a double-side polished quartz wafer using plasma-enhanced chemical vapor deposition (Oxford; Plasma Lab 100). Then we patterned on a positive-tone resist (ZEP-520A) using e-beam lithography (JEOL; JBX6300FS). We used alumina as a hard mask for etching the silicon nitride layer, so we deposited the alumina using e-beam evaporator (CHA; SEC-600) and did liftoff with 1-methyl-2-pyrrolidinone. We etched the silicon nitride layer with a plasma etcher (Oxford; PlasmaLab 100, ICP-180) using fluorine-based gases. In order to minimize the stray light of the meta-optics, we blocked the light except for the 16 kernel sub-optics by putting apertures around each one using photolithography (Heidelberg; DWL66+) and metal deposition followed by a liftoff process.

5.2.5 Optical Measurements

Two-dimensional point spread functions of 16 different optical kernels are measured simultaneously with a simple measurement setup. Single-mode optical fiber-coupled light source acts as a point source, and the meta-optics are placed 92 mm apart from the source. As we put the meta-optics on a three-axis linear stage and a kinetic mount with two rotation adjusting knobs, both the position and angle of the meta-optics can be well-defined with respect to the designed setup. Then we put a high-resolution color camera (GT-1930C with 5.86 μm per pixel resolution) 2.4 mm from the meta-optics to collect the point-spread functions of the kernels. We simply replaced the light source from the single-mode optical fiber to a micro-display presenting MNIST dataset of handwritten digits to get the convolved images. The camera could capture all 16 convolved images from different kernels at the same time, and we used Python code to automatically collect convolved images from 10,000 number of MNIST dataset.

5.3 Experiments and Results

In each convolutional layer of a CNN, an optimized kernel is convolved with the input to generate a feature map which is then passed to the next layer. Using the knowledge distillation approach, we optimize eight convolutional kernels (each 6×6 pixels in size) for the MNIST dataset of handwritten digits. The selected number and size of the kernels are based on previous experimental results [134]. As described in the next section, we design the optics to implement these optimized convolutional kernels and combine them with an electronic backend for image classification.

5.3.1 Compressing Multiple Convolutional Layers with Knowledge Distillation

To select the ideal network architecture for the linear optical-electronic hybrid system, underfitting and overfitting issues must be avoided. Smaller networks face underfitting concerns, particularly in optical settings that limit the system to just 8 kernels and remove nonlinear functions [135]. This is in stark contrast to AlexNet, which uses over 300 kernels [122]. However, complex models are prone to overfitting, and practical issues such as fabrication noise and misalignments introduce further challenges for optically implementing a large number of kernels. Our Schematic of the optical system is in Figure 5.2

Therefore, to obtain a balanced network that could be implemented optically, we use knowledge distillation to compress a AlexNet-Mod as a base model. AlexNet-Mod consists of 5 convolutional layers and 3 fully connected layers (8 total layers) which we compress to the desired structure of one convolutional layer and one fully connected layer (2 layers). The knowledge distillation approach assumes that the teacher network is already trained and performs the desired task with high accuracy; in this case, we use AlexNet-Mod as the teacher network, which achieves 98.9%$\pm$ 0.33% on training and 98.4%$\pm$0.32% on testing classification accuracy MNIST datasets over repeated trials. Additionally, AlexNet-Mod employs nonlinear activation functions (ReLU) to optimize performance; these are circumvented by knowledge distillation for a result that is compatible with our optical setting. In the compressed network,

we limit the number of kernels in the compressed convolutional layer to 8, and each kernel is 6×6 pixels in size. After training, the compressed electronic network achieves an approximate classification accuracy of 96% on both training and testing datasets.

To verify the performance of the optics, we measured the PSF of each sub-optic using a single-mode fiber as a point light source, as shown in the Fig. 5.2(a). The PSFs from all 16 sub-optics are simultaneously captured by the two-dimensional CMOS camera (more details in Materials and Methods). Accordingly, the convolved images from all 16 sub-optics are also captured at the same time when we replaced the single mode fiber with the display, as described in Fig. 5.2(a). Exemplary electronic convolutional kernels, which represent the ground truth PSFs for the optically-implemented kernels, are shown in Fig. 5.2(d). We also present the simulated PSFs from the meta-optics using angular spectrum propagation [136, 137]. The experimentally measured PSF shown in Fig. 5.2(d) match well to the simulated PSFs, confirming fabrication accuracy. However, due to the constraints on physically realizable PSFs, there are notable differences between the ground truth PSFs and experimentally measured PSFs. To correct for these differences, as well as slight noise and misalignments in the optical system, we introduce a calibration layer to the computational backend, further discussed in Section 5.2.2.

5.3.2 Hybrid Network Classification Results

To address optical noise and misalignment affecting image classification performance, an additional calibration layer is introduced to adjust optical representations for compatibility with the computational backend. Specifically, this calibration layer is a single fully connected neural network layer with an input dimension of 288 and an output dimension of 288. We fine-tune it with only 10% of the training dataset and ensures that the computational backend does not need to be retrained. Therefore, the electronic backend of the hybrid network consists of the calibration layer followed by the original compressed electronic backend.

We compare three CNN architectures (AlexNet-Mod, compressed electronic network, and hybrid optical-electronic network) in Table 5.1. The MAC for the convolutional layer

depends on the image size (H, W), kernel size (k), number of kernels (c_{out}), and input channels $(\#k)$, and the MACs are calculated as $c_{\text{in}}HWk^2\#k$. The MAC for fully connected layer depends on the input size (m) and output size (n), and the MACs are calculated as mn. The AlexNet-Mod achieves classification accuracy exceeding 98% on both training and testing datasets. The number of MAC operations of this network is 17 million with 8 bit precision. The compressed electronic CNN achieves greater than 96% accuracy; this 2% decline reflects the inherent challenges of compressing multiple layers into a single layer. Primarily due to the compression of the convolution layers, the number of MAC operations is reduced to 228,672. The hybrid network, which integrates the optical convolution layer with the calibration layer and single fully connected layer electronic backend, experimentally achieves classification accuracy of 93.9% (± 0.25%) and 93.4% (± 0.22%) on the training and testing datasets, respectively, and requires only 85,824 MAC operations, which is 0.5% and 37% of that required for AlexNet-Mod and the compressed electronic networks, respectively.

Table 5.1: Classification accuracy and computational complexity of different neural network architectures on the MNIST dataset. (Table from [JX3])

Network Architecture	Train (%)	Test (%)	MAC Operations
AlexNet-Mod	98.9 ± 0.33	98.4 ± 0.32	17,323,520
Compressed electronic CNN (without KD)	84.2 ± 0.47	82.1 ± 0.69	228,672
Compressed electronic CNN (KD)	97.2 ± 0.35	96.2 ± 0.29	228,672
Hybrid CNN (KD)	93.9 ± 0.25	93.4 ± 0.22	85,824

Figure 5.3 illustrates the confusion matrices for three neural network configurations. Each matrix visually represents the model's tested performance across different classes (in this case, digits labeled 0 through 9), with the true labels on the rows and the predicted labels on the columns. The multi-layer electronic CNN, AlexNet-Mod, displays high values along the diagonal, exceeding 98.1% accuracy on each class. The compressed electronic CNN, while having a slight decline in diagonal values, still demonstrates robust classification accuracy with a minimum of 94.3%. The hybrid network exhibits a more diverse range of values along the diagonal, with some classes exhibiting lower predictive accuracy compared to the compressed electronic network. We attribute this slight decline to noise in the optical experiment, which may be due to optics fabrication, camera sensor noise, and optical noise due to vibrations that cannot be fully compensated by the calibration layer. Despite these factors, the hybrid network still performs reasonably well with the network correctly predicting each class with a minimum of 87.6% accuracy. This indicates that the hybrid network maintains a reasonable level of accuracy against these noises and discrepancies.

5.4 Discussion

5.4.1 Ablation Study and Principal Component Analysis

To understand the contribution of each component of the hybrid optoelectronic CNN, we perform an ablation study and principal component analysis. In the ablation study, we evaluate the classification accuracy when using only the electronic backend structure for classification. We summarize the ablation study results in Table 5.2 and the numbers in brackets represent the accuracy gain compared to "Backend Only" and "Calibration + Backend", respectively. For a fair comparison, the backend layer here is the same structure as used in the hybrid network but re-optimized for the best performance in the absence of any convolutional frontend. Specifically, we compare the performance of a backend layer only (a single fully-connected layer), the calibration and backend layers together (two fully-connected layers), and the entire hybrid network. For a single fully-connected layer alone, an accuracy of 89% was attained,

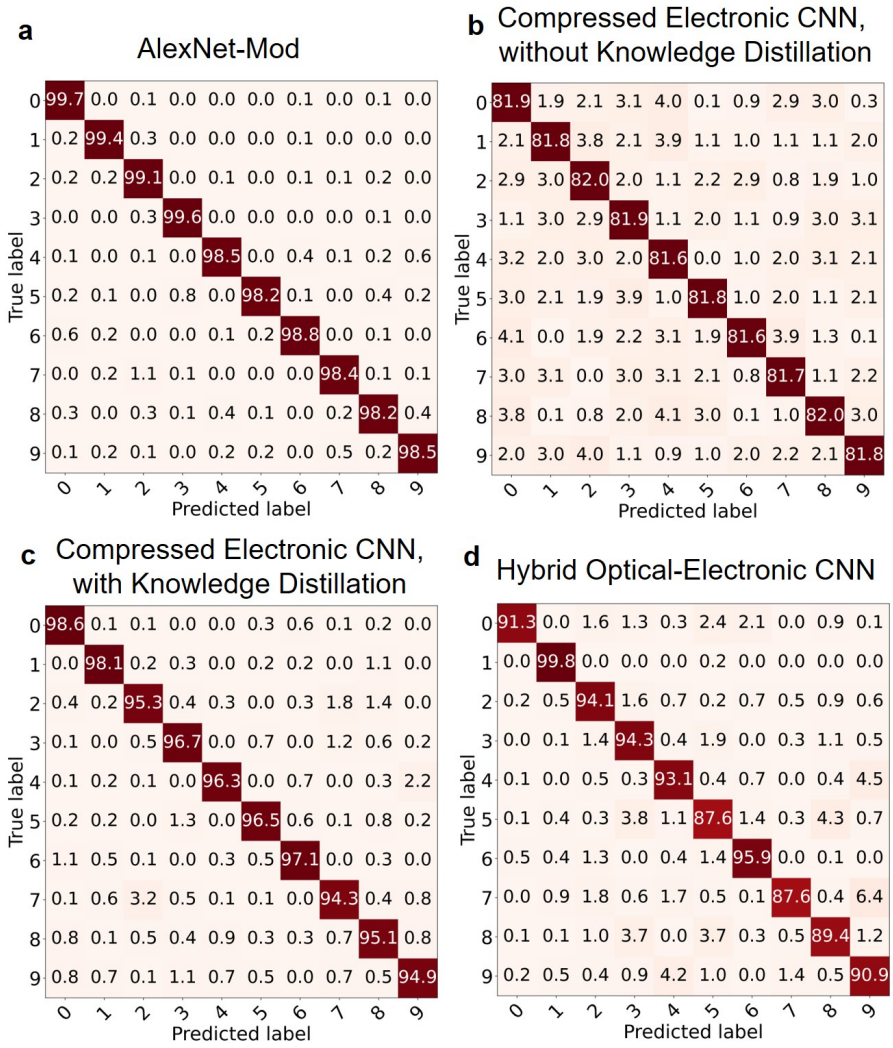


Figure 5.3: Confusion matrices for different network architectures. (a) Classification results for AlexNet-Mod (multiple-layer electronic CNN). (b) Classification results for the all-electronic CNN compressed without using knowledge distillation. (c) Classification results for the all-electronic CNN compressed with knowledge distillation. (d) Classification results for the hybrid optical-electronic CNN. (Figure from [JX3])

which is less than that of the hybrid network; this highlights the utility of the optical frontend. Further, for two fully-connected layers (representing the calibration and backend layer)

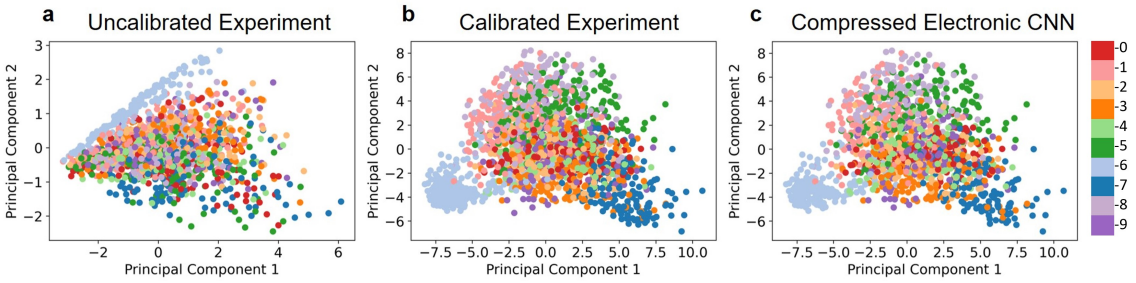


Figure 5.4: PCA of the hybrid CNN. (a) PCA of the uncalibrated experimental hybrid CNN classification data. (b) PCA of the calibrated experimental data, which has been re-mapped and exhibits clustering behavior similar to that of the compressed electronic CNN data. (c) PCA of the compressed electronic CNN data. (Figure from [JX3])

without any convolutional frontend, the accuracy is reduced to 84%. This can be attributed to the fact that two layers are fully-connected and, in the absence of nonlinear activation functions, tend to converge towards a "saddle point" in the optimization landscape [88]. We

Table 5.2: Ablation study of different network configurations, highlighting the contribution of calibration and optical components. (Table from [JX3])

Configuration	Train accuracy	Test accuracy
Backend Only	89%	87%
Calibration + Backend	84%	80%
Optics + Calibration + Backend	94% (+5% / +10%)	93% (+6% / +13%)

hypothesize that the calibration layer only re-maps the optical representations and does not improve the performance of the electronic backend alone. To examine this, we further analyze the effect of the calibration layer and the overall performance of the hybrid network as compared to an all-electronic network using Principal Component Analysis (PCA). PCA is a statistical method widely used in various fields, especially in analyzing the quality of neural

networks [138], to project original, high-dimensional data into a new, simpler coordinate system for explicit interpretation. Specifically, PCA computes the eigenvalue decomposition of the covariance matrix or the singular value decomposition of input data to determine the principal direction (known as "principal components"). Principal component 1 denotes the axis of maximum variance, encapsulating the most substantial relationships among variables, while principal component 2, orthogonal to principal component 1, captures the second most significant variance direction. Notably, the first two principal components typically contain the most crucial information. In this study, we compress the output dimensions from electronic and optical convolutional layers into two principal components, respectively, to compare the classification efficacy of each approach; this is observed by comparing the clusters observed in PCA visualizations [139].

In Fig. 5.4, we use PCA to show that the raw experimental data do identify the fundamental components necessary for classification, but that the calibration function is necessary to shift optical representations to a form that is compatible with the pre-designed electronic backend. As shown in Fig. 5.4a and 5.4c, we observe that both the all-electronic CNN outputs and the uncalibrated hybrid network experimental results can effectively distinguish between different classes due to the clustering behavior of specific classes, e.g. light blue (number 6) and navy blue (number 7). This clustering behavior indicates that despite any observable shifts in the PCA plot, the fundamental capacity of the hybrid network to classify data remains comparable to that of all-electronic networks. However, due to differences between the optical experiment output and the expected input to the electronic backend, directly using the original backend network results in a notable drop in accuracy, down to 16.3%. The calibration layer is designed to re-calibrate the outputs from the optical convolution layer back to the original outputs, thereby enabling the use of the original backend without retraining. As shown in Fig. 5.4b and 5.4c, the calibrated experiment result exhibits very similar clustering behavior to the all-electronic network, further demonstrating that the hybrid network classification is comparable to that of the all-electronic network.

5.4.2 PSF-Engineered Meta-Optics

We emphasize two advantages of our PSF-engineering method to perform optical convolution. Firstly, we highlight the simplicity of the optical system, as this method requires only a display, a single layer of optic, and a camera, making it compact and simple to execute. Incoherent illumination is used, so this approach can be applied to real-world image classification scenarios. Secondly, we highlight the ease of integration with the optimized electronic system. One of the challenges faced by optical computing is that electronic computing is already extremely powerful, having had decades of research and development into algorithms and hardware [25]. In our approach, the optics are designed to implement the electronically-optimized kernels. These kernels may be modified to reflect improvements in electronic CNN models and architectures, and the optics can accordingly be adapted to implement convolutions with arbitrary kernel matrices.

Furthermore, the Gerchberg-Saxon (GS) algorithm [140] used to design the optics is a well-established technique. The GS algorithm is an iterative phase retrieval algorithm to determine the phase (in the optic plane) that produces an intensity pattern in another desired plane (the focal plane). In other words, the meta-optics are phase-only holograms producing the desired PSFs as their images. Due to the fact that only amplitude, and not phase, contributes to the intensity pattern, the iteratively designed phase masks are not unique. More details on the implementation of the GS algorithm are available in the Supplement. In the Supplement, we also discuss an alternative design method based on automatic differentiation, which also produces viable optics but we found the GS-designed optics to produce slightly brighter, clearer images.

There are, however, two major limitations of the PSF-engineering approach. One limitation is that there is no guarantee that the desired PSF is physically realizable. That is, a single phase mask that satisfies the desired amplitude constraints may not exist. However, by introducing an electronic calibration layer, the resultant PSF does not need to be perfect in order to effectively classify the data. Alternatively, to ensure physically realizable PSFs, one

could adopt an end-to-end optimization scheme wherein the phase mask is simultaneously optimized with a backend. However, training this large phase mask (on the order of $10^5 - 10^6$ unit cells per kernel optic) is prohibitively costly and the design space is potentially too large to attain convergence. In contrast, separately training the electronic convolutional kernels and the optics are both reasonable steps, which as demonstrated are effective when combined.

A second limitation of the described approach is that we assume the PSF is spatially invariant. In reality, light from different spatial locations on the imaging object intersects the meta-optic at various angles of incidence, resulting in a different PSF for the off-axis rays. In contrast, we design the optics assuming a normal incident illumination. To mitigate this discrepancy, we ensure the incoming angles of incidence are relatively small by placing the display far away from the optics (90 mm) relative to the focal length (2.4 mm). Therefore, for a displayed image size of 8 mm $\times$ 8 mm, we ensure that the maximum deviation from normal incidence is 3.6°, and therefore the assumption of spatial invariance is reasonable for our system. However, for a large field of view imaging system, we may need to explicitly model the spatially varying PSF. We note that Wei et al. [130] use reparameterization techniques to design spatially varying kernel optics and report higher classification accuracy using this method (73.8%) versus designing optics with the assumption of spatial invariance (71.6 %) on the CIFAR-10 dataset. In another variation of a PSF-engineering technique, Zheng et al. [28] engineer the PSF of polarization-sensitive meta-optics to provide an array of focal spots which produce images of intensities relative to the kernel weights. However, we note that this approach requires a more complex optical system and incurs transmission losses under ambient light due to polarization sensitivity.

5.4.3 Outlook - Computational Effectiveness of the Hybrid Convolutional Neural Network

The number of MAC operations required of a network serves as a metric of computational complexity which is independent of the employed hardware technology. In a modern digital system, one MAC operation consumes approximately $1pJ$ [27, 21], so the hybrid network is expected to reduce the required power to classify an input from $17\mu J$ to $85nJ$ based on the

reduction in MAC operations. The latency of such a classification task is also expected to decrease proportionately. For input which is already in the optical domain, the power and latency required to capture an image and convert it to digital input is the same regardless of the network. Specifically, the hand-written digits of MNIST were captured using a standard camera, and the optical frontend of our network uses a standard camera sensor with meta-optics replacing the refractive camera lens. The network's performance could be further enhanced by adding additional calibration layers, but incorporating more fully connected layers might risk gradient vanishing or explosion [141]. We also find that a larger number of kernels could increase overall accuracy, but this would require increasing the number of sub-optics in the meta-optic layer and thereby the overall footprint of the meta-optical layer. This footprint is ultimately limited by the camera sensor size.

The benefit of optically implementing the convolutional step becomes more significant as the number of input pixels is increased. The electronic computational complexity of each convolutional layer is determined by the height and width of the input images (H, W), as well as the size of the kernels (k^2), and is $\mathcal{O}(HWk^2)$ [JX1]. However, the computational complexity decreases to $\mathcal{O}(1)$ in the optical convolutional layer. For example, when the MNIST dataset's typical image size of 28×28 pixels is increased to 100×100 pixels, the electronic computational complexity of each convolutional layer is expected to increase 12.76 times (assuming the kernel size remains unchanged), but an optically implemented convolution would not incur any increase in computation time. Therefore, as the resolution of real-world images continues to increase, hybrid networks such as the one described offer a promising solution to scaling problems incurred by all-electronic networks.

In summary, we demonstrate single-layer optical convolution with an electronic backend to achieve similar accuracy as AlexNet-Mod on MNIST hand-written digit classification, with 99.5% reduction in computational complexity. To circumvent the nonlinearity of AlexNet-Mod, we use knowledge distillation to compress the CNN into linear layers which are then implemented in a hybrid format. As a further innovation, we implement the convolution optically via engineering the PSF of meta-optics, which results in a more compact and

resilient optical frontend than the commonly used $4f$ lens system and does not require coherent illumination or polarization control. This hybrid approach integrates seamlessly with existing CNN architectures, utilizing simple optical design and requiring no re-training of the electronic backend to classify the data in experiment. This approach is also suitable for scaling to higher-resolution datasets; unlike in all-electronic networks where the convolution time scales with the number of input pixels, for optical convolution the processing time is independent of the resolution of the dataset. This chapter serves as a baseline for other optical and hybrid neural networks for higher bandwidth as well as lower power and latency in increasingly prevalent CNN applications.

Chapter 6

TRANSFERABLE POLYCHROMATIC OPTICAL ENCODER

6.1 Motivation

Visual information plays a crucial role in human response, particularly in situations where reaction time is limited to a few tens to hundreds of milliseconds [142]. Though the human brain has efficiency far exceeding that of any other human-made computing systems, it still cannot process the entire collected visual data due to its massive amount of information. Most likely, our brain performs early visual processing to extract essential features for efficient and rapid interpretation without handling the entire visual data [143, 144, 145].

With the dramatic development of artificial intelligence (AI), computers can process the visual information like human brain, thanks to artificial neural network (ANN), enabling computer/machine vision [146, 147, 4, 148, 149]. Despite impressive progress, real-time inference with limited computational resources remains very challenging even with more efficient algorithms. For example, in a flying object (i.e., habitat drones [150]) on-site data processing is plagued by severe heating, battery capacity and weight handling challenges. Utilizing cloud based systems poses challenges associated with data security and additional data transfer latency [151, 152].

Optical neural networks have emerged as a potential platform to circumvent these trade-offs, since an optical system can process multidimensional information with large spatio-temporal bandwidth [7]. Recently, integrated photonics and free-space or fiber optics have been employed to implement some parts of an ANN for image compression/encryption [6, 153] and classification [14, 154, 15, 155, 156]. However, most of them are highly restricted on solving a relatively simple gray-scale datasets (i.e., MNIST and fashion-MNIST) and only a couple of systems have shown their implementation for more complicated multichannel datasets

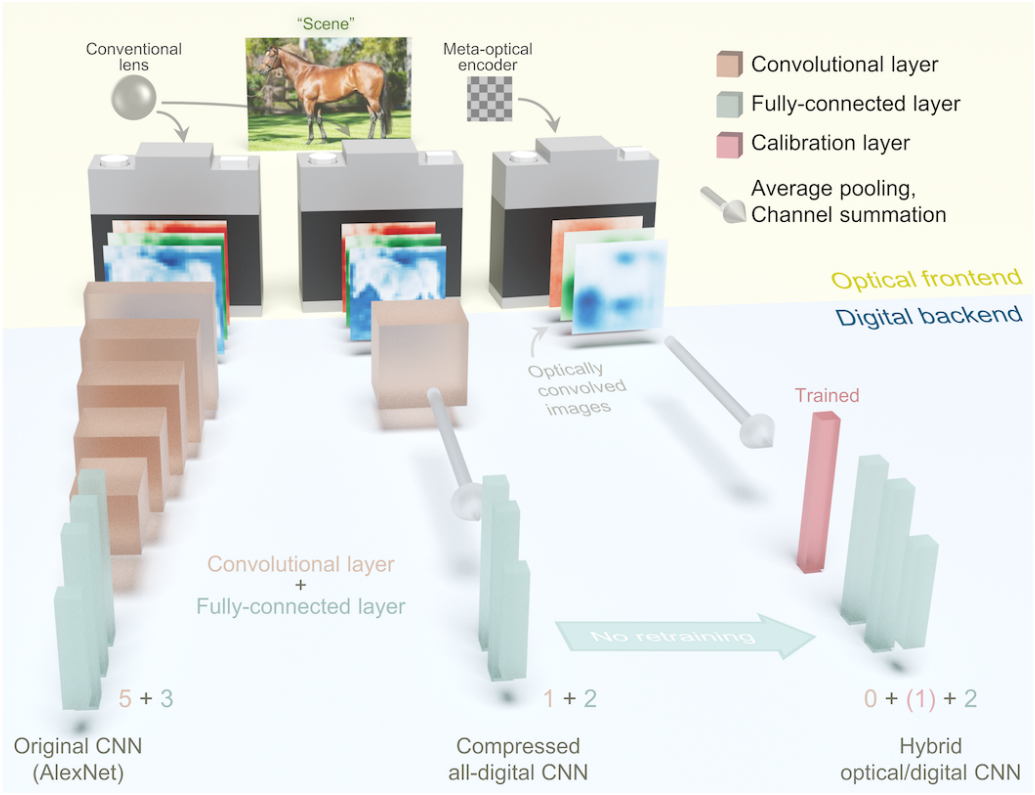


Figure 6.1: Schematic process flows of different image classification methods using original CNN, compressed all-electronic CNN, and hybrid optical/digital CNN. (Figure from [JX4])

(i.e., CIFAR-10 and ImageNet) [14, 15]. For these complex datasets, the optical system often become extremely large (with multiple stacks of the photonic circuits) [15], otherwise the classification accuracy remains low ($\sim 60\%$ accuracy for CIFAR-10 classification tasks) [14, 58, 59]. In addition, the most successful ANN architectures utilize nonlinear activation functions that are challenging to implement optically. Proposed solutions, including atomic vapor cells [19, 20] and image intensifiers [47], introduce significant experimental complexity, and additional power consumption.

To leverage the strengths of both optical and digital computing systems, an encoder-decoder inspired hybrid optical/digital architecture is a promising approach [4, 5, 149, 157]. Specifically, an analog linear optical frontend (denoted as the optical encoder) performs bulk

of linear computational tasks, while the digital backend implements the nonlinear operations. One intriguing possibility is to employ a static optical frontend, which is data agnostic, whereas the backend is trained and reconfigured. This resolves usual issues of modulation speed, errors, and system size in all-optical systems. An optical encoder is particularly suitable for convolutional neural network (CNN) architectures, where convolutional layers act as feature extractors, encoding high-dimensional images into low-dimensional features [JX1]. In fact, every free-space optic inherently performs a two-dimensional convolution operation during the imaging under incoherent light. The captured image is a convolution of the scene and the optic's incoherent point-spread-function (PSF) [105]. Thus, by engineering the PSF, an optical encoder can perform the desired convolution and replace the initial layers of a CNN.

Recently, PSF-engineered optical encoder has been employed to classify MNIST handwritten dataset and a reasonable classification accuracy with much less computational costs compared to the AlexNet is demonstrated [1]. We note that, however, MNIST images are monochrome, and is almost linearly separable (0.84% loss without any nonlinearity [158]). The monochrome nature of the images makes the PSF-engineering approach wavelength agnostic. On the other hand, datasets such as CIFAR-10 [159] or ImageNet subset (High-10) [160][JX1] are not separable by linear layers. Moreover, they consist of colored images, where the actual color information is exploited in classification.

Here, we demonstrate a polychromatic optical encoder with PSF-engineered meta-optics to classify the CIFAR-10 dataset. We first compressed the architecture into a single convolutional layer and two fully-connected layers using Knowledge Distillation. Then, we physically realized the convolution layer using an array of metasurfaces, where each metasurface, thanks to the inherent chromaticity, performs a separate convolution for each color channel. As a result, the hybrid CNN with an optical encoder reduces the total number of multiply–accumulate (MAC) operations at the digital backend by an factor of $\sim 24,000$. The reduction of number of MAC operation directly corresponds to the computational costs, i.e., power and latency [161]. It is worth noting that we always require an imaging system (i.e., lens and camera) to capture the

image under ambient illumination, before we deliver the image data to the computational backend. Hence, with a single meta-optical encoder, we are not adding any additional optics, but simply replacing a conventional lens with PSF-engineered meta-optics. This makes our optical system compact and fully compatible with conventional optical imaging systems, while the other systems such as, integrated-photonic systems require pre-processing of the data[15] and in-sensor computing needs a customized sensor design [162].

Furthermore, we adopt the same meta-optics (optical convolutional layer) which was optimized for CIFAR-10 dataset to High-10 dataset to explore the generality of optical encoders. In practice, a static optical encoder should be applicable for any scene. While, one approach is to employ reconfigurable frontend, e.g. based on non-volatile phase change materials [163] or liquid crystals [164], the performance of these reconfigurable front end in terms of individual pixel control, power consumption, and operating speed are still inferior for practical deployment. Remarkably, with the same passive optical encoder (optimized for CIFAR-10 dataset), we achieved a high classification accuracy (for High-10 dataset) by fine-tuning the digital backend with additional fully-connected layer (via transfer learning approach). This ability to generalize the frontend is crucial for any ANNs as it enhances their versatility, efficiency, and robustness. A network that generalizes well can be applied to different tasks without extensive re-training, saving time, reducing costs for meta-surface fabrications, and conserving computational resources for real-world applications.

6.2 Methods

6.2.1 Optical Encoder Pipeline

Our optical encoder concept is described in Figure 6.1. The original CNN, i.e., AlexNet, has five convolutional layers and three max pooling layers at the front, followed by three fully-connected layers at the end. Replacing all individual five convolutional layers with five sequential optics is extremely difficult because of misalignment, large system size, lack of nonlinearity, and low signal-to-noise ratio, issues that compound with increasing number

of optical elements. Therefore, we compressed the AlexNet to a single convolutional layer and two fully-connected layers using knowledge distillation method [63], which reduces the complexity of the architecture with a minimal compromise in accuracy.

6.2.2 Knowledge Distillation and Computational Back-end

As previously discussed, the Knowledge distillation loss is calculated as a weighted summation of the two losses

$$\mathcal{L}(x, \Phi) = \alpha \mathcal{L}_C(y, p^{hl}) + (1 - \alpha)\mathcal{L}_k((p^{sl,t}; T = \tau), (p^{sl,s}; T = \tau)) \tag{6.1}$$

where x corresponds to the input, y is the training data, Φ are the student model weights, $\mathcal{L}_C$ is the cross-entropy loss function, $\mathcal{L}_k$ is the Kullback-Leibler (KL) Divergence Loss function [132], p^{hl} corresponds to the student model hard predictions, $p^{sl,s}$ corresponds to the student predictions under given teacher model probabilities $p^{sl,t}$, and α is the weighting parameter.

The optical fabrication and alignment noise are unavoidable in meta-surface kernels. These include scaling, translation, rotation, image aberration, and optical noises. To address this issue, we propose adding a calibration function to remap the optical convolution outputs to align with those of the previously trained back-end. Specifically, we use a fully connected layer as the calibration function and corresponding loss function is defined as

$$\mathcal{L} = \min(f_{\text{calibrate}}(\textit{Hybrid optical/digital CNN}, \textit{Compressed CNN}). \tag{6.2}$$

This approach aims to refine the experimental outputs to align more closely with the pre-designed electronic network. To prevent overfitting, we strategically limit our training to only 20% of the available data, ensuring that our model remains efficient [JX5, JX10].

6.2.3 Meta-optics design

For 16 digital kernels for each R, G, and B channels, we have 32 meta-optical kernels as we use a single meta-optics for all RGB channels but we cannot represent both positive

and negative weights with optics. Hence, we create 16 positive kernels and 16 negative kernels, then perform digital substraction on the digital backend. Each of our convolutional meta-optics has 3200×3200 scatterers, with 2×2 scatters constitute a group to enhance the robustness of fabrication. Based on the ground-truth digital convolutional kernels, we defined optical PSFs for each RGB channels, and inverse-design the meta-optics having those PSFs at each RGB wavelengths using TensorFlow Adam optimizers.

6.2.4 Meta-optics fabrication

Our meta-optics operate at visible wavelength ($\lambda \sim 400\ nm - 700\ nm$). We use silicon nitride on quartz substrate for the meta-optics to have high transparency at the whole visible regime. We deposit a thick silicon nitride layer ($800\ nm$) on top of the double-polished quartz substrate using plasma-enhanced chemical vapor deposition (Oxford; Plasma Lab 100). We spin coat and bake electron beam resist (ZEP-520A) on top of the silicon nitride layer, followed by a spin coat anti-charging agent (DisCharge H20). We pattern using electron beam lithography (JEOL; JBX6300FS), and develop the resist using amyl acetate. After that, we deposit via electron beam evaporation (CHA; SEC-600) and do lift-off an alumina layer ($\sim 65\ nm$) for hard mask . Finally, we etch the silicon nitride with an alumina hard mask using plasma etcher with fluorine-based gas (Oxford; PlasmaLab 100, ICP-180). The sub-wavelength structured meta-optics has a period of 293 nm, which is a half of the camera pixel size collecting the image.

6.2.5 Optical measurements

We measure the PSF by placing a laser and a pinhole ($\phi = 25\ \mu m$), representing a point source. Then we place the convolutional meta-optics on 3-axis stage with rotational knobs to align the meta-optics centered and parallel to the beam path. High resolution color camera (GT-1930C) which has a pixel size of $5.86\ \mu m$ is placed $2.4mm$ away from the meta-optics. We measure the PSFs for each RGB color light by replacing the laser with three different wavelengths (Thorlabs; CPS450, CPS532, and CPS635). For image convolution measurements

of the CIFAR-10 dataset, we put the micro-display at the pinhole position, then connect to the computer to show the color images. Since a single meta-optics can represent three different RGB kernels at the same time, a color camera which have RGB color pixels can extract the convolved images at three different channels. This can eventually save the space of the meta-optics and camera, which is critical in real-world applications. The point source is replaced by an arbitrary two-dimensional image, $f(x, y)$. We can express the image as a sum of the three color channels, $f_R(x, y) + f_G(x, y) + f_B(x, y)$. The convolutional meta-optics perform a convolution for each color, and as a result, a convolved image, $\sum_{i=R,G,B} f_i(x, y) * PSF_i(x, y)$, will be imaged on the camera. Since we determined the enlargement factor of 2 for the PSF, we use the same enlargement factor for the CIFAR-10 image as well. According to the camera pixel size, 5.86 μm, and and CIFAR-10 image size, 32×32, the projected image size on the camera has to be about 374 $\mu m \times 374$ μm. At the given values of distance between the display and meta-optics and meta-optics to the camera, we can end up with the CIFAR-10 image size on the display to be 16.0 $mm \times 16.0$ mm. We use 10,000 images for training (subset of original $50,000$ images)and 10,000 images for testing, with an exposure time of 500 ms. Among the 10,000 images of training and testing dataset, 186 and 201 images are not involved on training and testing, respectively, due to the overexposure issue. All the measurement parameters and number of images are the same for the High-10 dataset for transfer learning process.

Another critical factor is the exposure time. Since the optical features are captured by a CCD camera, the exposure time significantly influences the final performance. If the optical features are overexposed, texture information, such as the fur of a cat, might be missing. Conversely, if the optical features are underexposed, most information may also be lost, resulting in a lack of distinction between highlights and shadows in the image. To find the most appropriate exposure, we could use a similar approach to modern cameras, where "18% gray" is considered as the mid-point between black and white on a logarithmic or exponential curve. This standard can help us achieve balanced exposure, ensuring that the captured optical features are neither overexposed nor underexposed.

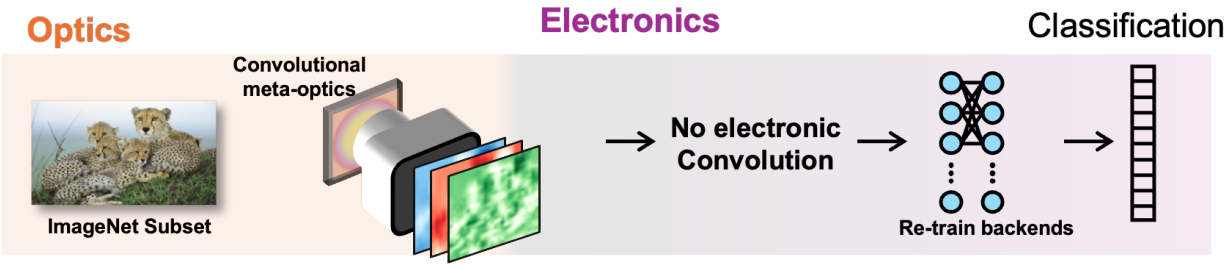


Figure 6.2: Schematics of the transfer learning process. (Figure from [JX4])

6.2.6 Computational backend

As previously discussed, optical fabrication and alignment noise are unavoidable in meta-surface kernels. These include scaling, translation, rotation, image aberration, and optical noises. To address this issue, we propose adding a calibration function to remap the optical convolution outputs to align with those of the previously trained backend. Specifically, we use a fully connected layer as the calibration function and corresponding loss function is defined as:

$$\mathcal{L} = \min(f_{\text{calibrate}}(\textit{Hybrid optical/digital CNN}, \textit{Compressed CNN})) \tag{6.3}$$

This approach aims to refine the experimental outputs to align more closely with the pre-designed network. To prevent overfitting, we strategically limit our training to only 20% of the available data, ensuring that our model remains efficient.

6.2.7 Transfer Learning

Generalization performance is a key feature to test our hybrid optical/digital CNN. Ensuring that the network can generalize well to new, unseen data is crucial for several reasons. First, our hybrid network is compressed from AlexNet, which was originally designed with a large dataset. The pre-trained AlexNet achieves high accuracy across various datasets and can be easily adapted or fine-tuned to out-of-distribution datasets. This adaptability is essential for

practical applications where the data distribution may differ from the training set. Second, exploring the generalization capabilities of hybrid models is important because designing and fabricating different meta-surface kernels for different tasks is inefficient. By enhancing generalization, we can use a single hybrid model for multiple tasks, reducing the need for extensive redesigns and fabrications. Figure 6.2, we deploy the optical frontend as the pre-trained model and adapt it to a new dataset, ImageNet sub-dataset, High10.

To implement the transfer learning, we add two types of losses: feature loss and label loss. The feature loss minimizes the discrepancy between the optical features and the electronic features, ensuring that the representations learned by the optical and electronic components are aligned. The label loss minimizes the discrepancy between the model's predictions and the actual labels, improving the overall prediction accuracy. During the transfer learning process, the optical front-end and electronic back-end remain unchanged. we add two fully connected layers between the optical front end and back-ends and fine-tune these layers using the two losses. Specifically, the function is:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{feature}} + \beta \mathcal{L}_{\text{label}}, \tag{6.4}$$

where $\mathcal{L}_{\text{feature}}$ is the feature loss and $\mathcal{L}_{\text{label}}$ is the label loss, with α, β as the respective weights balancing these losses.

6.3 Experiments and Results

A photograph of the fabricated chip is shown in Figure 6.3a. A single chip contains a total of 32 convolutional meta-optics (corresponding to 16 positive and 16 negative convolutional kernels) and additional 5 metalenses which are focusing light at the focal plane, ensuring the alignment (e.g., tilt, rotation, and distance) between the meta-optics and the camera. Figure 6.3b shows the schematic of the PSF measurement setup. By changing the laser diodes, we illuminate individual RGB coherent light onto the camera through the meta-optics and experimentally characterize the polychromatic PSFs. A pinhole of $25\mu m$ diameter creates

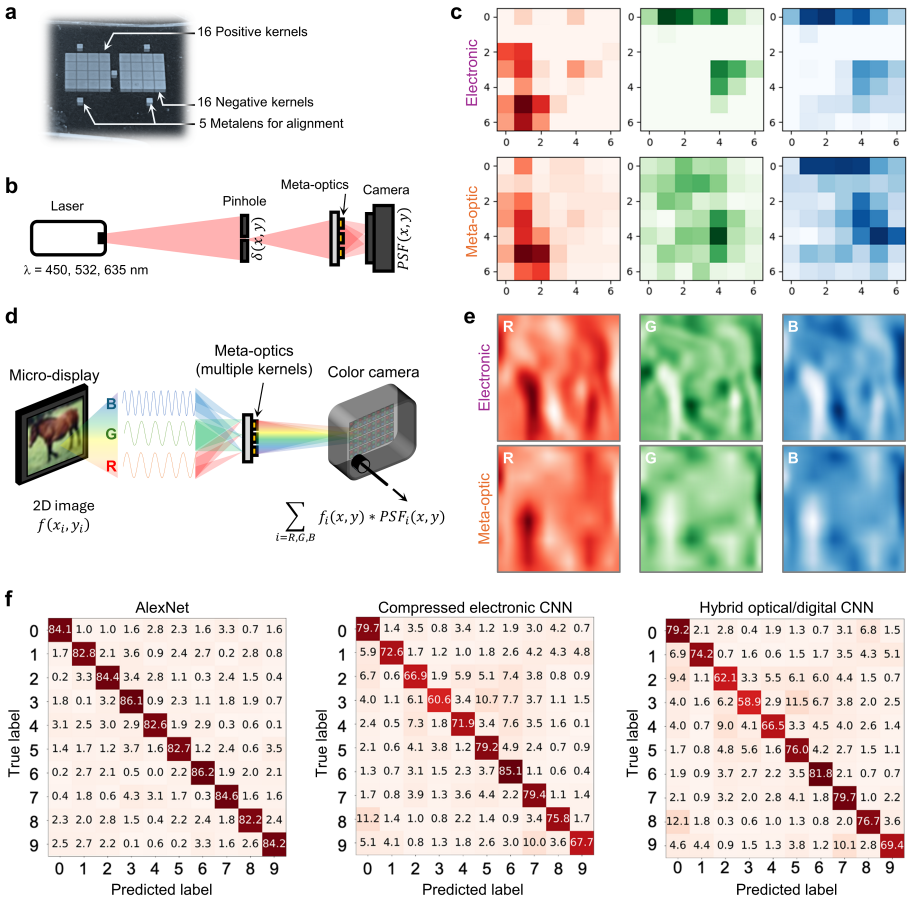


Figure 6.3: Optical characterizations of the meta-optic encoder. (a) Photograph of the fabricated optical encoder, consisting of 16 positive and 16 negative convolutional kernels and 5 algnment metalenses. (b) Schematics of the polychromatic PSFs measurement setup. (c) Ground-truth and measured RGB PSFs for a particular polychromatic kernel (positive kernel number 7). (d) Schematics of the meta-optical convolved image measurement setup with a micro-display. A color camera capture convolved convolved images with a single shot. (e) Electrically (above) and optically (below) convolved images of a particular CIFAR-10 image in individual RGB colors. (f) Confusion matrices of CIFAR-10 dataset classification tasks with different network architectures. (Figure from [JX4])

an approximate point source and the position of the optics, i.e., pinhole, meta-optics, and camera, remains the same while changing the laser diodes.

Figure 6.3c shows both the ground-truth PSFs and measured PSFs for a particular kernel (positive kernel number 7) for individual RGB wavelengths, which are not exactly the same. To quantitatively analyze the difference between two PSFs, we define a cosine similarity (η) as

$$\eta = \Sigma_i(A_i B_i)/\sqrt{\Sigma_i(A_i^2)}\sqrt{\Sigma_i(B_i^2)}, \tag{6.5}$$

where A_i and B_i are the ground-truth and measured intensity profiles of the PSF for RGB wavelengths, respectively. The calculated η for RGB wavelengths are about 0.88, 0.56, and 0.81, respectively. Imperfections in the fabrications and measurements can be one reason. At the same time, there is a fundamental limit originated from the polychromatic nature, in which not all the polychromatic PSFs are physically reliable as the phases at different wavelengths are not completely independent (but actually dependent) to each other with the $\phi - w$ relationships. Creating more physically-reliable PSFs by taking account both the optical front-end and computational back-end altogether, instead of optimizing only in the computational side and replacing the convolutional layer with optics, may increase the η. However, most importantly, we achieve fine classification accuracy (compared to the other)[14] with minimal additional losses from the imperfect meta-optical convolutional layer, which represents that the following calibration layer and two fully-connected layers of computational back-end can compromise the minor errors coming from the optical front-end.

Then, we test the polychromatic optical encoder for CIFAR-10 dataset. By replacing the pinhole with a micro-display, we can convolve the CIFAR-10 images with the characterized PSFs of the meta-optics (Figure 6.3d). The displayed image size is carefully adjusted according to the convolutional kernel size and the enlargement factor on the camera. Figure 6.3e shows the electronically and meta-optically convolved RGB images of one of the CIFAR-10 dataset. The meta-optically convolved image loses some of the high resolution components, probably due to the imperfect fabrication and alignment errors which are already recognisable from the PSF measurements. Because the input intensity of the light at each meta-optics varies

depending on their spatial position and we separately normalized the intensity of the PSF for each color, and we introduce an additional layer, so called calibration layer, which accounts for these factors and adds only minimal computational cost before going through the fully-connected layers.

Table 6.1: Classification performance of different network architectures on the CIFAR-10 dataset. (Table from [JX4])

Network Architecture	Train accuracy (%)	Test accuracy (%)
AlexNet	83.04 ± 0.87	81.03 ± 0.89
Compressed electronic CNN	76.94 ± 0.52	76.59 ± 0.50
Compressed optical/digital CNN (without calibration)	56.78 ± 0.91	56.39 ± 0.92
Compressed optical/digital CNN (with calibration)	73.18 ± 0.58	72.06 ± 0.57

Figure 6.3f shows the confusion matrices of the classification accuracy of the CIFAR-10 data with original CNN (AlexNet), compressed CNN using knowledge distillation, and hybrid optical/digital CNN using convolutional meta-optics after the compression. Even though there are slight differences between the optical and electronic convolution results (Figure 6.3e), after tuning the calibration layer, we can still get similar (less than 5%) accuracy for both training and testing dataset (Table 6.1). Additionally, this hybrid approach significantly reduces computational costs which can be represented by the number of multiply-accumulate (MAC) operations. From the original CNN to compressed CNN we can reduce the computational load, which is represented by a number of MAC operations, by a factor of $\sim 1,400$, while we can reduce further by a factor of ~ 17 after replacing a convolutional layer with meta-optics.

To analyze the effectiveness of the meta-optical convolutional layer, we utilize principal component analysis (Figure 6.4). For the original CNN and compressed all-digital CNN, each class is well-separated (Figures 6.4 a and b), implying that we can extract out the key features of the CIFAR-10 image dataset after convolution. On the other hand, after the optical convolution using meta-optics, without calibration, different classes of the image were

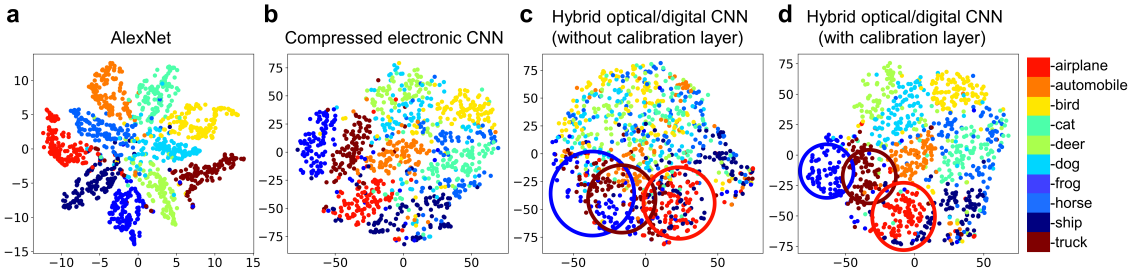


Figure 6.4: Principal component analysis for CIFAR-10 image dataset. (a) Original CNN, AlexNet. (b) Compressed all-electronic CNN. (c) Hybrid optical/digital CNN without calibration layer. (d) Hybrid optical/digital CNN with additional calibration layer. (Figure from [JX4])

very difficult to distinguish (Figure 6.4c). Additionally, some clusters exhibit larger sizes and overlapping regions than Figure 6.4(a-b), e.g., the navy blue, brown, and red clusters (confidence ellipses). However, after introducing the calibration layer, the clustering regions become smaller, and the separations between classes increases. As shown in Figure 6.4d, each class becomes well-separated and distinguishable, similar to the compressed CNN. This critical role of the calibration layer is consistent with the classification accuracy without and with the calibration layer (Table 6.1). We note that the calibration layer can potentially be compressed into the pretrained digital backend via additional training and will not affect the number of MAC operation for inference.

Our convolutional meta-optics implements convolutional kernels which came from compressed CNN for CIFAR-10 data. Unlike computational neural networks, optical implementations are extremely difficult to modify once they are fabricated. This necessitates different convolutional meta-optics for different datasets. However, we found that the convolutional layer that we optimized for CIFAR-10 can be readily adapted to classify another dataset High-10, with a transfer learning process. We added an additional fully-connected layer, which we call a "transfer learning layer", that is located in between the former fully-connected layers

and convolutional layer. By training the transfer learning layer, we can fit the other dataset, i.e., High-10, to the CNN which is pre-optimized for a particular dataset, i.e., CIFAR-10, with only fine-tuning a small part of the original network without changing the former network structure (see details in Methods). The High-10 image dataset is polychromatic (RGB) and has a size of 224×224 size. To use the former CNN optimized for CIFAR-10 data for High-10 data, we resize the High-10 images to 32×32 size, same as the CIFAR-10 data.

Table 6.2: Transfer learning results on High10 (Table from [JX4])

Network Architecture	Train accuracy (%)	Test accuracy (%)
AlexNet	85.31 ± 0.27	84.95 ± 049
Compressed CNN (without Transfer Learning)	41.43 ± 0.26	40.44 ± 0.39
Compressed CNN (with Transfer Learning)	67.43 ± 0.22	66.01 ± 0.13
Hybrid optical/digital CNN (with Transfer Learning)	63.46 ± 0.46	59.73 ± 0.91

Without applying a transfer learning method, the training and testing accuracy is rather low around 40%. However, after transfer learning, we achieve much higher training and testing accuracy ($\sim 67.43\%$ and $\sim 66.01\%$, respectively) on the High-10 data with the convolutional layer and two fully-connected layers. We further experimentally verified this approach works in our hybrid optical/digital CNN using the same convolutional meta-optics that we used for the CIFAR-10 data digital backend structure with one additional fully-connected layer. The average training and testing experiment accuracy of the High-10 data are similar (less than 5% loss) to the compressed all-digital CNN, which is about the same of the CIFAR-10 case.

6.4 Discussion

6.4.1 Multichannel dataset

The advantages of the knowledge distillation and meta-optical encoder are a dramatic reduction of computational complexity, which is represented by the MAC operation. For the

CIFAR-10 dataset, our hybrid optical/digital CNN reduced the number of MAC operation by a factor of $\sim 24,000$. This reduction is about an order of magnitude higher than that of the MNIST hand-written dataset, where the meta-optical encoder reduced the number of MAC operation only by a factor of $\sim 5,400$ [1].

On the other hand, the classification accuracy drops are more significant for the CIFAR-10 dataset compared to the MNIST dataset. The train (test) accuracy for CIFAR-10 dataset of our hybrid CNN drops by $\sim 9.86\%$ ($\sim 8.97\%$) from the original CNN. For MNIST dataset, the train (test) accuracy of our hybrid CNN drops by $\sim 5.0\%$ ($\sim 5.0\%$) from the original CNN [1]. While this classification accuracy drop in CIFAR-10 dataset is not negligible, our PSF-engineered optical encoder has significantly large classification accuracy compared to the other free-space optical neural network architectures (which are compatible with conventional camera systems). Our encoder has a classification test (train) accuracy of $\sim 73.2\%$ ($\sim 72.1\%$) for CIFAR-10 dataset without retraining the backend and only projecting by a calibration layer. These test (train) accuracy can be improved further up to $\sim 75.1\%$ ($\sim 73.2\%$) if we retrain the backend, which is better than the previous state-of-the-art result ($\sim 72.8\%$) which used a complex end-to-end optimization as well as the backend retraining with 50 number of kernels. Our hybrid optical/digital CNN can be further improved by using complex meta-atoms to reproduce better PSFs optically and using advanced compression method to reduce the loss during the knowledge distillation. The other reports have much less accuracy $\sim 63\%$ compared to ours.

For a ImageNet subset, High-10, we have the same amount of reduction in number of MAC operation as the CIFAR-10 dataset since we used the identical CNN architecture. The train (test) accuracy of our hybrid CNN heavily drops by $\sim 21.85\%$ ($\sim 25.22\%$) compared the original CNN. The majority of losses occur during the network compression as we share the convolutional layer and fully-connected layers optimized for CIFAR-10 dataset. However, albeit to the losses, our transfer learning results has a classification accuracy of $\sim 61\%$, still better than the other free-space optical neural networks system [14]. Here, we selected the ImageNet dataset which has more complicated and distinct classes from the CIFAR-10

dataset. We did not change the optical frontend, but only fine-tuned the digital backend of two fully-connected layers and an additional transfer learning layer, to show the versatility of our hybrid CNN system.

6.4.2 Energy consumption

In practice, we can implement our hybrid optical/digital CNN simply by replacing a lens with meta-optics during imaging. Hence, the energy consumption will solely be determined by the number of MAC operations. However, it is important to also consider the power from the sensor. Specifically the sensor power depends on the number of pixels being passed to the digital backend. For an original CNN, we only need 32×32 pixels to capture the image. On the other hand, hybrid CNN needs 6×6 pixels for imaging one convolved image, considering the average pooling, which ends up with $32 \times 6 \times 6$ pixels for all positive and negative kernels. Hence, our hybrid CNN requires a bit larger number of pixels for imaging compared to the original CNN.

The color camera we used (Allied Vision Prosilica; GT 1930 C) has a total power consumption of $3.4W$ with 50.70 frames per second and $1,936 \times 1,216$ pixels, which ends up with $28nJ$ per frame and pixel. Thus we estimate that the original CNN and hybrid CNN requires an energy of about $29.1\mu J$ and $32.8\mu J$, respectively, for the image capturing process per a single image. However, the energy consumption for the computational backend is much larger for the original CNN compared to the hybrid CNN. For state-of-the-art computational system, an energy consumption per a single MAC operation is $\sim 1pJ$. Thus the energy consumption for a single object classification task for the hybrid CNN is about $150nJ$, which is more than four orders of magnitude smaller than that of the original CNN, $3.65mJ$. Considering the sensor power, the total system level energy consumption for a single object classification task dropped from $3.68mJ$ to $0.03mJ$. We note that, we can trade-off the sensor power (by reducing the number of kernels) with computational backend power (by increasing MAC operations). However, having more operation in the optical encoder with a simple computational backend will always be preferred to reduce the latency.

6.4.3 Applications

The hybrid CNN has a strong advantages in terms of latency and energy consumption compared to the original CNN. Additionally, it can be adequately integrated on the commercial imaging system (e.g., camera) without modifying the physical architecture other than the lens with a PSF-engineered meta-optics. Moreover, the ability to encode a colorful image brings up a potential to utilize the encoder for real-world scenes. However, sacrifice of the accuracy is extremely crucial and not negotiable for cases where the safety matters (e.g., autonomous driving vehicles). In other words, if the object classification is applied for statistical analysis (where the ensemble average can minimize the individual inaccuracy), we can endure the loss of classification accuracy. Habitat monitoring drones can be an example. Especially, in case of drones, restrictions to minimize their weight is crucial, which force it store only essential features. Then on-site data processing can be beneficial. As our optical encoder can minimize the latency and energy consumption for the on-site data processing, the habitat drones can investigate much larger areas with a single flight.

Chapter 7

BALANCED NEURAL TANGENT KNOWLEDGE DISTILLATION FOR INCREMENTAL LEARNING

7.1 Motivation

While knowledge distillation and small neural networks have shown promising results on MNIST and CIFAR-10, their effectiveness diminishes when applied to large-scale datasets. The challenge lies in their limited capacity to retain rich feature representations and generalize across complex distributions. As dataset complexity increases, the performance gap between distilled models and full-scale architectures widens, making existing distillation techniques insufficient for high-dimensional tasks (as shown in Figure 7.1).

Here, we propose a novel Gradient Tangent Kernel (GTK) loss, inspired by tangent kernel theory. Neural tangent kernels describe the evolution of a neural network's weights during training [165, 166]. To transfer learned generic representations, we use the previous tangent kernel to regulate the student model's tangent kernel. We define GTK loss as a cosine similarity loss that minimizes discrepancies between the gradients (with respect to parameters) of the previous and current tangent kernels. GTK differs from previously proposed tangent kernels, as NTK is designed only for infinite-width neural networks, while GTK is applicable to finite-width networks [86, 34].

To validate the effectiveness of GTK loss, we applied it to Class Incremental Learning (CIL), which focuses on methods where agents are expected to learn incrementally as new tasks arrive, with only limited exemplars from previous tasks [167, 168]. This learning scenario differs significantly from conventional learning, especially for classification tasks, due to the data imbalance between current and previous tasks [79]. Since full past data are not retained, the available data for previous tasks is significantly less than that for the

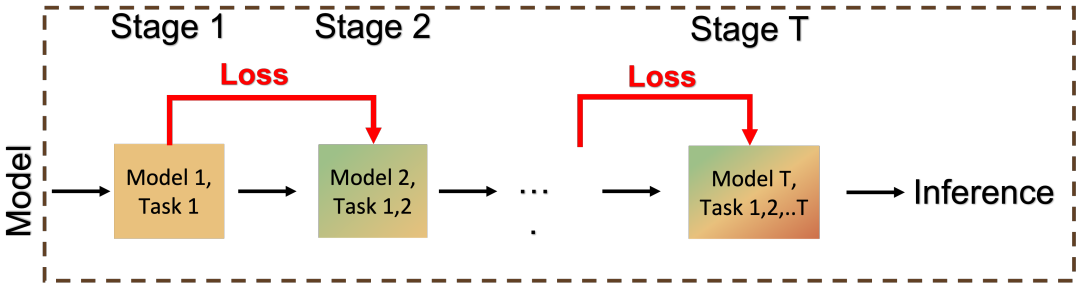


Figure 7.1: Continual Learning Framework: Models are trained sequentially across multiple tasks. Each model is associated with a specific set of tasks, and loss functions are used to update model parameters for the current tasks. As the training progresses, new models incorporate knowledge from previously learned tasks, culminating in a generalized pseudo-task. During inference, the current model is able to make predictions based on the accumulated knowledge. (Figure from [JX5])

current task. Therefore, the CIL aims to achieve an equilibrium between current and previous representations [34].

Instead of training a fixed model for all tasks, we propose to learn a set of generalized parameters and consolidate them into a generic model. The model predicts the task that is at hand and would adapt to the corresponding task automatically during inference. To achieve that, the gradients with respect to the parameters of the generic model are supposed to consider the contributions of all tasks in a balanced way [169]. Thus, we propose a novel Tangent Kernel Optimization approach for Incremental Learning (TKIL). During updates of the generic model, TKIL tunes different task-specific models representing current and previous tasks using GTK loss, then averages the task-specific gradient updates to update the generic model. Since TKIL updates the task parameters separately, it overcomes the bias towards the current task by design. In these updates, the GTK loss transfers knowledge from the previous model to task-specific models, preventing them from diverging. During inference, the generic model collaborates with the inference pipeline to predict the task at

hand when given testing points and adapts itself to that task for classification.

In summary, our main contributions in this section are: **1)** We propose a novel class incremental learning approach, TKIL, that addresses the imbalances in memory-based incremental learning. **2)** The core of TKIL is a novel tangent kernel (GTK) loss for finite-width neural networks. We show that minimizing GTK loss associated with TKIL achieves more balanced representations. Such representations allow TKIL to generate robust task predictions and to update corresponding task-specific models during inference. **3)** Extensive experiments on MNIST, SVHN, CIFAR-100, and ImageNet show that TKIL achieves robust accuracy on task predictions and this accuracy translates to outperforming existing incremental learning methods.

7.2 Methods

We denote a sequence of data as a batch of datasets $\mathcal{D}=\{D_1, D_2, \ldots, D_T, \ldots\}$. D_T is T-th dataset in $\mathcal{D}$, which contains the training images $\boldsymbol{x}=\{x_i\}_{i=1}^n$ and labels $\boldsymbol{y}=\{y_i\}_{i=1}^n$. Each D_T dataset includes m classes, with total $m \times T$ classes. At T-th incremental stage, only complete data for current classes D_T and a small set of previous exemplars $\mathcal{M}_T=\{M_1 \subseteq D_1, M_2 \subseteq D_2, \ldots, M_{T-1} \subseteq D_{T-1}\}$ in a fixed memory buffer are available for training. We denote the T-th generic model in incremental learning as $F_T=F(\boldsymbol{\theta}_T, \boldsymbol{x})$, where **bold** $\boldsymbol{\theta}_T = \{\phi_T, \theta_T\}$ is all parameters in the network.

Incremental learning progressively learns N tasks with m classes per task. We consider $F_T(\boldsymbol{\theta}_T)$ as T-th stage generic model with feature extractor layers and fully connected layers. Training for the first task ($T = 1$) is straightforward. We initialize the parameters of the generic model ($\boldsymbol{\theta}_1$) with Gaussian distribution and optimize the model with D_1 with BCE classification loss.

The tangent kernel optimization approach (see Algorithm 1) is used to train subsequent T-th tasks ($T \in [2, N]$). In particular, we collect current data D_T and memory buffer M_T as the training data and divide them into multiple large batches. Our approach combines **Task-Specific Model Training** and **Generic Model Update** as a single step, and then,

repeats the step for all batches. *Task-Specific Model Training* aims to preserve learned representations from the previous generic model. We separate different task images from one large batch into T mini-batches and feed them to T task-specific models only once. *Generic Model Update* aims to find an unbiased updating direction for all T tasks. To accomplish this, T task-specific models are collapsed into a single generic model that can accommodate T tasks. We describe the training procedures in detail below.

Task-Specific Model Training: When a new task is considered for the T-th incremental learning stage, TKIL first trains T task-specific models. Each task-specific model $F_{T,task\ i} (i \leq T)$ is initialized with parameters of the current generic model ($\boldsymbol{\theta}_T$). One large batch B_T from $\mathcal{M}_T \cup D_T$ is being separated into different tasks as T *mini-batches*. For each task i, an i-th task-specific model $F_{T,task\ i}$ is created and is updated only with i-th mini-batch that corresponds to this task. To transfer learned representations from the previous generic model to task-specific models, we employ different losses as follows.

(1) For task-specific models representing previous tasks ($F_{T,task\ i}, i \in [1, T-1]$), we use three losses: Classification, Knowledge Distillation and GTK. Classification loss minimizes the difference between predicted logits $F(\boldsymbol{x})$ and labels $\boldsymbol{y}$. KD Loss penalizes the change with respect to the output from the previous generic model as a BCE loss. GTK loss rectifies the gradients and avoids divergence from the learned feature representations. Thus, the overall objective function $\mathcal{L}_{all}$ contains three loss functions, expressed as (also described in Algorithm 1, lines 8-12)

$$\mathcal{L}_{Class} = \mathcal{L}(F_{T,\ task\ i}(\boldsymbol{x}), \boldsymbol{y}), \tag{7.1}$$

$$\mathcal{L}_{KD} = \mathcal{L}(F_{T-1}(\boldsymbol{x}), F_{T,\ task\ i}(\boldsymbol{x})), \tag{7.2}$$

$$\mathcal{L}_{GTK} = \mathcal{L}(G_{T,\ task\ i}, G_{T-1}), \tag{7.3}$$

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[\alpha\mathcal{L}_{Class} + \beta\mathcal{L}_{KD} + \gamma\mathcal{L}_{GTK}], \tag{7.4}$$

Where α, β and γ are hyperparameters.

(2) For the T-th task-specific model, $F_{T,task\ T}$, we employ the classification loss only as at the first time that this task is being learned. The loss is expressed as (also shown in

Algorithm 1, lines 14-15)

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[\mathcal{L}(F_{T,\ task\ T}(\boldsymbol{x}), \boldsymbol{y})]. \tag{7.5}$$

When sequentially introducing more tasks, imbalances in training data points result in a larger mini-batch size for the current task model $F_{T,task\ T}$ (e.g.,, 512 samples) compared to other tasks (e.g.,, 128 samples). Since TKIL updates the task parameters separately, it inherently overcomes the bias of task imbalances.

Generic Model Update: At the end of batch training, the trained task-specific models are collapsed to a single generic model F_T, which represents the average direction of tasks [170, 171]. Since dynamic expansion methods show that each task-specific model represents one gradient updating direction, we obtain the generic model by computing the average of these updates [172] (also shown in Algorithm 1, line 19)

$$F_T = \frac{1}{T}\sum_{i}^{T} F_{T,task\ i}. \tag{7.6}$$

Due to imbalanced distribution of data between the current data and the memory buffer, training a fixed model becomes more biased towards the current task, e.g.,, BIC and SS-IL train mixed samples from the memory buffer and the current data together [173, 174]. Dynamic expansion approaches train separate task parameters to mitigate overfits to the current task (e.g.,, iTAML, KD), but they fail to maintain learned representations in the previous generic model [172, 79]. TKIL trains task models and leverages GTK loss to avoid dramatic changes in feature representations. Therefore, as the training progresses, the generic model is enhanced by learning new tasks while simultaneously preserving previous representations.

7.3 Experiments and Results

7.3.1 Results: CIFAR-100, ImageNet-100, MNIST, and SVHN

We test TKIL on various incremental learning scenarios and compare it with existing methods on CIFAR-100 and ImageNet-100 in Table 7.1. TKIL achieves more optimal accuracy than other compared methods in all scenarios. The margin in the accuracy of TKIL vs. existing

Algorithm 1 TKIL Algorithm in one batch

Require: Dataset: D_T, Memory: $\mathcal{M}_T$, batch $B_T \subseteq \{D_T \cup \mathcal{M}_T\}$, Hyperparameters: α, β, γ

Initialize the current Model (For the first mini-batch):

  $F_{T-1}(\phi_{T-1}, \theta_{T-1}) \longrightarrow F_T(\phi_T, \theta_T)$

Task-Specific Model Training:

Separate B_T into T mini-batch

for $i = 1, 2, 3, ..T$ **do**

  $F_T \longrightarrow F_{T,task\ i}$

  if $i < T$ **then**

    $\mathcal{L}_{\text{Class}} = \sum_j^{task\ i} \mathcal{L}_{\text{BCE}}(F_{T,task\ i}(\boldsymbol{x}_j), \boldsymbol{y}_j)$

    $\mathcal{L}_{\text{KD}} = \sum_j^{task\ i} \mathcal{L}_{\text{BCE}}(F_{T,task\ i}(\boldsymbol{x}_j), F_{T-1}(\boldsymbol{x}_j))$

    $\mathcal{L}_{\text{GTK}} = \sum_j^{task\ i} \mathcal{L}_{\text{BCE}}(G_{T,task\ i}(\boldsymbol{x}_j), G_{T-1}(\boldsymbol{x}_j))$

    $\mathcal{L}_{\text{all}} = \alpha\mathcal{L}_{\text{Class}} + \beta\mathcal{L}_{\text{KD}} + \gamma\mathcal{L}_{\text{GTK}}$

    Backpropagation for $F_{T,task\ i}$

  else if $i = T$ **then**

    $\mathcal{L}_{\text{Class}} = \sum_j^{task\ T} \mathcal{L}_{\text{BCE}}(F_{T,task\ T}(\boldsymbol{x}_j), \boldsymbol{y}_j)$

    Backpropagation for $F_{T,task\ T}$

  end if

end for

Generic Model Update:

$F_T \longleftarrow \frac{1}{T}\sum_i^T F_{T,task\ i}$

approaches is particularly evident in large tasks (stages) scenarios, and is consistent with the intent of TKIL to balance performance across tasks and classes. For example, on CIFAR-100 with 25 incremental stages, TKIL achieves an improvement of 9.4% in accuracy vs. the second-best method, AFC. In such a scenario, the accuracy of the existing state-of-the-art method, iTAML, drops to 55.9% from 77.6% due to unstable prediction of tasks, where AFC becomes leading method among existing methods with 64.1%. In Table 7.2, we conduct

Table 7.1: Performance comparison between TKIL and other SOTA methods on CIFAR-100 (left-half) and ImageNet-100 (right-half). (Table from [JX5])

Methods	**CIFAR-100**, Memory size $\mathcal{M} = 2k$					**ImageNet-100**, Memory size $\mathcal{M} = 2k$				
Stages	25	10	5	5	10	25	10	5	5	10
New classes per stage	2	5	10	20	10	2	5	10	20	10
Joint Training (Upper Bound)	86.3%			84.6%		81.3%			76.8%	
iCaRL [34]	50.6%	53.8%	58.1%	57.2%	52.6%	54.6%	60.8%	65.6%	60.1%	59.6%
iTAML [172]	55.9%	74.9%	75.4%	74.5%	74.6%	64.7%	69.5%	71.9%	69.3%	70.4%
RMM [175]	59.5%	60.9%	69.5%	62.7%	60.6%	68.8%	71.4%	73.8%	70.5%	69.4%
SS-IL [173]	58.0%	71.5%	75.1%	74.8%	71.1%	69.5%	71.7%	73.5%	68.8%	67.6%
Mnemonics [176]	61.0%	62.3%	64.1%	63.3%	62.2%	69.7%	71.4%	72.6%	70.6%	70.4%
PODNet [81]	62.7%	64.1%	64.5%	58.9%	59.7%	68.3%	74.3%	75.6%	72.5%	71.5%
AFC [79]	64.1%	64.3%	65.9%	64.9%	64.4%	73.4%	75.8%	75.9%	72.9%	71.7%
TKIL (Ours)	**73.5%**	**80.5%**	**83.6%**	**80.6%**	**82.5%**	**77.3%**	**78.5%**	**79.7%**	**75.7%**	**75.3%**

similar experiments on MNIST and SVHN. TKIL achieves 97% or higher accuracy on these benchmarks when learning 2 classes each time. We also observe that the accuracy of TKIL is approaching the upper bound in both two datasets (76% for ImageNet-100 and 85% for CIFAR-100). As a result, TKIL translates to more optimal accuracy for multi-class problems.

7.3.2 Results: ImageNet-1k

Table 7.3 shows the results of CIL algorithms and TKIL on a large-scale dataset with different memory settings. Our results indicate that TKIL is the most accurate method in all settings. The second accurate baseline, SS-IL, is lower by about 5% when applied to a compact model, ResNet-18. This gap indicates that the forgetting constraints strategies fail to maintain the generalization in feature layers. Since SS-IL trains a fixed model and only tunes classification layers for all tasks, the generic parameters in feature extractor layers are not learned. We

Table 7.2: Performance comparison between the TKIL and other state-of-the-art methods on MNIST and SVHN (5 stages, 2 new classes per stage, Memory size = $2k$). (Table from [JX5])

Methods	**MNIST**	**SVHN**
EWC [177]	19.80%	18.21%
RPS-net [178]	96.16%	88.91%
iTAML [172]	97.15%	92.93%
TKIL (Ours)	**97.91% (+0.76%)**	**97.51% (+5.58%)**

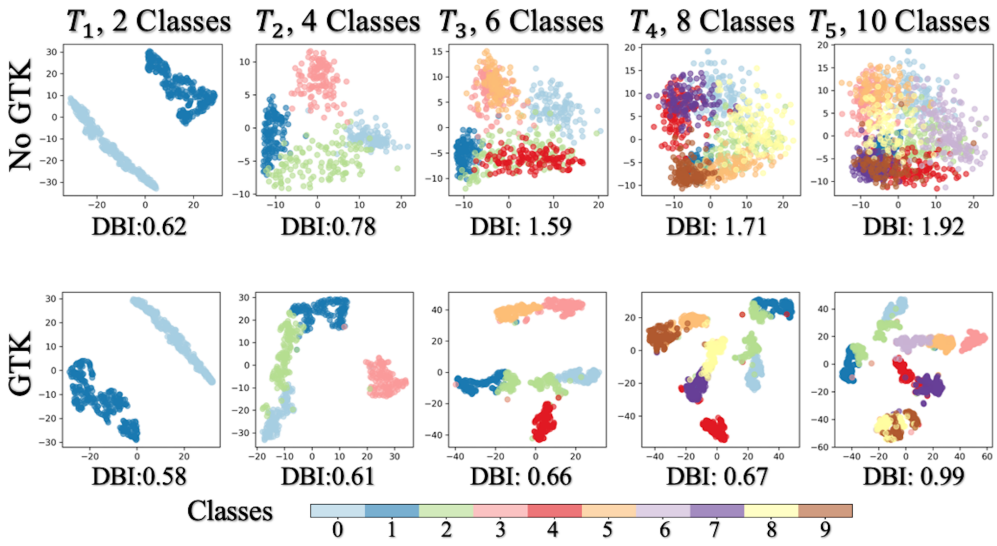


Figure 7.2: T-SNE visualization (colors indicate classes) of feature representations from the last feature extractor on MNIST with 5 Tasks and 2 new classes introduced in each stage. Representations by GTK (bottom) are clustered more efficiently (supported by Davies-Bouldin Index (DBI - lower better)) than representations that do not use GTK (top). (Figure from [JX5])

reaffirm it with more comparisons in the ablation study, which shows that TKIL is more robust than SS-IL and outperforms SS-IL in all incremental learning settings in CIFAR-100

Table 7.3: Performance comparison between TKIL and other SOTA methods on ImageNet-1k. Memory size: $\mathcal{M}$, Upper Bound: Joint Training. (Table from [JX5])

Methods	**ImageNet-1k**	**ImageNet-1k**	
	$\mathcal{M} = 10k$	$\mathcal{M} = 20k$	$\mathcal{M} = 20k$
Stages	10	5	10
New classes per Stages	100	200	100
Joint Training (ResNet-18)		67.9%	
Joint Training (ResNet-34)		71.2%	
iCaRL (ResNet-34) [34]	44.8%	51.5%	46.8%
BIC (ResNet-34) [174]	48.5%	62.6%	58.7%
Mnemonics (ResNet-34) [176]	48.6%	64.5%	63.5%
PODNet (ResNet-34) [81]	48.8%	64.1%	62.0%
SS-IL (ResNet-18) [173]	57.3%	59.6%	59.4%
SS-IL (ResNet-34) [173]	64.5%	65.5%	65.2%
TKIL (ResNet-18)	**64.9%**	**66.9%**	**65.7%**
TKIL (ResNet-34)	**65.6%**	**68.9%**	**67.9%**

and ImageNet-100.

7.3.3 Qualitative Results

In Figure 7.2, we illustrate t-SNE visualization of features representation with MNIST dataset. The feature representations are taken from the final layer of the feature extractor and are projected into a 2D space. We find that the features without GTK (top) are not well clustered when adding more stages, while representations obtained with GTK loss (bottom) are more efficiently clustered, as indicated by the lower Davies-Bouldin Index. The possible reason for

the successful clustering in TKIL could be that GTK Loss reduces divergence between task models. Despite the utilization of dynamic expansions in the non-GTK approach, a high level of variation among task-specific models results in an overfitted generic model, especially when scaling up to more stages. In this scenario, the non-GTK approach separates only the newly added classes (i.e., the brown class), but fails to classify the previously learned tasks (i.e., the navy blue class). Therefore, we observe that GTK loss has a more robust effect on preserving learned representations compared to conventional methods such as KD or MSE loss.

Chapter 8

NEURAL TANGENT KNOWLEDGE DISTILLATION FOR OPTICAL CONVOLUTIONAL NETWORKS

8.1 Motivation

Optical Neural Networks (ONNs) offer a promising approach to achieve efficient computation and energy use compared to digital implementations such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), making them well-suited for resource-constrained, real-time physical systems [179, 180]. For example, ONNs have been proposed for power-limited applications (illustrated in Figure 8.1.a), including satellites [181], unmanned aerial vehicles [182], smart home devices [183], autonomous driving systems [3], and wearable electronics [184].

Among different ONN implementations, hybrid optical-electronic architectures are practical options under current hardware constraints [179]. In such systems, the optical frontend accelerates computation at the speed of light, while the digital backend refines predictions to improve robustness [29]. The optical frontend generally consists of a single linear layer (as shown in Figure 8.1.a), since: (1) implementing nonlinear activation functions in physical optics remains extremely challenging due to material and device limitations; and (2) without nonlinearity, multiple linear transformations can be mathematically compressed into a single linear transformation. Moreover, from a theoretical standpoint, the universal approximation property (UAP) could still be satisfied by shallow networks, suggesting that hybrid ONNs retain sufficient expressive power for a wide range of tasks when appropriately optimized [185, 186, 187].

Despite their promises, ONNs remain difficult to design and train due to both **architectural limitations** and **fabrication-related challenges**. First, existing ONN architectures

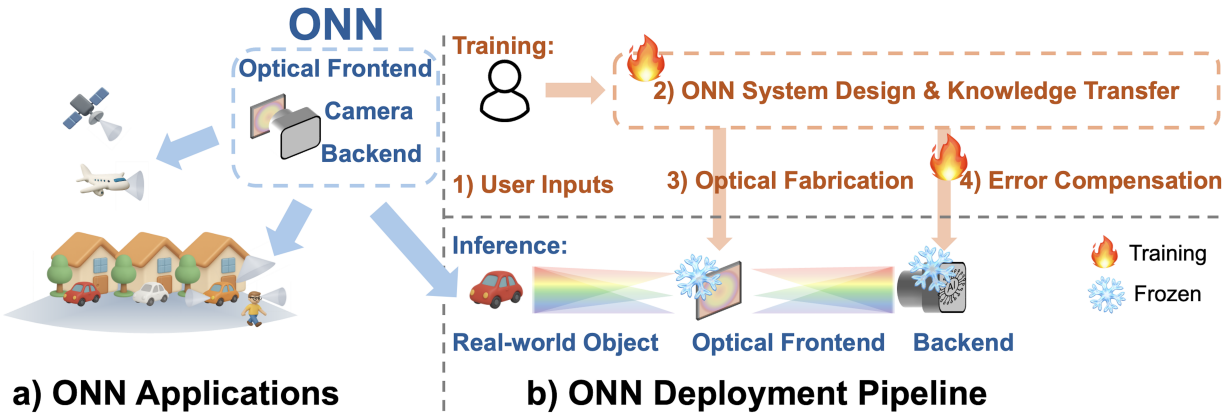


Figure 8.1: Overview of potential applications for ONNs and our proposed deployment pipeline. (a) ONNs for real-time decision-making in power-constrained scenarios. (b) Our proposed pipeline includes user-driven design, knowledge transfer training, fabrication, and error compensation. (Figure from [JX6])

are typically significantly simpler than modern deep CNNs or ViTs. These simplifications cannot be directly obtained through pruning or quantization [30, 31, 188]. Second, physical fabrication and experimental deployment inevitably introduce various sources of noise, such as optical misalignment, material variability, and measurement noise, further degrading performance [54]. While some end-to-end optimization strategies have been proposed to address these challenges, they are typically designed for a specific dataset (e.g., MNIST) and tailored to a particular optical system, rather than providing a generalized solution (as also summarized in Related Works). In contrast, we aim to develop a task-agnostic and hardware-agnostic pipeline that can generalize across different datasets and optical hardware setups.

To address these challenges, knowledge transfer, particularly Knowledge Distillation (KD), offers a promising solution by transferring knowledge from pre-trained digital networks to optical models [JX1]. Moreover, recent work shows that successful KD implicitly leads to student-teacher Neural Tangent Kernel (NTK) similarity, where NTK captures how the

network's predictions change with respect to small changes in its parameters [189]. As we show here, utilizing NTK for matching is particularly effective for ONNs, as the NTK provides a linear approximation of network behavior, naturally aligning with the linear operations performed by optical systems.

Thus, we propose a Neural Tangent Knowledge Distillation (NTKD) pipeline that generalizes across different optical network designs and datasets to support multiple tasks such as classification and segmentation (also shown in Figure 8.1.b). The pipeline starts with specifying the task, the dataset, and the optical structure. Then, NTKD optimization transfers knowledge from digital teacher models to hybrid ONNs by matching their NTKs, effectively transferring the relational structure between classes rather than just matching final predictions. Furthermore, the pipeline compensates for errors introduced during fabrication and experimental deployment by aligning the student's and teacher's NTKs through a small fraction (e.g., 10%) of real experimental data.

In summary, our contributions are as follows:

• We introduce a Neural Tangent Knowledge Distillation (NTKD) pipeline that supports diverse tasks and optical structures, addressing the challenges of shallow architectures and physical imperfections.

• We experimentally validate our pipeline with different ONN implementations on both classification and segmentation tasks, demonstrating its effectiveness through both simulations and fabrications.

• We leverage NTK analysis to estimate the achievable accuracy of given hybrid ONNs, providing theoretical guidance on their design and optimization.

8.2 Methods

8.2.1 Optical Frontend Design

At the initialization of the pipeline, user inputs are required to define the optical system (also shown in Figure 8.2.1). Specifically, the user specifies (1) the physical size of the optical

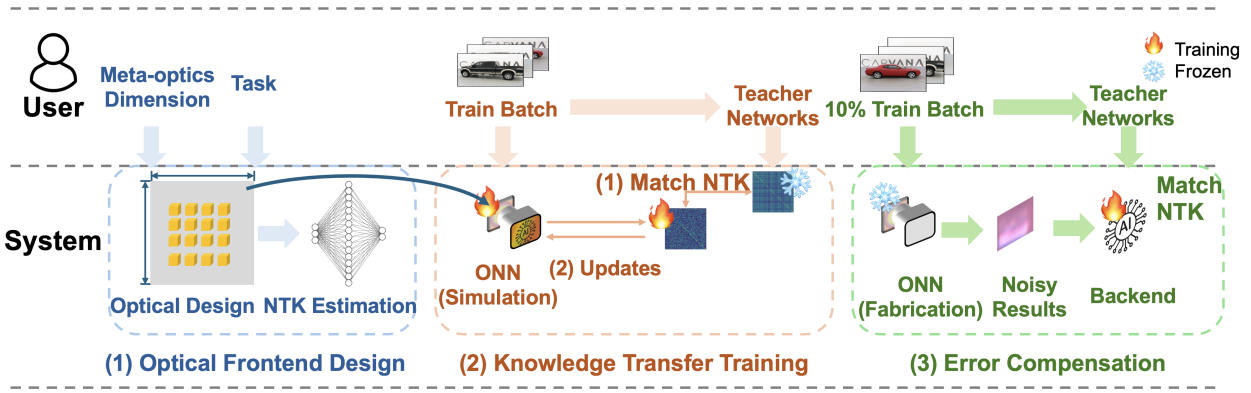


Figure 8.2: Overview of the pipeline. It consists of three steps: (1) Optical Frontend Design based on user-specified inputs, (2) Knowledge Transfer Training using Neural Tangent Kernel (NTK) matching, and (3) Error Compensation for fabricated optical frontends. (Figure from [JX6])

frontend (e.g., the number of meta-optic kernels), (2) the target dataset for the task, and (3) the desired network structure, such as the number of layers and channels. The optical convolution is realized either through a 4f system or via point spread function (PSF)-based free-space propagation [JX4][51].

Optical Frontend Layout: We consider a metasurface of size (h, w), onto which we aim to place $n_{kernels}$ square optical kernels, each of size k (in mm), with a minimum edge-to-edge spacing d to satisfy fabrication constraints. To compute the maximum number of kernels that can be placed while preserving symmetry, we define

$$n_{\text{cols}} = \left\lfloor \frac{w-d}{k+d} \right\rfloor, \quad n_{\text{rows}} = \left\lfloor \frac{h-d}{k+d} \right\rfloor, \quad n_{kernels} = n_{\text{cols}} \times n_{\text{rows}}. \tag{8.1}$$

Performance Estimation: Once the physical layout of the ONN is determined, we aim to estimate its expected performance without empirical training. We adopt the Neural Tangent Kernel (NTK) framework, which captures the training dynamics of **infinitely wide neural networks** under gradient descent. In particular, we introduce a reference network that

shares the same architecture as the designed ONN (e.g., number of layers and connectivity) but has infinite width at each layer. Under this assumption, the predictions of the reference network correspond to NTK regression [86, 190]. Let the reference network $f(x;\theta)$ be a neural network parameterized by θ, which maps an input x to an output $f(x;\theta)$. The NTK is defined as

$$\Theta(x, x') = \nabla_\theta f(x;\theta)^\top \nabla_\theta f(x';\theta), \tag{8.2}$$

where $\nabla_\theta f(x;\theta)$ is the Jacobian of the network output with respect to its parameters. Let $\{x_i^{\text{train}}, y_i^{\text{train}}\}_{i=1}^{n_{\text{train}}}$ be the training data and $\{x_i^{\text{test}}\}_{i=1}^{n_{\text{test}}}$ be the test data. We compute

$$\Theta_{\text{train,train}} = \Theta(x^{\text{train}}, x^{\text{train}}) \in \mathbb{R}^{n_{\text{train}} \times n_{\text{train}}}, \quad \Theta_{\text{test,train}} = \Theta(x^{\text{test}}, x^{\text{train}}) \in \mathbb{R}^{n_{\text{test}} \times n_{\text{train}}}. \tag{8.3}$$

and use kernel regression to predict outputs on the test set

$$f(x^{\text{test}};\theta) = \Theta_{\text{test,train}} \left(\Theta_{\text{train,train}} + \lambda I\right)^{-1} y^{\text{train}}, \tag{8.4}$$

where λ is a regularization parameter, which is selected via grid search on a validation set.

The NTK-based performance estimation serves as a diagnostic tool to evaluate whether the specified ONN architecture is expressive enough for the given task. While this estimation is not used for training or loss computation, it provides an early signal to guide architectural decisions and allows users to iteratively refine the optical design before full training and fabrication. For example, if the estimated test accuracy is much lower than the expected performance, it may suggest a mismatch between the ONN's capacity (e.g., depth) and task complexity.

8.2.2 Knowledge Transfer Training

After the user specifies the ONN architecture, we train the system for the target task. We define a supervised learning problem with input-output pairs (x, y), where x represents the input samples and y denotes the corresponding ground-truth labels. The network parameters (θ), including the optical frontend and the digital backend, are initialized and optimized jointly (shown in Figure 8.2.2).

End-to-end loss: The first loss term we consider is a standard end-to-end supervision loss, which directly minimizes the discrepancy between the network's predictions and the ground-truth labels. Formally, we optimize the following objective

$$\mathcal{L}_{\text{E2E}} = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell(f_{ONN}(x;\theta), y) \right], \tag{8.5}$$

where $\ell(\cdot,\cdot)$ is a standard loss function such as cross-entropy for classification tasks, $f_{ONN}(x;\theta)$ denotes the output of the network with parameters θ, and $\mathcal{D}$ represents the training dataset.

Neural Tangent Knowledge Distillation (NTKD) Loss: In addition to the standard end-to-end loss, we introduce a knowledge transfer loss based on Neural Tangent Kernel (NTK). Specifically, we assume access to a pretrained teacher network, such as LeNet for MNIST, AlexNet for CIFAR-10, or U-Net for image segmentation tasks.

Given a minibatch of input samples $\{x_i\}_{i=1}^{n_{\text{batch}}}$, we compute the Jacobian matrices of both the teacher network (f_{teacher}) and student ONN (f_{ONN}) with respect to their parameters $(\theta_{\text{teacher}}, \theta_{\text{ONN}})$

$$J_{\text{teacher}} = \left[\frac{\partial f_{\text{teacher}}(x_i)}{\partial \theta_{\text{teacher}}} \right]_{i=1}^{n_{\text{batch}}}, \quad J_{\text{ONN}} = \left[\frac{\partial f_{\text{ONN}}(x_i)}{\partial \theta_{\text{ONN}}} \right]_{i=1}^{n_{\text{batch}}}. \tag{8.6}$$

The Jacobians $J_{\text{teacher}} \in \mathbb{R}^{n_{\text{batch}} \times p_{\text{teacher}} \times n_{\text{class}}}$ and $J_{\text{ONN}} \in \mathbb{R}^{n_{\text{batch}} \times p_{\text{ONN}} \times n_{\text{class}}}$ may differ in width depending on the number of trainable parameters in each network. Here, p_{teacher} and p_{ONN} denote the number of parameters in the teacher and ONN, respectively. Their corresponding NTK matrices,

$$\Theta_{\text{teacher}} = J_{\text{teacher}} J_{\text{teacher}}^{\top}, \quad \Theta_{\text{ONN}} = J_{\text{ONN}} J_{\text{ONN}}^{\top}, \tag{8.7}$$

are both of size $n_{\text{batch}} \times n_{\text{batch}}$. We define the NTKD loss by minimizing the discrepancy between the NTK matrices of the teacher network and the ONN (e.g., MSE)

$$\mathcal{L}_{\text{NTKD}} = \mathbb{E}_{\{x_i\}_{i=1}^{n_{\text{batch}}}\sim\mathcal{D}} \left[\ell\left(\Theta_{\text{teacher}}, \Theta_{\text{ONN}}\right) \right]. \tag{8.8}$$

Then, we minimize a weighted sum of two losses, controlled by hyperparameters α and β

$$\min_{\theta} \left(\alpha \mathcal{L}_{\text{E2E}} + \beta \mathcal{L}_{\text{NTKD}} \right). \tag{8.9}$$

8.2.3 Error Compensation

The physical fabrication uses the optimized simulation parameters obtained through the process described in Section 8.2.2. The fabrication fixes the optical frontend, and only the digital backend remains tunable. Due to unavoidable fabrication and experimental errors, discrepancies arise between the designed and realized optical system. Given an input image a and convolution kernel k, the ideal output is $y = a * k$. The fabricated optical convolution output with noise at location (i, j) is

$$\tilde{y}_{i,j} = \alpha\beta \sum_{m=1}^{k_{size}} \sum_{n=1}^{k_{size}} a_{i+m-1,j+n-1} \left(k_{m,n} + \delta_{m,n}\right) + \epsilon_{i,j}. \tag{8.10}$$

Here, the scaling factors α (image brightness) and β (image–kernel misalignment) can be experimentally calibrated to match the designed system, while the sensor noise ϵ is primarily determined by the imaging device characteristics. We further quantify the impact of fabrication noise δ (with proof provided in the Appendix). In particular, we show that perturbations in the NTK caused by kernel fabrication errors (δ_{ij}) scale as

$$\|\Delta\Theta_{\text{ONN}}\| \sim O\left(\frac{\|\delta\|}{m}\right), \tag{8.11}$$

where $\Delta\Theta_{\text{ONN}}$ denotes the NTK perturbation, and m is the number of kernels (i.e., the network width). This result indicates that networks with more kernels are inherently more robust to fabrication noise. Indeed, if $m \gg \|\delta\|$, the impact of fabrication noise on the prediction can be considered negligible. If δ is interpreted as a gradient descent step, Eq. 8.11 is consistent with NTK theory: as $m \to \infty$, the NTK (Θ) remains constant during training, implying $\Delta\Theta \to 0$ [86].

To correct these errors, we re-apply minimization in Eq. 8.9, but restrict optimization to the unfrozen backend parameters (shown in Figure 8.2.3). The teacher network takes the raw input images and computes its NTK matrix over a batch of samples, as defined in Eq. 8.7. The student network receives feature maps from the fixed optical frontend, which processes the same batch of input images and produces an NTK matrix of size $n_{\text{batch}} \times n_{\text{batch}}$.

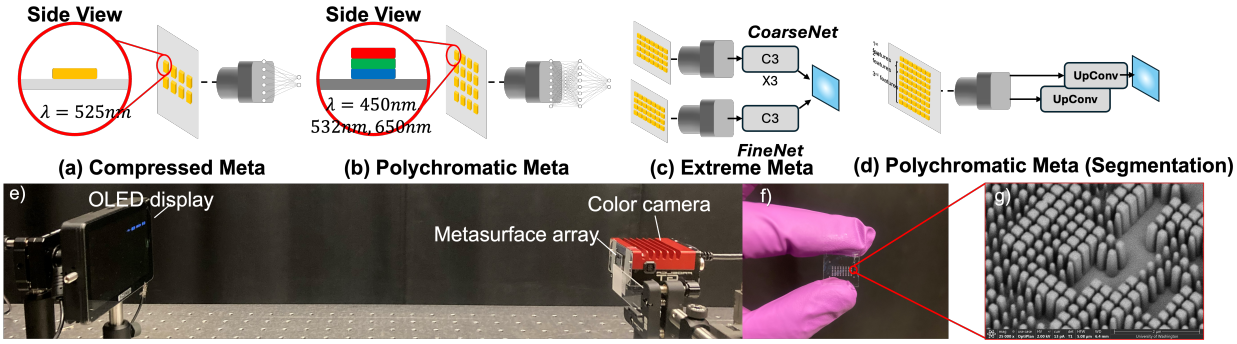


Figure 8.3: Optical systems. (a) Compressed Meta ONN on MNIST [1]; (b) Polychromatic Meta ONN on CIFAR-10 [2]; (c) ExtremeMETA for segmentation with dual optical frontends [3]; (d) Customized Polychromatic Meta ONN for segmentation (Ours); (e) Optical measurement setup; (f) Fabricated PSF-engineered meta-optics; (g) Scanning electron microscopy image of the meta-optics. (Figure from [JX6])

8.3 Experiments and Results

8.3.1 Optical Settings, Pre-trained teachers, Datasets and Evaluation Metrics

Optical Settings: Figure 8.3 demonstrates four different optical systems used in the experiments. For monochromatic image classification, we conducted experiments on the Compressed Meta ONN architecture [1] using the MNIST dataset [191]. This system consists of a single optical frontend with 8 kernels (7×7) and a compact digital backend composed of two fully connected layers. For polychromatic image classification, we evaluated the Polychromatic Meta ONN [JX4] on the CIFAR-10 dataset [121]. This model consists of a single optical frontend with 16 kernels (7×7) and a digital backend consisting of three fully connected layers.

For image segmentation, we performed experiments on both the Extreme Meta ONN [3] and a modified version of the Polychromatic Meta ONN [JX4], using Kaggle's Carvana Image Masking dataset. ExtremeMETA consists of two parallel polychromatic optical frontends, followed by a dual-path digital backend composed of CoarseNet for global feature

extraction and FineNet for detail enhancement, with their outputs fused to produce the final segmentation. Our customized polychromatic segmentation system extends the Polychromatic Meta ONN by incorporating a single optical frontend with 56 kernels (8, 16, and 32 kernels to capture hierarchical depth representations, each of size 3×3) and a backend composed of upconvolutional layers to support dense prediction tasks. For a fair comparison between the two systems, we matched the number of optical kernels.

In experiments, we manipulate polychromatic−red, green, and blue−point spread functions (PSFs) of the meta-optics using a gradient descent algorithm. Physical shapes and dimensions of the PSF-engineered meta-optics are shown in Figure 8.3. Since the meta-optics are designed with PSFs that function as convolutional kernels, optical convolution occurs naturally during image capture. Our experimental setup is straightforward: we replace a conventional imaging lens with PSF-engineered meta-optics. The meta-optics are carefully positioned and aligned with the color camera, enabling the capture of PSFs and convolved images using a laser pointer and an OLED display, respectively. A schematic representation and a photograph of the setup are provided in Figure 8.3.

Pre-trained teachers and Datasets: The MNIST and CIFAR-10 datasets each consist of $50,000$ training images and $10,000$ testing images. The Carvana dataset, originally introduced in Kaggle's Carvana Image Masking Challenge, contains 5,088 high-resolution 1920×1280 car images. We adopt LeNet (99.1% accuracy on MNIST), AlexNet (84.5% on CIFAR-10), and a full U-Net (95.4% mIoU on Segmentation) as teacher models for their respective tasks.

Evaluation Metrics: For classification tasks, we ran each experiment five times with different random seeds and reported the mean and standard deviation (std) of the classification accuracy. For segmentation tasks, we similarly conducted five independent runs and reported the mean Intersection over Union (mIoU) along with the standard deviation.

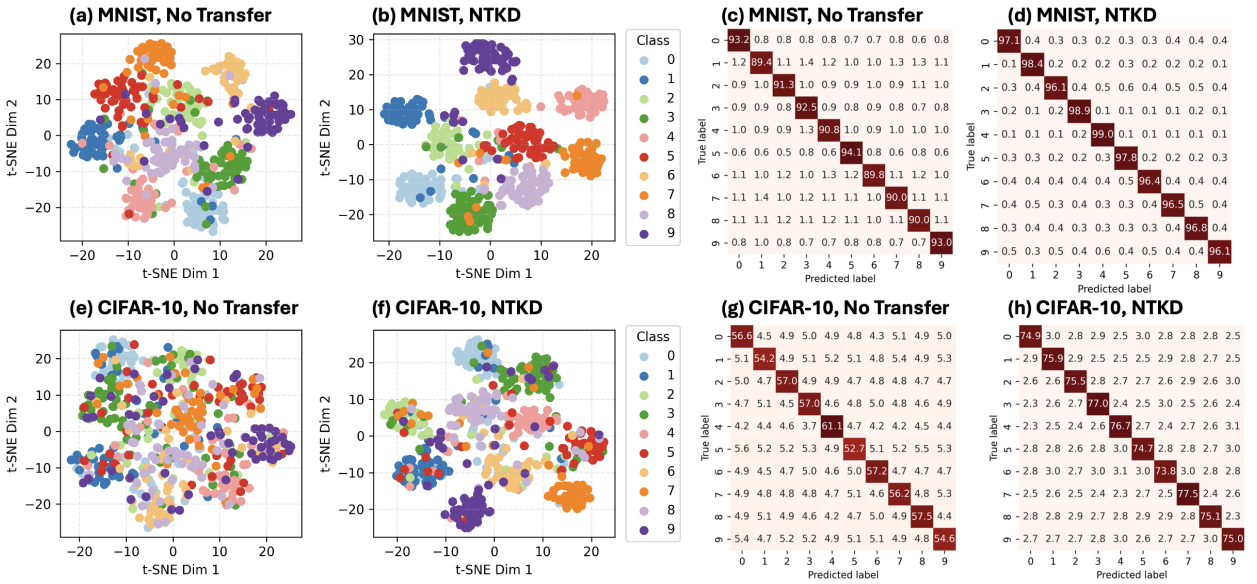


Figure 8.4: Simulation: T-SNE and confusion matrices in MNIST (a-d) and CIFAR-10 (e-h). (Figure from [JX6])

8.3.2 Main Results

Simulation Results: Table 8.1 summarizes the accuracy of different ONN training strategies in classification and segmentation tasks. For monochromatic classification, NTKD achieved 97.3% accuracy and outperformed both KD-based transfer (95.9%) and the non-transfer baseline (91.4%). Similar trends were observed in the more challenging polychromatic

Table 8.1: Performance comparison of ONN methods across different tasks and training strategies. (Table from [JX6])

Methods	**Monochromatic Classification (%)**	**Polychromatic Classification (%)**	**Polychromatic Segmentation (mIoU)**	
	Compressed Meta (2025) [1]	Polychromatic Meta (2025) [JX4]	Extreme Meta (2025) [3]	Polychromatic Meta (Ours)
Simulation, No Transfer	91.4 ± 0.8%	56.4 ± 1.9%	68.3 ± 0.5%	74.3 ± 0.4%
Simulation, KD	95.9 ± 0.6%	72.5 ± 2.1%	75.3 ± 0.2%	86.7 ± 0.4%
Simulation, NTKD	97.3 ± 0.6%	75.6 ± 0.9%	80.1 ± 0.2%	91.5 ± 0.4%

classification setting, where NTKD achieved 75.6%, surpassing KD (72.5%) and baseline (56.4%). For segmentation tasks, we evaluated models on both the Extreme Meta and our proposed Polychromatic Meta datasets. NTKD consistently achieved higher mIoU scores across both optical systems, surpassing KD-based transfer and end-to-end training without transfer (75.3% and 86.7%, respectively).

These results indicate that training ONNs end-to-end without transfer often leads to suboptimal performance. Incorporating knowledge transfer through KD or NTKD improves learning efficacy and overall segmentation quality. In particular, the NTKD approach outperformed KD in different tasks, demonstrating its ability to guide representation learning in optical neural networks.

Figure 8.4 demonstrates knowledge transfer strategies on MNIST and CIFAR-10 representations and classification performance. Compared to the no-transfer baseline, these strategies improve class separability in t-distributed Stochastic Neighbor Embedding (t-SNE) visualizations and reduce noise in confusion matrices. NTKD transfer shows improved clustering and accuracy in experiments.

Fabrication and Compensation Results: Table 8.2 summarizes the impact of error compensation strategies on ONN performance in classification and segmentation tasks. Due to unavoidable fabrication and experimental errors, optical frontends suffer significant accuracy drops, especially in the polychromatic setting. Without compensation, the monochromatic system exhibits an 8.1% drop, and the more fabrication-sensitive polychromatic system shows

Table 8.2: Evaluation of ONN error compensation methods on fabricated optical systems across both classification and segmentation tasks.

Method	**Monochromatic Classification (%)** Compressed Meta (2025) [1]	**Polychromatic Classification (%)** Polychromatic Meta (2025) [JX4]	**Polychromatic Segmentation (mIoU)** Polychromatic Meta (Ours)
No compensation (baseline)	89.2%	47.3%	49.7%
Error compensation (End-to-End)	93.2 ± 0.1%	70.4 ± 2.1%	62.7 ± 0.9%
Error compensation (NTKD)	95.1 ± 0.1%	74.9 ± 1.3%	81.2 ± 0.6%

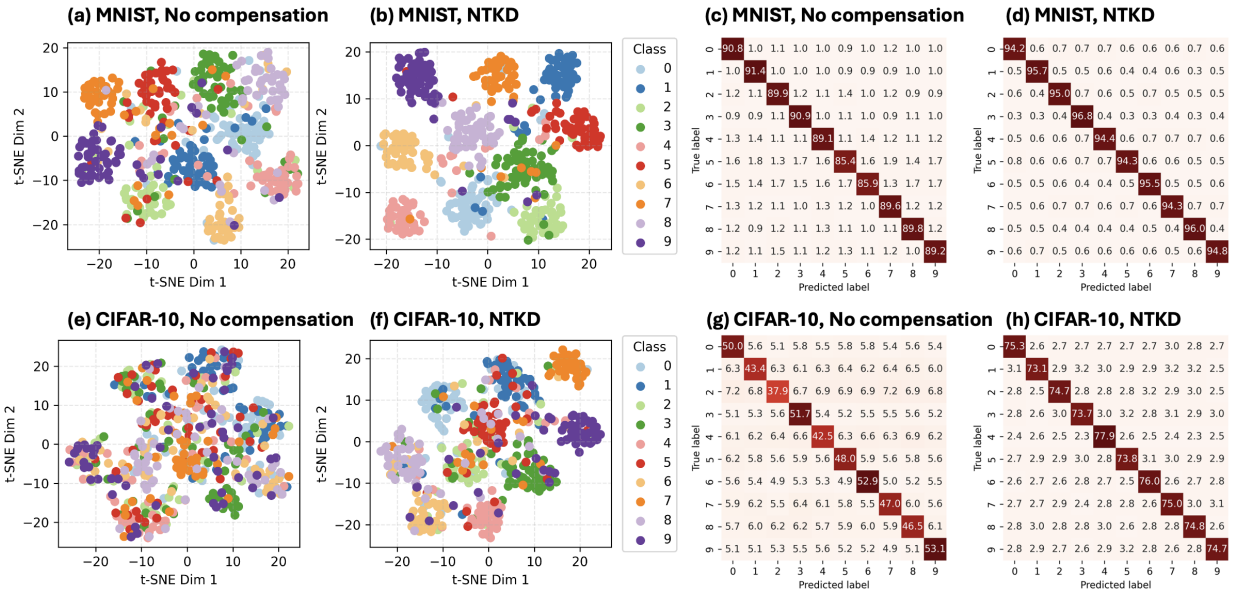


Figure 8.5: Fabrication: T-SNE and confusion matrices on MNIST (a–d) and CIFAR-10 (e–h). (Figure from [JX6])

a 28.3% drop on CIFAR-10 and a 41.8% reduction in mIoU on the Carvana segmentation task. This performance gap highlights the increased challenge of fabricating RGB-sensitive kernels in polychromatic ONNs compared to monochromatic ONNs. Our results demonstrate that knowledge transfer methods are able to assist with denoising that gap. NTKD compensation yields higher accuracy in both cases (95.1% for monochromatic, 74.9% for polychromatic and 81.2% for image segmentation task), outperforming end-to-end deep learning compensation, and validating its effectiveness in robust corrections for fabrication.

Figure 8.5 compares the t-SNE visualizations and confusion matrices of MNIST and CIFAR-10 representations under different compensation strategies. Without compensation (Figures 8.5 a, c, e, g), both optical systems exhibit class overlap in the feature space and reduced classification accuracy, primarily due to fabrication errors and optical misalignments. The NTKD correction (Figures 8.5 b, d, f, h) compensates for these errors and improves the clustering structure and classification accuracy, demonstrating robustness and generalization

Table 8.3: Random PSF kernel design across different tasks. (Table from [JX6])

	Monochromatic Classification (%)	**Polychromatic Classification (%)**	**Polychromatic Segmentation (mIoU)**
8 kernels	96.12%	36.73%	49.3%
500 kernels	96.81%	49.32%	64.1%
1000 kernels	97.24%	56.23%	69.9%
∞ kernels	97.72%	67.33%	72.1%

across datasets and optical systems.

8.4 Discussion

Backend Complexity of ONNs: The complexity of the backend plays a critical role in the performance of optical systems, particularly in hybrid optical-digital architectures. When a strong digital backend is employed, it can effectively denoise and recover accurate outputs, even when the optical frontend introduces significant noise. However, a strong backend not only diminishes the contribution of the optical frontend but also significantly increases power consumption—undermining the core motivation for adopting optical computing in resource-constrained environments. In such scenarios, digital networks (e.g., ViT or U-Net) are often a more practical and effective choice than hybrid ONNs with disproportionately strong backends. For practical deployment, we need to carefully balance the computational load between optics and computational backend and find a tradeoff between acceptable energy consumption/ latency and acceptable accuracy, which will be an application dependent trade-off.

Random vs. Designed Parameters: Another possible direction for designing and training ONNs is to use randomly initialized optical parameters while training only the digital backends. This approach aims to avoid the need for extensive simulation and hardware-in-the-loop optimization of the optical frontends. We conducted experiments using a single optical convolutional layer and a lightweight backend—consisting of a single fully connected layer for classification, or a single upsampling layer for segmentation. As shown in Table 8.3,

increasing the number of random kernels consistently improves performance across tasks. For example, in polychromatic classification, accuracy improves from 36.73% (8 kernels) to 56.23% (1000 kernels), and further to 67.33% in the NTK regime, which approximates an infinite number of random kernels. Similarly, in polychromatic segmentation, mIoU rises from 49.3% to 69.9% and reaches 72.1% under NTK estimation. These results demonstrate that while increasing the number of random PSFs improves performance, they still underperform our approach (designed kernels with knowledge transfer).

Scalability of ONNs: Scaling current ONNs remains challenging, as most designs rely on shallow structures with limited linear computational capacity. Implementing nonlinear operations in ONNs is especially difficult due to physical constraints, such as the limited pixel size. These hardware limitations make it hard for ONNs to support deep and expressive architectures like those used in digital networks. We observed that using different kernels to simulate multiple layers of a digital network leads to better performance, compared to simply compressing a deep CNN into a single-layer ONN. To test the scalability of ONNs under these limitations, we evaluated a polychromatic ONN on the ImageNet-100 dataset and achieved a top-1 accuracy of 46.32%. In summary, scalable and expressive ONNs will ultimately rely on physical advances enabling deep and nonlinear optical computations. To that end, image intensifiers present an interesting opportunity for cascading multiple metasurfaces, as they can provide both nonlinear activation and signal regeneration [47].

Parameter-Space Compressibility: To examine redundancy of the convolutional layer, we compute the Jacobian $J \in \mathbb{R}^{n \times p}$ of LeNet outputs with respect to its 2,572 convolutional parameters. We then form the parameter Gram matrix $J^\top J \in \mathbb{R}^{p \times p}$ and analyze its cumulative eigenvalue spectrum to quantify the parameter effective dimensions. Figure 8.6.a plots the cumulative eigenvalue power of $J^\top J$. 108 parameters account for 95% of the total power, and 288 parameters account for 99%, indicating strong compressibility. To validate this, we trained compressed models with varying parameter counts (Figure 8.6.b), and analyzed the trade-off between model size and accuracy. For example, a model with just 98 parameters in the convolutional layer achieved 95.2% accuracy, closely matching the 95% cumulative NTK

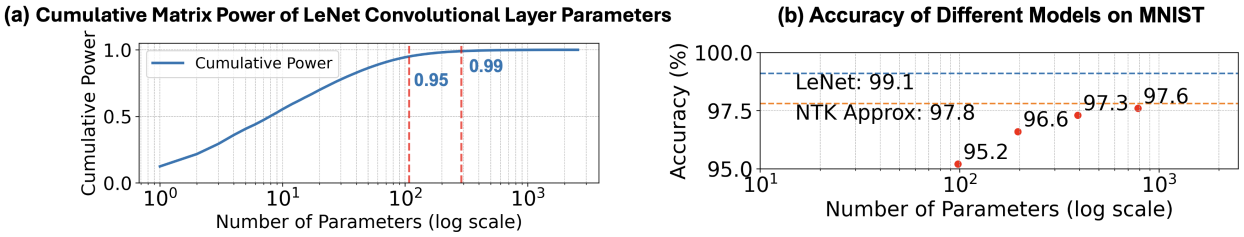


Figure 8.6: NTK compressibility analysis. (a) Cumulative eigenvalue power of the Parameter matrix $g = J^\top J$ for LeNet's convolutional layers, showing that 95% and 99% of power are concentrated in just 108 and 288 parameters. (b) Accuracy of models with varying parameter counts. (Figure from [JX6])

power threshold. Notably, when the cumulative matrix power exceeded 99%, further increases in parameter count without changes to the model structure yielded diminishing returns. For example, increasing parameters from 98 to 784—twice that of Compressed Meta [JX2]—only improved accuracy by 0.5%. This marginal gain did not surpass the accuracy predicted by NTK analysis, highlighting the inefficiency of overparameterization and suggesting the use of dimensionality reduction. The Gram matrix can help estimate the efficient number of parameters for a given task.

Fabrication Analysis: Several factors may help explain the discrepancy between the designed and measured kernels. First, the local periodic approximation — which simplified the metasurface optics (MO) design by assuming the scatterers were arranged periodically — neglected the coupling between adjacent dissimilar scatterers and could introduce phase errors. Second, unavoidable fabrication errors further contributed to the observed discrepancy. Third, the pre-designed kernel needed to be properly matched to the sensor's pixel array; any misalignment between the metasurface optics and the camera could also lead to deformation of the measured kernels. Regarding the polychromatic versus single-wavelength kernels, metasurfaces inherently suffered from strong chromatic aberrations, a universal characteristic of diffractive optics. While we co-optimized the kernel across multiple wavelengths during

the design process to ensure consistent behavior, the performance remained limited by the intrinsic material properties.

MACs and Power Consumption: We estimated the multiply–accumulate operations (MACs) and power consumption of hybrid ONNs using our polychromatic ONN as an example. The total number of MAC operations in the simulated digital network is approximately 239 MMACs (while the full U-Net requires 65.9 GMACs and Efficient U-Net reaches 1.37 GMACs), which is reduced to 65 MMACs after incorporating optical frontends [192]. The total energy consumption includes both image capture and digital computation. The full U-Net (pre-trained teacher) consumes 2.03 J for computation and 2.36 mJ for image capture, totaling **2.04 J** per image. The compact digital network requires 7.37 mJ for computation and 2.36 mJ for image capture, totaling **9.73 mJ** per image. In contrast, our hybrid ONN consumes 3.82 mJ for image capture and 2.01 mJ for backend processing, totaling **5.83 mJ**—representing over a 40% reduction in system-level energy consumption compared to the simulated digital network, and over 300× energy compression compared to the pre-trained teacher U-Net.

Chapter 9

CONCLUSION

In this thesis, I developed knowledge distillation methods to transfer knowledge from deep electronic neural networks into ONNs, with the goal of enhancing ONN performance under realistic physical constraints and narrowing the gap between ONNs and digital networks.

I first applied knowledge distillation to different optical systems with the goal of designing transferable and scalable architectures. Chapter 4 demonstrated knowledge transfer from nonlinear electronic models into $4f$-based ONNs, validating that teacher–student distillation can circumvent nonlinearity. Chapter 5 extended this approach to metasurface-based ONNs, showing that deep electronic convolutional layers can be compressed into a single optical layer while retaining strong performance. Chapter 6 further generalized these studies by introducing a polychromatic optical encoder, broadening the applicability of ONNs and enabling efficient multi-wavelength processing.

Furthermore, I introduced the Tangent Kernel Loss, a novel loss function that leverages tangent kernel theory to improve knowledge transfer in neural networks. Chapter 7 validated this method within a balanced continual learning framework, addressing incremental adaptation under imbalanced data distributions and achieving stable performance across sequential tasks. In Chapter 8, I extended this concept to ONNs by developing the NTKD pipeline. NTKD enables the transfer of feature representations from electronic teacher networks to optical students, providing scalable and architecture-agnostic guidance for hybrid electro-optical systems. The framework was evaluated across diverse computer vision tasks, including image classification and segmentation, demonstrating its effectiveness in both simulated and fabricated ONNs.

In summary, this thesis demonstrates that knowledge distillation, especially NTKD, can

enhance the overall performance of optical neural networks. These methods help narrow the gap between ONNs and electronic neural networks, while also mitigating fabrication errors, proving effective across both simulated and fabricated ONN systems. In the future, a promising direction is to design more complex ONN architectures that incorporate optical nonlinearity and knowledge distillation, thereby further approximating the representational power of modern deep neural networks.

JINLIN XIANG'S PUBLICATIONS

[JX1] Jinlin Xiang, Shane Colburn, Arka Majumdar, and Eli Shlizerman. "Knowledge distillation circumvents nonlinearity for optical convolutional neural networks." *Applied Optics* 61, no. 9 (2022): 2173–2183.

[JX2] Anna Wirth-Singh, Jinlin Xiang, Minho Choi, Johannes Fröch, Luocheng Huang, Eli Shlizerman, and Arka Majumdar. "Compressed Meta-Optical Encoder for Image Classification." In *CLEO: Fundamental Science*, pp. FF1J–1. Optica Publishing Group, 2024.

[JX3] Anna Wirth-Singh, Jinlin Xiang, Minho Choi, Johannes E. Fröch, Luocheng Huang, Shane Colburn, Eli Shlizerman, and Arka Majumdar. "Compressed meta-optical encoder for image classification." *Advanced Photonics Nexus*, 4(2):026009, 2025.

[JX4] Minho Choi, Jinlin Xiang, Anna Wirth-Singh, Seung-Hwan Baek, Eli Shlizerman, and Arka Majumdar. "Transferable polychromatic optical encoder for neural networks." *Nature Communications*, 16(1):5623, 2025.

[JX5] Jinlin Xiang and Eli Shlizerman. "Tkil: Tangent kernel optimization for class balanced incremental learning." In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 3529–3539. 2023.

[JX6] Jinlin Xiang, Minho Choi, Yubo Zhang, Zhihao Zhou, Arka Majumdar, and Eli Shlizerman. "Neural Tangent Knowledge Distillation for Optical Convolutional Networks." *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

[JX7] Mingfei Chen, Zijun Cui, Xiulong Liu, Jinlin Xiang, Caleb Zheng, Jingyuan Li, and Eli Shlizerman. "SAVVY: Spatial Awareness via Audio-Visual LLMs through Seeing

and Hearing." *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

[JX8] Jinlin Xiang, Bozhao Qi, Marc Cerou, Wei Zhao, and Qi Tang. "DN-ODE: Data-driven neural-ODE modeling for breast cancer tumor dynamics and progression-free survivals." *Computers in Biology and Medicine* 180 (2024): 108876.

[JX9] Jinlin Xiang, Hillol Sarker, Bozhao Qi, Ruisu Zhang, Roger Trullo, Salvatore Badalamenti, Maria Wiekowski, Annie Kruger, Etienne Pochet, Qi Tang, *et al.* "Endoscopic Scoring and Localization in Unconstrained Clinical Trial Videos." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 4006–4015, 2025. IEEE.

[JX10] Yang Zheng, Jinlin Xiang, Kun Su, and Eli Shlizerman. "BI-MAML: Balanced Incremental Approach for Meta Learning." *arXiv preprint arXiv:2006.07412*, 2020.

BIBLIOGRAPHY

[1] Anna Wirth-Singh, Jinlin Xiang, Minho Choi, Johannes E Fröch, Luocheng Huang, Shane Colburn, Eli Shlizerman, and Arka Majumdar. Compressed meta-optical encoder for image classification. *Advanced Photonics Nexus*, 4(2):026009–026009, 2025.

[2] Minho Choi, Jinlin Xiang, Anna Wirth-Singh, Seung-Hwan Baek, Eli Shlizerman, and Arka Majumdar. Transferable polychromatic optical encoder for neural networks. *arXiv preprint arXiv:2411.02697*, 2024.

[3] Quan Liu, Brandon T Swartz, Ivan Kravchenko, Jason G Valentine, and Yuankai Huo. Extrememeta: High-speed lightweight image segmentation model by remodeling multi-channel metamaterial imagers. *Journal of Imaging Science and Technology*, pages 1–10, 2025.

[4] Weihan Li, Qian Ma, Che Liu, Yunfeng Zhang, Xianning Wu, Jiawei Wang, Shizhao Gao, Tianshuo Qiu, Tonghao Liu, Qiang Xiao, et al. Intelligent metasurface system for automatic tracking of moving targets and wireless communications based on computer vision. *Nature Communications*, 14(1):989, 2023.

[5] Hojin Jang and Frank Tong. Improved modeling of human vision by incorporating robustness to blur in convolutional neural networks. *Nature Communications*, 15(1):1989, 2024.

[6] Xiao Wang, Brandon Redding, Nicholas Karl, Christopher Long, Zheyuan Zhu, James Skowronek, Shuo Pang, David Brady, and Raktim Sarma. Integrated photonic encoder for low power and high-speed image processing. *Nature Communications*, 15(1):4510, 2024.

[7] Peter L McMahon. The physics of optical computing. *Nature Reviews Physics*, 5(12):717–734, 2023.

[8] Pierre Ambs. Optical computing: A 60-year adventure. *Advances in Optical Technologies*, 2010(1):372652, 2010.

[9] Demetri Psaltis, David Brady, Xiang-Guang Gu, and Steven Lin. Holography in artificial neural networks. *Nature*, 343(6256):325–330, 1990.

[10] Michael Reck, Anton Zeilinger, Herbert J Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical review letters*, 73(1):58, 1994.

[11] Hebb Do. The organization of behavior. *New York*, 1949.

[12] John McCarthy, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4):12–12, 2006.

[13] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[14] Zhiwei Xue, Tiankuang Zhou, Zhihao Xu, Shaoliang Yu, Qionghai Dai, and Lu Fang. Fully forward mode training for optical neural networks. *Nature*, 632(8024):280–286, 2024.

[15] Zhihao Xu, Tiankuang Zhou, Muzhou Ma, ChenChen Deng, Qionghai Dai, and Lu Fang. Large-scale photonic chiplet taichi empowers 160-tops/w artificial general intelligence. *Science*, 384(6692):202–209, 2024.

[16] Xing Lin, Yair Rivenson, Nezih T. Yardimci, Muhammed Veli, Yi Luo, Mona Jarrahi, and Aydogan Ozcan. All-optical machine learning using diffractive deep neural networks. *Science*, 361(6406):1004–1008, 2018.

[17] Deniz Mengu, Yi Luo, Yair Rivenson, and Aydogan Ozcan. Analysis of diffractive optical neural networks and their integration with electronic neural networks. *IEEE Journal of Selected Topics in Quantum Electronics*, 26(1), 2020.

[18] Cong He, Dan Zhao, Fei Fan, Hongqiang Zhou, Xin Li adn Yao Li, Junjie Li, Fei Dong, Yin-Xiao Miao, Yongtian Wang, and Lingling Huang. Pluggable multitask diffractive neural networks based on cascaded metasurfaces. *Opto-Electronic Advances*, 7(23005), 2024.

[19] M. Yang, E. Robertson, K. Esguerra, K. Busch, and J. Wolters. Optical convolutional neural network with atomic nonlinearity. *Optics Express*, 31(10):16451–16459, 2023.

[20] A. Ryou, J. Whitehead, M. Zhelyeznyakov, P. Anderson, C. Keskin, M. Bajcsy, and A. Majumdar. Free-space optical neural network based on thermal atomic nonlinearity. *Photon. Res.*, 9:B128–B134, 2021.

[21] T. Wang, M. M. Sohoni, L. G. Wright, M. M. Stein, S.-.Y. Ma, T. Onodera, M. G. Anderson, and P. L. McMahon. Image sensing with multilayer nonlinear optical neural networks. *Nature Photonics.*, 17:408–415, 2023.

[22] S. Colburn, Y. Chu, A. Majumdar, and E. Shlizerman. Optical frontend for a convolutional neural network. *Applied Optics*, 58(12):3179–3186, 2019.

[23] Carlos Mauricio Villegas Burgos, Tianqi Yang, Yuhao Zhu, and A. Nickolas Vamivakas. Design framework for metasurface optics-based convolutional neural networks. *Appl. Opt.*, 60(15):4356–4365, May 2021.

[24] H. Zheng, Q. Liu, I. I. Zhou, Y. Kravchenko, Y. Huo, and J. Valentine. Meta-optic accelerators for object classifiers. *Science Advances*, 8(30), 2022.

[25] P. L. McMahon. The physics of optical computing. *Nat. Rev. Physics*, 5:717–734, 2023.

[26] T. Zhou, X. Lin, J. Wu, Y. Chen, H. Xie, Y. Li, J. Fan, H. Wu, L. Fang, and Q. Dai. Large-scale neuromorphic optoelectronic computing with a reconfigurable diffractive processing unit. *Nature Photonics*, 15:367–373, 2021.

[27] L. Huang, Q. Tanguy, J. E. Fröch, S. Mukherjee, K. F. Böhringer, and A. Majumdar. Photonic advantage of optical encoders. *Nanophotonics.*, 2023.

[28] H. Zheng, Q. Liu, I. I. Kravchenko, X. Zhang, Y. Huo, and J. G. Valentine. Multichannel meta-imagers for accelerating machine vision. *Nature Nanotechnology.*, 59, 2024.

[29] Shane Colburn, Yi Chu, Eli Shilzerman, and Arka Majumdar. Optical frontend for a convolutional neural network. *Applied optics*, 58(12):3179–3186, 2019.

[30] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[31] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[32] Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3957–3966, 2021.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[34] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[35] Ofri Goldenberg, Boris Ferdman, Elias Nehme, Yael Shalev Ezra, and Yoav Shechtman. Learning optimal multicolor psf design for 3d pairwise distance estimation. *Intelligent Computing*, 2022:0004, 2022.

[36] Eunsue Choi, Gyeongtae Kim, Jooyeong Yun, Yujin Jeon, Junsuk Rho, and Seung-Hwan Baek. 360° structured light with learned metasurfaces. *Nature Photonics*, pages 1–8, 2024.

[37] L Cutrona, EN Leith, C Palermo, and L Porcello. Optical data processing and filtering systems. *IRE Transactions on Information Theory*, 6(3):386–400, 1960.

[38] Julian Bueno, Sheler Maktoobi, Luc Froehly, Ingo Fischer, Maxime Jacquot, Laurent Larger, and Daniel Brunner. Reinforcement learning in a large-scale photonic recurrent neural network. *Optica*, 5(6):756–760, 2018.

[39] Xing Lin, Yair Rivenson, Nezih T Yardimci, Muhammed Veli, Yi Luo, Mona Jarrahi, and Aydogan Ozcan. All-optical machine learning using diffractive deep neural networks. *Science*, 361(6406):1004–1008, 2018.

[40] Alexander A Sawchuk and Timothy C Strand. Digital optical computing. *Proceedings of the IEEE*, 72(7):758–779, 1984.

[41] B Hunt. A matrix theory proof of the discrete convolution theorem. *IEEE Transactions on Audio and Electroacoustics*, 19(4):285–288, 1971.

[42] W. Lohmann and W. T. Rhodes. Two-pupil synthesis of optical transfer functions. *Appl. Opt.*, 17(7):1141–1151, 1978.

[43] J. N. Mait and W. T. Rhodes. Two-pupil synthesis of optical transfer functions: 2: Pupil function relationships. *Appl. Opt.*, 25(12):2003–2007, 1986.

[44] J. N. Mait. Pupil-function design for complex incoherent spatial filtering. *JOSA*, 4(7):1185–1193, 1987.

[45] Mingwei Yang, Elizabeth Robertson, Luisa Esguerra, Kurt Busch, and Janik Wolters. Optical convolutional neural network with atomic nonlinearity. *Optics Express*, 31(10):16451–16459, 2023.

[46] Albert Ryou, James Whitehead, Maksym Zhelyeznyakov, Paul Anderson, Cem Keskin, Michal Bajcsy, and Arka Majumdar. Free-space optical neural network based on thermal atomic nonlinearity. *Photonics Research*, 9(4):B128–B134, 2021.

[47] Tianyu Wang, Mandar M Sohoni, Logan G Wright, Martin M Stein, Shi-Yuan Ma, Tatsuhiro Onodera, Maxwell G Anderson, and Peter L McMahon. Image sensing with multilayer nonlinear optical neural networks. *Nature Photonics*, 17(5):408–415, 2023.

[48] Hanyu Zheng, Quan Liu, You Zhou, Ivan I Kravchenko, Yuankai Huo, and Jason Valentine. Meta-optic accelerators for object classifiers. *Science Advances*, 8(30):eabo6410, 2022.

[49] Tiankuang Zhou, Xing Lin, Jiamin Wu, Yitong Chen, Hao Xie, Yipeng Li, Jingtao Fan, Huaqiang Wu, Lu Fang, and Qionghai Dai. Large-scale neuromorphic optoelectronic computing with a reconfigurable diffractive processing unit. *Nature Photonics*, 15(5):367–373, 2021.

[50] Luocheng Huang, Quentin AA Tanguy, Johannes E Fröch, Saswata Mukherjee, Karl F Böhringer, and Arka Majumdar. Photonic advantage of optical encoders. *Nanophotonics*, 13(7):1191–1196, 2024.

[51] Julie Chang, Vincent Sitzmann, Xiong Dun, Wolfgang Heidrich, and Gordon Wetzstein. Hybrid optical-electronic convolutional neural networks with optimized diffractive optics for image classification. *Scientific reports*, 8(1):1–10, 2018.

[52] Carlos Mauricio Villegas Burgos, Tianqi Yang, Yuhao Zhu, and A Nickolas Vamivakas. Design framework for metasurface optics-based convolutional neural networks. *Applied Optics*, 60(15):4356–4365, 2021.

[53] Zibo Hu, Shurui Li, Russell LT Schwartz, Maria Solyanik-Gorgone, Mario Miscuglio, Puneet Gupta, and Volker J Sorger. High-throughput multichannel parallelized diffraction convolutional neural network accelerator. *Laser & Photonics Reviews*, 16(12):2200213, 2022.

[54] Yanbing Liu, Jianwei Qin, Yan Liu, Xi Yue, Xun Liu, Guoqing Wang, Tianyu Li, Ye Ye, and Wei Li. Physics-constrained comprehensive optical neural networks. *Advances in Neural Information Processing Systems*, 37:92036–92054, 2024.

[55] Kaixuan Wei, Xiao Li, Johannes Froech, Praneeth Chakravarthula, James Whitehead, Ethan Tseng, Arka Majumdar, and Felix Heide. Spatially varying nanophotonic neural networks. *Science Advances*, 10(45):eadp0391, 2024.

[56] Hanyu Zheng, Quan Liu, Ivan I Kravchenko, Xiaomeng Zhang, Yuankai Huo, and Jason G Valentine. Multichannel meta-imagers for accelerating machine vision. *Nature nanotechnology*, 19(4):471–478, 2024.

[57] Cong He, Dan Zhao, Fei Fan, Hongqiang Zhou, Xin Li, Yao Li, Junjie Li, Fei Dong, Yin-Xiao Miao, Yongtian Wang, et al. Pluggable multitask diffractive neural networks based on cascaded metasurfaces. *Opto-Electron. Adv*, 7(2):230005, 2024.

[58] Yuchi Huo, Hujun Bao, Yifan Peng, Chen Gao, Wei Hua, Qing Yang, Haifeng Li, Rui Wang, and Sung-Eui Yoon. Optical neural network via loose neuron array and functional learning. *Nature Communications*, 14(1):2535, 2023.

[59] Md Sadman Sakib Rahman and Aydogan Ozcan. Time-lapse image classification using a diffractive neural network. *Advanced Intelligent Systems*, 5(5):2200387, 2023.

[60] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. In *arXiv preprint arXiv:1609.07061*, 2017.

[61] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in Neural Information Processing Systems*, pages 1269–1277, 2014.

[62] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, 8, 1995.

[63] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[64] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[65] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[66] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell. Characterizing and avoiding negative transfer. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 11293–11302, Long Beach, Jun. 2019.

[67] K. Weiss, T.M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *J. Big Data*, 3(1), Dec. 2016.

[68] J. Huang, A.J. Smola, A. Gretton, K.M. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Proc. 20th Annual Conference on Neural Information Processing Systems*, pages 601–608, Vancouver, Dec. 2006.

[69] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. Bünau, and M. Kawanabe. Direct importance estimation for covariate shift adaptation. *Ann. Inst. Stat. Math.*, 60(4):699–746, Dec. 2008.

[70] Ziyang Zheng, Zhengyang Duan, Hang Chen, Rui Yang, Sheng Gao, Haiou Zhang, Hongkai Xiong, and Xing Lin. Dual adaptive training of photonic neural networks. *Nature Machine Intelligence*, 5(10):1119–1129, 2023.

[71] James Spall, Xianxin Guo, and Alexander I Lvovsky. Hybrid training of optical neural networks. *Optica*, 9(7):803–811, 2022.

[72] Hans-Christian Ruiz Euler, Marcus N Boon, Jochem T Wildeboer, Bram van de Ven, Tao Chen, Hajo Broersma, Peter A Bobbert, and Wilfred G van der Wiel. A deep-learning approach to realizing functionality in nanoelectronic devices. *Nature nanotechnology*, 15(12):992–998, 2020.

[73] Yubo Zhang, Rui Chen, Minho Choi, Johannes E Fröch, and Arka Majumdar. General image compression using random-psf metasurfaces and computational back-end. In *2024 Conference on Lasers and Electro-Optics (CLEO)*, pages 1–3. IEEE, 2024.

[74] Mengran Zhao, Shitao Zhu, Die Li, Thomas Fromenteze, Mohsen Khalily, Xiaoming Chen, Vincent Fusco, and Okan Yurduseven. Frequency-diverse bunching metasurface antenna for microwave computational imaging. *IEEE Transactions on Antennas and Propagation*, 2024.

[75] Anders Pors, Fei Ding, Yiting Chen, Ilya P Radko, and Sergey I Bozhevolnyi. Random-phase metasurfaces at optical wavelengths. *Scientific reports*, 6(1):28448, 2016.

[76] Matthieu Dupré, Liyi Hsu, and Boubacar Kanté. On the design of random metasurface based devices. *Scientific reports*, 8(1):7162, 2018.

[77] Parker R Wray and Harry A Atwater. Light–matter interactions in films of randomly distributed unidirectionally scattering dielectric nanoparticles. *ACS Photonics*, 7(8):2105–2114, 2020.

[78] Mary Phuong and Christoph Lampert. Towards understanding knowledge distillation. In *International Conference on Machine Learning*, pages 5142–5151, 2019.

[79] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. *arXiv preprint arXiv:2204.00895*, 2022.

[80] Khurram Javed and Faisal Shafait. Revisiting distillation and incremental classifier learning. In *Asian Conference on Computer Vision*, pages 3–17. Springer, 2018.

[81] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European Conference on Computer Vision*, pages 86–102. Springer, 2020.

[82] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

[83] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[84] Philip De Rijk, Lukas Schneider, Marius Cordts, and Dariu Gavrila. Structural knowledge distillation for object detection. *Advances in Neural Information Processing Systems*, 35:3858–3870, 2022.

[85] Zailiang Chen, Xianxian Zheng, Hailan Shen, Ziyang Zeng, Yukun Zhou, and Rongchang Zhao. Improving knowledge distillation via category structure. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 205–219. Springer, 2020.

[86] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[87] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *Advances in Neural Information Processing Systems*, 32, 2019.

[88] Arthur Jacot, François Ged, Berfin Şimşek, Clément Hongler, and Franck Gabriel. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity. *arXiv preprint arXiv:2106.15933*, 2021.

[89] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[90] Theo Gasser and Hans-Georg Müller. Kernel estimation of regression functions. In *Smoothing Techniques for Curve Estimation: Proceedings of a Workshop held in Heidelberg, April 2–4, 1979*, pages 23–68. Springer, 1979.

[91] Bram Wallace, Ziyang Wu, and Bharath Hariharan. Can we characterize tasks without labels or features? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1245–1254, 2021.

[92] E Kakkava, N Borhani, Y Pu, C Moser, and D Psaltis. Image classification and reconstruction through multimode fibers by deep neural networks. In *2018 Conference on Lasers and Electro-Optics Pacific Rim (CLEO-PR)*, pages 1–2. IEEE, 2018.

[93] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[94] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[95] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[96] Sparsh Mittal. A survey on optimized implementation of deep learning models on the nvidia jetson platform. *Journal of Systems Architecture*, 97:428–442, 2019.

[97] Michael Mathieu, Mikael Henaff, and Yann LeCun. Fast training of convolutional networks through ffts. *arXiv preprint arXiv:1312.5851*, 2013.

[98] Nicolas Vasilache, Jeff Johnson, Michael Mathieu, Soumith Chintala, Serkan Piantino, and Yann LeCun. Fast convolutional nets with fbfft: A gpu performance evaluation. *arXiv preprint arXiv:1412.7580*, 2014.

[99] Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. In *Advances in Neural Information Processing Systems*, pages 10117–10126, 2018.

[100] Harry Pratt, Bryan Williams, Frans Coenen, and Yalin Zheng. Fcnn: Fourier convolutional neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 786–798. Springer, 2017.

[101] Oren Rippel, Jasper Snoek, and Ryan P Adams. Spectral representations for convolutional neural networks. In *Advances in neural information processing systems*, pages 2449–2457, 2015.

[102] Bochen Guan, Jinnian Zhang, William A Sethares, Richard Kijowski, and Fang Liu. Specnet: Spectral domain convolutional neural network. *arXiv preprint arXiv:1905.10915*, 2019.

[103] Junxuan Li, Shaodi You, and Antonio Robles-Kelly. A frequency domain neural network for fast image super-resolution. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[104] Gordon Wetzstein, Aydogan Ozcan, Sylvain Gigan, Shanhui Fan, Dirk Englund, Marin Soljačić, Cornelia Denz, David AB Miller, and Demetri Psaltis. Inference in artificial intelligence with deep optics and photonics. *Nature*, 588(7836):39–47, 2020.

[105] Joseph W Goodman. *Introduction to Fourier optics*. Roberts and Company Publishers, 2005.

[106] J Feldmann, N Youngblood, M Karpov, H Gehring, X Li, M Stappers, M Le Gallo, X Fu, A Lukashchuk, AS Raja, J. Liu, C. D. Wright, A. Sebastian, T. J. Kippenberg, W. H. P. Pernice, and H. Bhaskaran. Parallel convolutional processing using an integrated photonic tensor core. *Nature*, 589(7840):52–58, 2021.

[107] Yichen Shen, Nicholas C Harris, Scott Skirlo, Mihika Prabhu, Tom Baehr-Jones, Michael Hochberg, Xin Sun, Shijie Zhao, Hugo Larochelle, Dirk Englund, and Marin Soljačić. Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 11(7):441, 2017.

[108] Dirk Englund, Arka Majumdar, Michal Bajcsy, Andrei Faraon, Pierre Petroff, and Jelena Vučković. Ultrafast photon-photon interaction in a strongly coupled quantum dot-cavity system. *Physical review letters*, 108(9):093604, 2012.

[109] Asit Mishra and Debbie Marr. Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy. *arXiv preprint arXiv:1711.05852*, 2017.

[110] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 6356–6364, 2017.

[111] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2604–2613, 2019.

[112] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[113] Demetri Psaltis, David Brady, Xiang-Guang Gu, and Steven Lin. Holography in artificial neural networks. In *Landmark Papers On Photorefractive Nonlinear Optics*, pages 541–546. World Scientific, 1995.

[114] Taiwei Lu, Shudong Wu, Xin Xu, and TS Francis. Two-dimensional programmable optical neural network. *Applied optics*, 28(22):4908–4913, 1989.

[115] Demetri Psaltis, David Brady, and Kelvin Wagner. Adaptive optical networks using photorefractive crystals. *Applied Optics*, 27(9):1752–1759, 1988.

[116] Maike Prossotowicz, Andreas Heimes, Daniel Flamm, Florian Jansen, Hans-Jürgen Otto, Aleksander Budnicki, Alexander Killi, and Uwe Morgner. Coherent beam combining with micro-lens arrays. *Optics Letters*, 45(24):6728–6731, 2020.

[117] Ihsan Fsaifes, Louis Daniault, Severine Bellanger, Matthieu Veinhard, Jerome Bourderionnet, Christian Larat, Eric Lallier, Eric Durand, Arnaud Brignon, and Jean-Christophe Chanteloup. Coherent beam combining of 61 femtosecond fiber amplifiers. *Optics Express*, 28(14):20152–20161, 2020.

[118] Tim D Gerke and Rafael Piestun. Aperiodic volume optics. *Nature photonics*, 4(3):188–193, 2010.

[119] Alan Zhan, Shane Colburn, Christopher M Dodson, and Arka Majumdar. Metasurface freeform nanophotonics. *Scientific reports*, 7(1):1–9, 2017.

[120] Kaggle dogs vs. cats redux: Kernels edition. https://www.kaggle.com/c/dogs-vs-cats, 2016.

[121] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. Technical report.

[122] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[123] X. Li, G. Zhang, H. H. Huang, Z. Wang, and W. Zheng. Performance analysis of gpu-based convolutional neural networks. *45th ICPP*, pages 67–76, 2016.

[124] J. Chang, V. Sitzmann, X. Dun, W. Heidrich, and G. Wetzstein. Hybrid optical-electronic convolutional neural networks with optimized diffractive optics for image classification. *Scientific Reports*, 8:12324, 2018.

[125] Alexander Montes McNeil, Yuxiao Li, Allen Zhang, Michael Moebius, and Yongmin Liu. Fundamentals and recent developments of free-space optical neural networks. *Journal of Applied Physics*, 136(3), 2024.

[126] Haijia Chen, Shaozhen Lou, Quan Wang, Peifeng Huang, Huigao Duan, and Yueqiang Hu. Diffractive deep neural networks: Theories, optimization, and applications. *Applied Physics Reviews*, 11(2), 2024.

[127] L. Cutrona, E. Leith, C. Palermo, and L. Porcello. Optical data processing and filtering systems. *IRE Trans. Inf. Theory*, 6(3):386–400, 1960.

[128] J. N. Mait, G. W. Euliss, and R. A. Athale. Computational imaging. *Adv. Opt. Photon.*, 10:409–483, 2018.

[129] Z. Hu, S. Li, R. L. T. Schwartz, M. Solyanik-Gorgone, M. Misculglio, P. Gupta, and V. J. Sorger. High-throughput miltichannel parallelized diffraction convolutional neural network accelerator. *Laser Photonics Rev.*, 16:2200213, 2022.

[130] K. Wei, X. Li, J. Fröch, P. Chakravarthula, J. Whitehead, E. Tseng, A. Majumdar, and F. Heide. Spatially varying nanophotonic neural networks. *Arxiv*, page 2308.03407, 2023.

[131] Xuhao Luo, Yueqiang Hu, Xiangnian Ou, Xin Li, Jiajie Lai, Na Liu, Xinbin Cheng, Anlian Pan, and Huigao Duan. Metasurface-enabled on-chip multiplexed diffractive neural networks in the visible. *Light: Science & Applications*, 11(158), 2022.

[132] Geonseok Seo, Jaeyoung Yoo, Jaeseok Cho, and Nojun Kwak. Kl-divergence-based region proposal network for object detection. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 2001–2005. IEEE, 2020.

[133] V. Liu and S. Fan. S4: A free electromagnetic solver for layered periodic structures. *Computer Physics Communications.*, 183:2233–2244, 2012.

[134] Şaban Öztürk, Umut Özkaya, Bayram Akdemir, and Levent Seyfi. Convolution kernel size effect on convolutional neural network in histopathological image processing applications. In *2018 International Symposium on Fundamentals of Electrical Engineering (ISFEE)*, pages 1–5. IEEE, 2018.

[135] Han Cai, Chuang Gan, Ji Lin, and Song Han. Network augmentation for tiny deep learning. *arXiv preprint arXiv:2110.08890*, 2021.

[136] J. W. Goodman. *Introduction to Fourier Optics*. Robers & Company Publishers, Englewood, Colorado, 2005.

[137] Kyoji Matsushima and Tomoyoshi Shimobaba. Band-limited angular spectrum method for numerical simulation of free-space propagation in far and near fields. *Optics express*, 17(22):19662–19673, 2009.

[138] Ji Ma and Yuyu Yuan. Dimension reduction of image deep feature using pca. *Journal of Visual Communication and Image Representation*, 63:102578, 2019.

[139] Gordana Ivosev, Lyle Burton, and Ron Bonner. Dimensionality reduction and visualization in principal component analysis. *Analytical chemistry*, 80(13):4933–4944, 2008.

[140] R. W. Gerchberg. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

[141] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[142] Bernhard Graimann, Brendan Z Allison, and Gert Pfurtscheller. *Brain-computer interfaces: Revolutionizing human-computer interaction*. Springer Science & Business Media, 2010.

[143] Andrea De Cesarei, Geoffrey R Loftus, Serena Mastria, and Maurizio Codispoti. Understanding natural scenes: Contributions of image statistics. *Neuroscience & Biobehavioral Reviews*, 74:44–57, 2017.

[144] M Florencia Iacaruso, Ioana T Gasler, and Sonja B Hofer. Synaptic organization of visual space in primary visual cortex. *Nature*, 547(7664):449–452, 2017.

[145] Jonathan J Nassi and Edward M Callaway. Parallel processing strategies of the primate visual system. *Nature reviews neuroscience*, 10(5):360–372, 2009.

[146] Zheyu Yang, Taoyi Wang, Yihan Lin, Yuguo Chen, Hui Zeng, Jing Pei, Jiazheng Wang, Xue Liu, Yichun Zhou, Jianqiang Zhang, et al. A vision chip with complementary pathways for open-world sensing. *Nature*, 629(8014):1027–1033, 2024.

[147] Daniel Gehrig and Davide Scaramuzza. Low-latency automotive vision with event cameras. *Nature*, 629(8014):1034–1040, 2024.

[148] Alexander E Siemenn, Eunice Aissi, Fang Sheng, Armi Tiihonen, Hamide Kavak, Basita Das, and Tonio Buonassisi. Using scalable computer vision to automate high-throughput semiconductor characterization. *Nature Communications*, 15(1):4654, 2024.

[149] Yuekun Yang, Chen Pan, Yixiang Li, Xingjian Yangdong, Pengfei Wang, Zhu-An Li, Shuang Wang, Wentao Yu, Guanyu Liu, Bin Cheng, et al. In-sensor dynamic computing for intelligent machine vision. *Nature Electronics*, 7(3):225–233, 2024.

[150] Amy S Woodget, Robbie Austrums, Ian P Maddock, and Evelyn Habit. Drones and digital photogrammetry: from classifications to continuums for monitoring river habitat and hydromorphology. *Wiley Interdisciplinary Reviews: Water*, 4(4):e1222, 2017.

[151] Tao Zhang. Toward automated vehicle teleoperation: Vision, opportunities, and challenges. *IEEE Internet of Things Journal*, 7(12), 2020.

[152] Yingzhang Wu, Wenbo Li, Jie Zhang, Bangbei Tang, Jinlin Xiang, Shen Li, and Gang Guo. Driver's hand-foot coordination and global-regional brain functional connectivity under fatigue: Via graph theory and explainable artificial intelligence. *IEEE Transactions on Intelligent Vehicles*, 9(2):3493–3508, 2023.

[153] Yitong Chen, Tiankuang Zhou, Jiamin Wu, Hui Qiao, Xing Lin, Lu Fang, and Qionghai Dai. Photonic unsupervised learning variational autoencoder for high-throughput and low-latency image transmission. *Science Advances*, 9(7):eadf8437, 2023.

[154] Liane Bernstein, Alexander Sludds, Christopher Panuski, Sivan Trajtenberg-Mills, Ryan Hamerly, and Dirk Englund. Single-shot optical neural network. *Science Advances*, 9(25):eadg7904, 2023.

[155] Fei Xia, Kyungduk Kim, Yaniv Eliezer, SeungYun Han, Liam Shaughnessy, Sylvain Gigan, and Hui Cao. Nonlinear optical encoding enabled by recurrent linear scattering. *Nature Photonics*, pages 1–9, 2024.

[156] Bijie Bai, Yuhang Li, Yi Luo, Xurong Li, Ege Çetintaş, Mona Jarrahi, and Aydogan Ozcan. All-optical image classification through unknown random diffusers using a single-pixel diffractive network. *Light: Science & Applications*, 12(1):69, 2023.

[157] Yuzhu Ji, Haijun Zhang, Zhao Zhang, and Ming Liu. Cnn-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances. *Information Sciences*, 546:835–857, 2021.

[158] Jianchao Yang, Kai Yu, and Thomas Huang. Supervised translation-invariant sparse coding. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3517–3524. IEEE, 2010.

[159] Krizhevsky Alex. Learning multiple layers of features from tiny images. *https://www. cs. toronto. edu/kriz/learning-features-2009-TR. pdf*, 2009.

[160] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[161] Maxwell G Anderson, Shi-Yuan Ma, Tianyu Wang, Logan G Wright, and Peter L McMahon. Optical transformers. *arXiv preprint arXiv:2302.10360*, 2023.

[162] Feichi Zhou and Yang Chai. Near-sensor and in-sensor computing. *Nature Electronics*, 3(11):664–671, 2020.

[163] Virat Tara, Rui Chen, Johannes E Fröch, Zhuoran Fang, Jie Fang, Romil Audhkhasi, Minho Choi, and Arka Majumdar. Non-volatile reconfigurable transmissive notch filter using wide bandgap phase change material antimony sulfide. *IEEE Journal of Selected Topics in Quantum Electronics*, 2024.

[164] Dohyun Kang, Hyeonsu Heo, Younghwan Yang, Junhwa Seong, Hongyoon Kim, Joohoon Kim, and Junsuk Rho. Liquid crystal-integrated metasurfaces for an active photonic platform. *Opto-Electronic Advances*, pages 230216–1, 2024.

[165] Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. Enhanced convolutional neural tangent kernels. *arXiv preprint arXiv:1911.00809*, 2019.

[166] Nikolaos Tsilivis and Julia Kempe. What can the neural tangent kernel tell us about adversarial robustness? *arXiv preprint arXiv:2210.05577*, 2022.

[167] Dan Zhang, Fangfang Zhou, Yuwen Jiang, and Zhengming Fu. Mm-bsn: Self-supervised image denoising for real-world with multi-mask based on blind-spot network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4188–4197, 2023.

[168] Yongsheng Mei, Tian Lan, and Guru Venkataramani. Exploiting partial common information microstructure for multi-modal brain tumor segmentation. *arXiv preprint arXiv:2302.02521*, 2023.

[169] Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33:9900–9911, 2020.

[170] Grégoire Petit, Adrian Popescu, Eden Belouadah, David Picard, and Bertrand Delezoide. Plastil: Plastic and stable memory-free class-incremental learning. *arXiv preprint arXiv:2209.06606*, 2022.

[171] Shixiang Tang, Dapeng Chen, Jinguo Zhu, Shijie Yu, and Wanli Ouyang. Layerwise optimization by gradient decomposition for continual learning. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 9634–9643, 2021.

[172] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13588–13597, 2020.

[173] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 844–853, 2021.

[174] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019.

[175] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. *Advances in Neural Information Processing Systems*, 34:3478–3490, 2021.

[176] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 12245–12254, 2020.

[177] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[178] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[179] Runqin Xu, Pin Lv, Fanjiang Xu, and Yishi Shi. A survey of approaches for implementing optical neural networks. *Optics & Laser Technology*, 136:106787, 2021.

[180] Guibin Zhao, Pengfei Li, Zhibo Zhang, Fusen Guo, Xueting Huang, Wei Xu, Jinyin Wang, and Jianlong Chen. Towards sar automatic target recognition: Multi-category sar image classification based on light weight vision transformer. In *2024 21st Annual International Conference on Privacy, Security and Trust (PST)*, pages 1–6. IEEE, 2024.

[181] Tingzhao Fu, Jianfa Zhang, Run Sun, Yuyao Huang, Wei Xu, Sigang Yang, Zhihong Zhu, and Hongwei Chen. Optical neural networks: progress and challenges. *Light: Science & Applications*, 13(1):263, 2024.

[182] Yanbing Liu, Shaochong Liu, Tao Li, Tianyu Li, Wei Li, Guoqing Wang, Xun Liu, Wei Yang, and Yuan'an Liu. Towards constructing a doe-based practical optical neural system for ship recognition in remote sensing images. *Signal Processing*, 221:109488, 2024.

[183] Adith Boloor, Weikai Lin, Tianrui Ma, Yu Feng, Yuhao Zhu, and Xuan Zhang. Private-eye: In-sensor privacy preservation through optical feature separation. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2357–2367. IEEE, 2025.

[184] Baiheng Zhao, Junwei Cheng, Bo Wu, Dingshan Gao, Hailong Zhou, and Jianji Dong. Integrated photonic convolution acceleration core for wearable devices. *Opto-Electronic Science*, 2(12):230017–1, 2023.

[185] Zhiyuan Lu, Huan Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In *NeurIPS*, 2017.

[186] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and Computational Harmonic Analysis*, 48(2):787–794, 2020.

[187] Changshuo Bao, Qianxiao Li, and Haizhao Yang. Approximation analysis of convolutional neural networks from a feature extraction view. *Applied and Computational Harmonic Analysis*, 54:47–84, 2021.

[188] Mariia Seleznova, Dana Weitzner, Raja Giryes, Gitta Kutyniok, and Hung-Hsu Chou. Neural (tangent kernel) collapse. *Advances in Neural Information Processing Systems*, 36:16240–16270, 2023.

[189] Hrayr Harutyunyan, Ankit Singh Rawat, Aditya Krishna Menon, Seungyeon Kim, and Sanjiv Kumar. Supervision complexity and its role in knowledge distillation. *arXiv preprint arXiv:2301.12245*, 2023.

[190] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[191] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[192] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ArXiv*, abs/1905.11946, 2019.

[193] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[194] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4293–4302, 2016.

[195] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[196] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[197] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[198] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

[199] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, 2018.

[200] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *arXiv preprint arXiv:2010.15277*, 2020.

[201] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[202] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[203] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *European Conference on Computer Vision*, pages 699–715. Springer, 2020.

[204] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019.

[205] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

[206] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017.

[207] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11321–11329, 2019.

[208] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128, 2009.

[209] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[210] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[211] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[212] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[213] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *arXiv preprint arXiv:2012.02780*, 2020.

[214] Yen-Chang Hsu, Yen-Cheng Liu, Anita Ramasamy, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

[215] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2544–2553, 2021.

[216] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[217] Davide Maltoni and Vincenzo Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, 2019.

[218] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020.

[219] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 583–592, 2019.

[220] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12183–12192, 2020.

[221] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217, 2020.

[222] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135:38–54, 2021.

[223] Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A Alemi, Jascha Sohl-Dickstein, and Samuel S Schoenholz. Neural tangents: Fast and easy infinite neural networks in python. *arXiv preprint arXiv:1912.02803*, 2019.

[224] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[225] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[226] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[227] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

[228] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. *arXiv preprint arXiv:2203.11473*, 2022.

[229] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021.

[230] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*. Granada, 2011.

[231] Kwang-Ting Cheng and Yi-Chu Wang. Using mobile gpu for general-purpose computing–a case study of face recognition on smartphones. In *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test*, pages 1–4. IEEE, 2011.

[232] Xiaomeng Xin, Yiran Zhong, Yunzhong Hou, Jinjun Wang, and Liang Zheng. Memory-free generative replay for class-incremental learning. *arXiv preprint arXiv:2109.00328*, 2021.

[233] Quang Pham, Chenghao Liu, and Steven Hoi. Continual normalization: Rethinking batch normalization for online continual learning. *arXiv preprint arXiv:2203.16102*, 2022.

[234] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

[235] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on class incremental blurry task configuration with anytime inference. *arXiv preprint arXiv:2110.10031*, 2021.

[236] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020.

[237] Chris Dongjoo Kim, Jinseo Jeong, and Gunhee Kim. Imbalanced continual learning with partitioning reservoir sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 411–428. Springer, 2020.

[238] Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *International Conference on Machine Learning*, pages 1952–1961. PMLR, 2020.

[239] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[240] Wei Liu, Karoll Quijano, and Melba M Crawford. Yolov5-tassel: detecting tassels in rgb uav imagery with improved yolov5 based on transfer learning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:8085–8094, 2022.

[241] Xiaoling Luo, Xiaobo Ma, Matthew Munden, Yao-Jan Wu, and Yangsheng Jiang. A multisource data approach for estimating vehicle queue length at metered on-ramps. *Journal of Transportation Engineering, Part A: Systems*, 148(2):04021117, 2022.

[242] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456, 2021.

[243] Kengo Nakata, Youyang Ng, Daisuke Miyashita, Asuka Maki, Yu-Chieh Lin, and Jun Deguchi. Revisiting a knn-based image classification system with high-capacity storage. In *European Conference on Computer Vision*, pages 457–474. Springer, 2022.

[244] Pedro Morgado and Nuno Vasconcelos. Nettailor: Tuning the architecture, not just the weights. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3044–3054, 2019.

[245] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[246] Yunhui Guo, Yandong Li, Liqiang Wang, and Tajana Rosing. Depthwise convolution is all you need for learning multiple visual domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8368–8375, 2019.

[247] Fangfang Zhou, Zhengming Fu, and Dan Zhang. High dynamic range imaging with context-aware transformer. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2023.

[248] Thang Doan, Mehdi Abbana Bennani, Bogdan Mazoure, Guillaume Rabusseau, and Pierre Alquier. A theoretical analysis of catastrophic forgetting through the ntk overlap matrix. In *International Conference on Artificial Intelligence and Statistics*, pages 1072–1080. PMLR, 2021.

[249] Ryo Karakida and Shotaro Akaho. Learning curves for continual learning in neural networks: Self-knowledge transfer and forgetting. *arXiv preprint arXiv:2112.01653*, 2021.

[250] Benedikt Pfülb and Alexander Gepperth. A comprehensive, application-oriented study of catastrophic forgetting in dnns. *arXiv preprint arXiv:1905.08101*, 2019.

[251] Tyler Highlander and Andres Rodriguez. Very efficient training of convolutional neural networks using fast fourier transform and overlap-and-add. *arXiv preprint arXiv:1601.06815*, 2016.

[252] Xiaoyong Shen, Xin Tao, Hongyun Gao, Chao Zhou, and Jiaya Jia. Deep automatic portrait matting. In *European conference on computer vision*, pages 92–107. Springer, 2016.

[253] Jae-Han Lee, Minhyeok Heo, Kyung-Rae Kim, and Chang-Su Kim. Single-image depth estimation based on fourier domain analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 330–339, 2018.

[254] Armin Kappeler, Sushobhan Ghosh, Jason Holloway, Oliver Cossairt, and Aggelos Katsaggelos. Ptychnet: Cnn based fourier ptychography. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1712–1716. IEEE, 2017.

[255] James Tan, Zengguang Cheng, Xuan Li, Nathan Youngblood, Utku E Ali, C David Wright, Wolfram HP Pernice, and Harish Bhaskaran. Monadic pavlovian associative learning in a backpropagation-free photonic network. *arXiv preprint arXiv:2011.14709*, 2020.

[256] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[257] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012.

[258] Carvana image masking challenge. https://www.kaggle.com/c/carvana-image-masking-challenge, 2017.

[259] L. G. Wright, T. Onodera, M. M. Stein, T. Wang, D. T. Schachter, Z. Hu, and P. L. McMahon. Deep physical neural networks trained with backpropagation. *Nature*, 601:549–555, 2022.

[260] F. Ashtiani, A. J. Geers, and F. Aflatouni. An on-chip photonic deep neural network for image classification. *Nature.*, 606:501–506, 2022.

[261] K. Liao, T. Dai, Q. Yan, X. Hu, and Q. Gong. Integrated photonic neural networks: Opportunities and challenges. *ACS Photonics.*, 10(7):2001–2010, 2023.

[262] Xiangyan Meng, Guojie Zhang, Nuannuan Shi, Guangyi Li, José Azaña, José Capmany, Jianping Yao, Yichen Shen, Wei Li, Ninghua Zhu, et al. Compact optical convolution processing unit based on multimode interference. *Nature Communications*, 14(1):3000, 2023.

[263] S. B. Damelin and W. Miller. *The Mathematics of Signal Processing.* Cambridge University Press, Cambridge, UK, 2011.

[264] Praise. https://github.com/brandondube/praise/tree/master, 2019.

[265] Phase retrieval ii: Iterative transform algorithms. https://www.retrorefractions.com/blog/phase-retrieval-02/, 2023.

[266] Jiho Chang, Yoonsung Choi, Taegyoung Lee, and Junhee Cho. Reducing mac operation in convolutional neural network with sign prediction. In *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 177–182. IEEE, 2018.

[267] Quan Liu, Brandon T Swartz, Ivan Kravchenko, Jason G Valentine, and Yuankai Huo. Extrememeta: High-speed lightweight image segmentation model by remodeling multi-channel metamaterial imagers. *arXiv preprint arXiv:2405.17568*, 2024.

[268] Qixiang Cheng, Madeleine Glick, and Keren Bergman. Optical interconnection networks for high-performance systems. In *Optical fiber telecommunications VII*, pages 785–825. Elsevier, 2020.

[269] Edward L Ince. *Ordinary differential equations.* Courier Corporation, 1956.

[270] Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Society, 2022.

[271] Lucien Le Cam. The central limit theorem around 1935. *Statistical science*, pages 78–91, 1986.

[272] William J Vetter. Matrix calculus operations and taylor expansions. *SIAM review*, 15(2):352–369, 1973.

[273] David JC MacKay et al. Introduction to gaussian processes. *NATO ASI series F computer and systems sciences*, 168:133–166, 1998.

[274] Christopher Williams. Computing with infinite networks. *Advances in neural information processing systems*, 9, 1996.

[275] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*, pages 2285–2294, 2015.

[276] Shiyu Liang, Ruoyu Sun, and R. Srikant. Revisiting landscape analysis in deep neural networks: Eliminating decreasing paths to infinity, 2019.

[277] Tian Ding, Dawei Li, and Ruoyu Sun. Suboptimal local minima exist for wide neural networks with smooth activations. *Mathematics of Operations Research*, 47(4):2784–2814, 2022.

[278] Luca Venturi, Afonso Bandeira, and Joan Bruna. Spurious valleys in two-layer neural network optimization landscapes. *arXiv preprint arXiv:1802.06384*, 2018.

[279] Dawei Li, Tian Ding, and Ruoyu Sun. On the benefit of width for neural networks: Disappearance of basins. *SIAM Journal on Optimization*, 32(3):1728–1758, 2022.

[280] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2019.

[281] Dachao Lin, Ruoyu Sun, and Zhihua Zhang. Faster directional convergence of linear neural networks under spherically symmetric data. *Advances in Neural Information Processing Systems*, 34, 2021.

[282] Ruoyu Sun, Dawei Li, Shiyu Liang, Tian Ding, and Rayadurgam Srikant. The global landscape of neural networks: An overview. *IEEE Signal Processing Magazine*, 37(5):95–108, 2020.

[283] Adityanarayanan Radhakrishnan, Max Ruiz Luyten, Neha Prasad, and Caroline Uhler. Transfer learning with kernel methods. *Nature Communications*, 14(1):5570, 2023.

[284] Alistair S Dunham, Pedro Beltrao, and Mohammed AlQuraishi. High-throughput deep learning variant effect prediction with sequence unet. *Genome Biology*, 24(1):110, 2023.

[285] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pages 3–11. Springer, 2018.

[286] Guangda Ji and Zhanxing Zhu. Knowledge distillation in wide neural networks: Risk bound, data efficiency and imperfect teacher. *Advances in Neural Information Processing Systems*, 33:20823–20833, 2020.

[287] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Finite versus infinite neural networks: An empirical study. In *Advances in Neural Information Processing Systems*, volume 33, pages 15156–15172, 2020.

[288] Yoann Morello, Emilie Grégoire, and Sam Verboven. Exploring task affinities through ntk alignment and early training dynamics in multi-task learning. In *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*, 2024.

[289] Jinlin Xiang and Eli Shlizerman. Tkil: tangent kernel approach for class balanced incremental learning. *arXiv preprint arXiv:2206.08492*, 2022.

[290] Chenyu You, Jinlin Xiang, Kun Su, Xiaoran Zhang, Siyuan Dong, John Onofrey, Lawrence Staib, and James S Duncan. Incremental learning meets transfer learning: Application to multi-site prostate mri segmentation. In *International Workshop on Distributed, Collaborative, and Federated Learning*, pages 3–16. Springer, 2022.

[291] Ruisu Zhang, Wei Zhao, Bozhao Qi, Etienne Pochet, Yongjian Yang, Jinlin Xiang, Roger Trullo, Ohn Chow, Jimena Diaz De Leon, and Qi Tang. Advanced transformer-based system to localize and predict atopic dermatitis. In *2024 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI)*, pages 1–8. IEEE, 2024.

[292] TensorFlow Developers. Tensorflow. *Zenodo*, 2022.

[293] Yingzhou Lu, Chiung-Ting Wu, Sarah J Parker, Zuolin Cheng, Georgia Saylor, Jennifer E Van Eyk, Guoqiang Yu, Robert Clarke, David M Herrington, and Yue Wang. Cot: an efficient and accurate method for detecting marker genes among many subtypes. *Bioinformatics Advances*, 2(1):vbac037, 2022.

[294] Yi Fu, Yingzhou Lu, Yizhi Wang, Bai Zhang, Zhen Zhang, Guoqiang Yu, Chunyu Liu, Robert Clarke, David M Herrington, and Yue Wang. Ddn3. 0: Determining significant rewiring of biological network structure with differential dependency networks. *Bioinformatics*, 40(6):btae376, 2024.

[295] Baoqi Zhao, Xiong Yang, Hoileong Lee, and Bowen Dong. An improved educational competition optimizer with multi-covariance learning operators for global optimization problems. *Cluster Computing*, 28(15):964, 2025.

Appendix A

APPENDIX

A.1 Connection with Gaussian Processes

We provide a detailed proof, using NTK, demonstrating how it guarantees that infinite-width networks converge to a global minimum when trained to minimize empirical loss. Let us consider a fully-connected neural networks with parameter $\theta, f(.;\theta) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$. Layers are indexed from 0 (input) to L (output), each containing $n_0, \ldots, n_L$ neurons, including the input of size n_0 and the output of size n_L. There are $P = \sum_{l=0}^{L-1} (n_l + 1)\, n_{l+1}$ parameters in total and thus we have $\theta \in \mathbb{R}^P$. The training dataset contains N data points, $\mathcal{D} = \left\{\boldsymbol{x}^{(i)}, y^{(i)}\right\}_{i=1}^{N}$. All the inputs are denoted as $\mathcal{X} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$ and all the labels are denoted as $\mathcal{Y} = \left\{y^{(i)}\right\}_{i=1}^{N}$.

For $l = 0, \ldots, L-1$, each layer l defines an affine transformation $A^{(l)}$ with a weight matrix $\mathbf{w}^{(l)} \in \mathbb{R}^{n_l \times n_{l+1}}$ and a bias term $\mathbf{b}^{(l)} \in \mathbb{R}^{n_{l+1}}$, as well as a pointwise nonlinearity function $\sigma(.)$, which is Lipschitz continuous. We show the pre-activations and post-activations in detail below

$$A^{(0)} = \boldsymbol{x}, \tag{A.1}$$

$$\tilde{A}^{(l+1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_l}} \mathbf{w}^{(l)^\top} A^{(l)} + \beta \mathbf{b}^{(l)} \quad \in \mathbb{R}^{n_{l+1}}, \tag{A.2}$$

$$A^{(l+1)}(\boldsymbol{x}) = \sigma\left(\tilde{A}^{(l+1)}(\boldsymbol{x})\right) \quad \in \mathbb{R}^{n_{l+1}}. \tag{A.3}$$

Note that the NTK parameterization applies a rescale weight $1/\sqrt{n_l}$ on the transformation to avoid divergence with infinite-width networks. The constant scalar $\beta \geq 0$ controls how much effort the bias terms have.

Deep neural networks have a deep connection with Gaussian processes [274]. The output function of a L-layer network, $f_i(\boldsymbol{x};\theta)$ for $i = 1, \ldots, n_L$, are i.i.d. centered Gaussian process of covariance $\Sigma^{(L)}$, defined recursively as

$$\Sigma^{(1)}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{n_0}\boldsymbol{x}^\top \boldsymbol{x}' + \beta^2, \tag{A.4}$$

$$\lambda^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') = \begin{bmatrix} \Sigma^{(l)}(\boldsymbol{x}, \boldsymbol{x}) & \Sigma^{(l)}(\boldsymbol{x}, \boldsymbol{x}'), \\ \Sigma^{(l)}(\boldsymbol{x}', \boldsymbol{x}) & \Sigma^{(l)}(\boldsymbol{x}', \boldsymbol{x}') \end{bmatrix}, \tag{A.5}$$

$$\Sigma^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{f\sim\mathcal{N}\left(0,\lambda^{(l)}\right)}\left[\sigma(f(\boldsymbol{x}))\sigma\left(f\left(\boldsymbol{x}'\right)\right)\right] + \beta^2. \tag{A.6}$$

Proof of Case 1: Let's start with $L = 1$, when there is no nonlinearity function and the input is only processed by a simple affine transformation

$$f(\boldsymbol{x}; \theta) = \tilde{A}^{(1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_0}}\mathbf{w}^{(0)^\top}\boldsymbol{x} + \beta\mathbf{b}^{(0)}, \tag{A.7}$$

$$\text{where } \tilde{A}_m^{(1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_0}}\sum_{i=1}^{n_0} w_{im}^{(0)} x_i + \beta b_m^{(0)} \quad \text{for } 1 \le m \le n_1. \tag{A.8}$$

Since the weights and biases are initialized i.i.d., all the output dimensions of this network $\tilde{A}_1^{(1)}(\boldsymbol{x}), \ldots, \tilde{A}_{n_1}^{(1)}(\boldsymbol{x})$ are also i.i.d. Given different inputs, the m-th network outputs $\tilde{A}_m^{(1)}(.)$ have a joint multivariate Gaussian distribution, equivalent to a Gaussian process with covariance function. We know that mean $\mu_w = \mu_b = 0$ and variance $\sigma_w^2 = \sigma_b^2 = 1$. Then, we can obtain

$$\Sigma^{(1)}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\left[\tilde{A}_m^{(1)}(\boldsymbol{x})\tilde{A}_m^{(1)}\left(\boldsymbol{x}'\right)\right] \tag{A.9}$$

$$= \mathbb{E}\left[\left(\frac{1}{\sqrt{n_0}}\sum_{i=1}^{n_0} w_{i,m}^{(0)} x_i + \beta b_m^{(0)}\right)\left(\frac{1}{\sqrt{n_0}}\sum_{i=1}^{n_0} w_{i,m}^{(0)} x_i' + \beta b_m^{(0)}\right)\right] \tag{A.10}$$

$$= \frac{1}{n_0}\sigma_w^2 \sum_{i=1}^{n_0}\sum_{j=1}^{n_0} x_i x'_j + \frac{\beta\mu_b}{\sqrt{n_0}}\sum_{i=1}^{n_0} w_{im}\left(x_i + x_i'\right) + \sigma_b^2\beta^2 \tag{A.11}$$

$$= \frac{1}{n_0}\boldsymbol{x}^\top\boldsymbol{x}' + \beta^2. \tag{A.12}$$

Proof of Case 2: Using induction, we first assume the proposition is true for $L = l$, a l-layer network, and thus $\tilde{A}_m^{(l)}(.)$ is a Gaussian process with covariance $\Sigma^{(l)}$ and $\left\{\tilde{A}_i^{(l)}\right\}_{i=1}^{n_l}$ are i.i.d.

Then we need to prove the proposition is also true for $L = l + 1$. We compute the outputs by

$$f(\boldsymbol{x};\theta) = \tilde{A}^{(l+1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_l}}\mathbf{w}^{(l)^\top}\sigma\left(\tilde{A}^{(l)}(\boldsymbol{x})\right) + \beta\mathbf{b}^{(l)}, \tag{A.13}$$

$$\text{where } \tilde{A}_m^{(l+1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_l}}\sum_{i=1}^{n_l} w_{im}^{(l)}\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right) + \beta b_m^{(l)} \quad \text{for } 1 \le m \le n_{l+1}. \tag{A.14}$$

We can infer that the expectation of the sum of contributions of the previous hidden layers is zero

$$\mathbb{E}\left[w_{im}^{(l)}\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)\right] = \mathbb{E}\left[w_{im}^{(l)}\right]\mathbb{E}\left[\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)\right] = \mu_w\mathbb{E}\left[\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)\right] = 0, \tag{A.15}$$

$$\mathbb{E}\left[\left(w_{im}^{(l)}\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)\right)^2\right] = \mathbb{E}\left[w_{im}^{(l)}\right]\mathbb{E}\left[\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)^2\right] = \sigma_w^2\Sigma^{(l)}(\boldsymbol{x},\boldsymbol{x}) = \Sigma^{(l)}(\boldsymbol{x},\boldsymbol{x}). \tag{A.16}$$

Since $\left\{\tilde{A}_i^{(l)}(\boldsymbol{x})\right\}_{i=1}^{n_l}$ are i.i.d., according to the central limit theorem, when the hidden layer gets infinitely wide $n_l \to \infty$, $\tilde{A}_m^{(l+1)}(\boldsymbol{x})$ is Gaussian distributed with variance $\beta^2 + \text{Var}\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)$. Note that $\tilde{A}_1^{(l+1)}(\boldsymbol{x}), \ldots, \tilde{A}_{n_{l+1}}^{(l+1)}(\boldsymbol{x})$ are still i.i.d. $\tilde{A}_m^{(l+1)}(.)$ is equivalent to a Gaussian process with covariance function

$$\Sigma^{(l+1)}(\boldsymbol{x},\boldsymbol{x}') = \mathbb{E}\left[\tilde{A}_m^{(l+1)}(\boldsymbol{x})\tilde{A}_m^{(l+1)}(\boldsymbol{x}')\right] \tag{A.17}$$

$$= \frac{1}{n_l}\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x})\right)^\top\sigma\left(\tilde{A}_i^{(l)}(\boldsymbol{x}')\right) + \beta^2 \quad \text{;similar to how we get } \Sigma^{(1)}. \tag{A.18}$$

When $n_l \to \infty$, according to central limit theorem,

$$\Sigma^{(l+1)}(\boldsymbol{x},\boldsymbol{x}') \to \mathbb{E}_{f\sim\mathcal{N}\left(0,\Lambda^{(l)}\right)}\left[\sigma(f(\boldsymbol{x}))^\top\sigma\left(f(\boldsymbol{x}')\right)\right] + \beta^2. \tag{A.19}$$

The form of Gaussian processes in the above process is referred to as the Neural Network Gaussian Process (NNGP) in [89, 86, 88].

A.2 Deterministic NTK

When $n_1, \ldots, n_L \to \infty$ (network with infinite width), the NTK converges to be:(1) deterministic at initialization, meaning that the kernel is irrelevant to the initialization values and

only determined by the model architecture; and (2) stays constant during training.

Proof of Case 1: First of all, we have $K^{(0)} = 0$. When $L = 1$, we can get the representation of NTK directly. It is deterministic and does not depend on the network initialization. There is no hidden layer, so there is nothing to take on infinite width. Thus, we can obtain

$$f(\boldsymbol{x};\theta) = \tilde{A}^{(1)}(\boldsymbol{x}) = \frac{1}{\sqrt{n_0}}\mathbf{w}^{(0)\top}\boldsymbol{x} + \beta\mathbf{b}^{(0)}, \tag{A.20}$$

$$K^{(1)}\left(\boldsymbol{x},\boldsymbol{x}';\theta\right) = \left(\frac{\partial f\left(\boldsymbol{x}';\theta\right)}{\partial \mathbf{w}^{(0)}}\right)^{\top}\frac{\partial f(\boldsymbol{x};\theta)}{\partial \mathbf{w}^{(0)}} + \left(\frac{\partial f\left(\boldsymbol{x}';\theta\right)}{\partial \mathbf{b}^{(0)}}\right)^{\top}\frac{\partial f(\boldsymbol{x};\theta)}{\partial \mathbf{b}^{(0)}} \tag{A.21}$$

$$= \frac{1}{n_0}\boldsymbol{x}^{\top}\boldsymbol{x}' + \beta^2 = \Sigma^{(1)}\left(\boldsymbol{x},\boldsymbol{x}'\right). \tag{A.22}$$

Proof of Case 2: When $L = l$, we assume that a l-layer network with $\tilde{P}$ parameters in total, $\tilde{\theta} = \left(\mathbf{w}^{(0)},\ldots,\mathbf{w}^{(l-1)},\mathbf{b}^{(0)},\ldots,\mathbf{b}^{(l-1)}\right) \in \mathbb{R}^{\tilde{P}}$, has a NTK converging to a deterministic limit when $n_1,\ldots,n_{l-1} \to \infty$. Then, we obtain

$$K^{(l)}\left(\boldsymbol{x},\boldsymbol{x}';\tilde{\theta}\right) = \nabla_{\tilde{\theta}}\tilde{A}^{(l)}(\boldsymbol{x})^{\top}\nabla_{\tilde{\theta}}\tilde{A}^{(l)}\left(\boldsymbol{x}'\right) \to K_{\infty}^{(l)}\left(\boldsymbol{x},\boldsymbol{x}'\right). \tag{A.23}$$

Note that $K_{\infty}^{(l)}$ has no dependency on θ. Next, let's check the case $L = l+1$. Compared to a l-layer network, a $(l+1)$-layer network has additional weight matrix $\mathbf{w}^{(l)}$ and bias $\mathbf{b}^{(l)}$ and thus the total parameters contain $\theta = \left(\tilde{\theta},\mathbf{w}^{(l)},\mathbf{b}^{(l)}\right)$. The output function of this $(l+1)$-layer network is

$$f(\boldsymbol{x};\theta) = \tilde{A}^{(l+1)}(\boldsymbol{x};\theta) = \frac{1}{\sqrt{n_l}}\mathbf{w}^{(l)\top}\sigma\left(\tilde{A}^{(l)}(\boldsymbol{x})\right) + \beta\mathbf{b}^{(l)}. \tag{A.24}$$

And we know its derivative with respect to different sets of parameters. Let denote $\tilde{A}^{(l)} = \tilde{A}^{(l)}(\boldsymbol{x})$ for brevity in the following equation

$$\nabla_{\mathbf{w}^{(l)}} f(\boldsymbol{x};\theta) = \frac{1}{\sqrt{n_l}} \sigma\left(\tilde{A}^{(l)}\right)^\top \quad \in \mathbb{R}^{1\times n_l}, \tag{A.25}$$

$$\nabla_{\mathbf{b}^{(l)}} f(\boldsymbol{x};\theta) = \beta, \tag{A.26}$$

$$\nabla_{\tilde{\theta}} f(\boldsymbol{x};\theta) = \frac{1}{\sqrt{n_l}} \nabla_{\tilde{\theta}} \sigma\left(\tilde{A}^{(l)}\right) \mathbf{w}^{(l)}, \tag{A.27}$$

$$= \frac{1}{\sqrt{n_l}} \begin{bmatrix} \dot{\sigma}\left(\tilde{A}_1^{(l)}\right) \frac{\partial \tilde{A}_1^{(l)}}{\partial \tilde{\theta}_1} & \dots & \dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right) \frac{\partial \tilde{A}_{n_l}^{(l)}}{\partial \tilde{\theta}_1} \\ \vdots & & \\ \dot{\sigma}\left(\tilde{A}_1^{(l)}\right) \frac{\partial \tilde{A}_1^{(l)}}{\partial \tilde{\theta}_{\tilde{P}}} & \dots & \dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right) \frac{\partial \tilde{A}_{n_l}^{(l)}}{\partial \tilde{\theta}_{\tilde{P}}} \end{bmatrix} \mathbf{w}^{(l)} \in \mathbb{R}^{\tilde{P}\times n_{l+1}}. \tag{A.28}$$

where $\dot{\sigma}$ is the derivative of σ and each entry at location $(p,m), 1\le p\le \tilde{P}, 1\le m\le n_{l+1}$ in the matrix $\nabla_{\tilde{\theta}} f(\boldsymbol{x};\theta)$ can be written as

$$\frac{\partial f_m(\boldsymbol{x};\theta)}{\partial \tilde{\theta}_p} = \sum_{i=1}^{n_l} w_{im}^{(l)} \dot{\sigma}\left(\tilde{A}_i^{(l)}\right) \nabla_{\tilde{\theta}_p} \tilde{A}_i^{(l)}. \tag{A.29}$$

The NTK for this $(l+1)$-layer network can be defined accordingly

$$K^{(l+1)}\left(\boldsymbol{x},\boldsymbol{x}';\theta\right) \tag{A.30}$$

$$= \nabla_\theta f(\boldsymbol{x};\theta)^\top \nabla_\theta f(\boldsymbol{x};\theta) \tag{A.31}$$

$$= \nabla_{\mathbf{w}^{(l)}} f(\boldsymbol{x};\theta)^\top \nabla_{\mathbf{w}^{(l)}} f(\boldsymbol{x};\theta) + \nabla_{\mathbf{b}^{(l)}} f(\boldsymbol{x};\theta)^\top \nabla_{\mathbf{b}^{(l)}} f(\boldsymbol{x};\theta) + \nabla_{\tilde{\theta}} f(\boldsymbol{x};\theta)^\top \nabla_{\tilde{\theta}} f(\boldsymbol{x};\theta) \tag{A.32}$$

$$= \frac{1}{n_l}\left[\sigma\left(\tilde{A}^{(l)}\right)\sigma\left(\tilde{A}^{(l)}\right)^\top + \beta^2\right. \tag{A.33}$$

$$\left. + \mathbf{w}^{(l)^\top} \begin{bmatrix} \dot{\sigma}\left(\tilde{A}_1^{(l)}\right)\dot{\sigma}\left(\tilde{A}_1^{(l)}\right) K_{11}^{(l)} & \dots & \dot{\sigma}\left(\tilde{A}_1^{(l)}\right)\dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right) K_{1n_l}^{(l)} \\ \vdots & & \\ \dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right)\dot{\sigma}\left(\tilde{A}_1^{(l)}\right) K_{n_l1}^{(l)} & \dots & \dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right)\dot{\sigma}\left(\tilde{A}_{n_l}^{(l)}\right) K_{n_ln_l}^{(l)} \end{bmatrix} \mathbf{w}^{(l)} \right]. \tag{A.34}$$

where each individual entry at location $(m,n), 1\le m,n\le n_{l+1}$ of the matrix $K^{(l+1)}$ can be written as

$$K_{mn}^{(l+1)} = \frac{1}{n_l}\left[\sigma\left(\tilde{A}_m^{(l)}\right)\sigma\left(\tilde{A}_n^{(l)}\right) + \beta^2 + \sum_{i=1}^{n_l}\sum_{j=1}^{n_l} w_{im}^{(l)} w_{in}^{(l)} \dot{\sigma}\left(\tilde{A}_i^{(l)}\right)\dot{\sigma}\left(\tilde{A}_j^{(l)}\right) K_{ij}^{(l)}\right]. \tag{A.35}$$

When $n_l \to \infty$, the section has the limit

$$\frac{1}{n_l}\sigma\left(\tilde{A}^{(l)}\right)\sigma\left(\tilde{A}^{(l)}\right) + \beta^2 \to \Sigma^{(l+1)}. \tag{A.36}$$

A.3 Noise Robustness in Wide Finite-Width Networks

Let the single-layer neural network be parameterized by θ, and its output for input x be $f(x;\theta)$. The NTK for inputs x and x' is defined as

$$\Theta(x, x') = \nabla_\theta f(x;\theta)^\top \nabla_\theta f(x';\theta), \tag{A.37}$$

where $\nabla_\theta f(x;\theta)$ represents the gradient of the network output with respect to the parameters.

A.3.1 Noise Perturbation to Parameters

Suppose a small noise δ is added to the parameters, resulting in $\theta' = \theta + \delta$. The perturbed NTK is expressed as

$$\Theta_\delta(x, x') = \nabla_{\theta'} f(x;\theta')^\top \nabla_{\theta'} f(x';\theta'). \tag{A.38}$$

Expanding $\nabla_{\theta'} f(x;\theta')$ using a first-order Taylor approximation

$$\nabla_{\theta'} f(x;\theta') \approx \nabla_\theta f(x;\theta) + H(x;\theta)\delta, \tag{A.39}$$

where $H(x;\theta)$ is the Hessian matrix of $f(x;\theta)$ with respect to θ.

Substituting this into $\Theta_\delta(x, x')$

$$\Theta_\delta(x, x') \approx \Theta(x, x') + \delta^\top H(x;\theta)^\top \nabla_\theta f(x';\theta) + \delta^\top H(x';\theta)^\top \nabla_\theta f(x;\theta) \tag{A.40}$$

$$+ \delta^\top H(x;\theta)^\top H(x';\theta)\delta. \tag{A.41}$$

The perturbation $\Delta\Theta(x, x')$ is given by

$$\Delta\Theta(x, x') = \Theta_\delta(x, x') - \Theta(x, x'). \tag{A.42}$$

A.3.2 Bounding the Perturbation

The magnitude of $\Delta\Theta(x, x')$ satisfies

$$\|\Delta\Theta(x, x')\| \approx \|\delta\| \cdot (\|H(x;\theta)\| \cdot \|\nabla_\theta f(x';\theta)\| + \|H(x';\theta)\| \cdot \|\nabla_\theta f(x;\theta)\|) \tag{A.43}$$

$$+ \|\delta\|^2 \cdot \|H(x;\theta)\| \cdot \|H(x';\theta)\|. \tag{A.44}$$

For small perturbations $\|\delta\| \ll 1$, the second-order term is negligible, and we obtain

$$\|\Delta\Theta(x, x')\| \sim O\left(\frac{\|\delta\|}{m}\right), \tag{A.45}$$

which vanishes as $m \to \infty$.