Hardware-software Co-design of Slimmed Optical Neural Networks

Zheng Zhao¹, Derong Liu¹, Meng Li¹, Zhoufeng Ying¹, Lu Zhang², Biying Xu¹, Bei Yu², Ray Chen¹, David Pan¹
The University of Texas at Austin¹
The Chinese University of Hong Kong²
Introduction

- Emergence of dedicated AI accelerators
  - Optical neural network processor
    - Speed-of-light floating point matrix-vector multiplication
    - >100GHz detection rate
    - Ultra-low energy consumption if configured
  - Great number of components, sensitivity to noise
Previous Optical Neural Network (ONN)

- SVD decompose $W = U \Sigma V^*$
- $U$ and $V^*$ are unitary matrices
  - A unitary $A$ satisfies $AA^* = I$
  - Implemented by Mach-Zehnder interferometers array
- $\Sigma$ is a diagonal matrix
  - Diagonal values are non-negative real
  - Implemented by optical attenuators
- $\sigma$ is non-linear activation
  - Implemented by saturable absorber
Implementing Unitary $U$ and $V^*$

- Mach-Zehnder interferometers (MZI) for $U$ and $V^*$
  - A single MZI implements a 2-dim unitary
    $$\text{out} = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix} \cdot \text{in}$$
  - An array of $n(n-1)/2$ MZIs implements an $n$-dim unitary

- Given an $n$-dim unitary, $\phi$’s can be uniquely computed

$$T_{i,j} = \begin{pmatrix} I & 0 & 0 & 0 & 0 \\ 0 & \cos \phi & 0 & \sin \phi & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & -\sin \phi & 0 & \cos \phi & 0 \\ 0 & 0 & 0 & 0 & I \end{pmatrix}$$

$$\text{out} = \prod_{i=n}^{2} \prod_{j=1}^{i-1} T_{i,j} \cdot \text{in}.$$
Layer size measured by # of MZIs = $m(m-1)/2 + n(n-1)/2$

Software training and hardware implementation

- Train $W$ directly in software $\rightarrow$ SVD-decomp to obtain $U$, $\Sigma$, $V^*$
Slimmed Architecture

- $\Sigma$: diagonal network
- $U$: unitary network
- $T$: sparse tree network
- Use less # of MZIs = $n(n-1)/2$
  - 1 unitary matrix to maintain the expressivity
  - An area-efficient tree network to match the dimension

![Diagram](image)
Co-design Overview

- An arbitrary weight $W$ is **not** $TU\Sigma$-decomposable
- Co-design solution: training and implementation are coupled
  - $T$ and $\Sigma$: Train the device parameters, constraints embedded
  - $U$: Add unitary regularization then approximate with true unitary

![Diagram of co-design overview]

Previous Train and Impl.

- Software Training
- Optical Implementation

$W$ decomp

$U^* \Sigma U$
Tree Network

- Tree network ($T$) to match the different dimension
  - Suppose in-dim > out-dim
  - $\alpha$: linear transfer coefficient

\[
\text{out} = \begin{pmatrix}
\alpha_{1,1} & \alpha_{1,2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \alpha_{2,1} & \alpha_{2,2} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3}
\end{pmatrix} \cdot \text{in}.
\]

\[
y = \sum_{i=1}^{N} \alpha_i \cdot x_i
\]
Tree Network Implementation

- Implemented with MZIs or directional couplers
- A 2 x 1 subtree

\[ y = \alpha_1 \cdot x_1 + \alpha_2 \cdot x_2 \]

A 2 x 1 subtree can be implemented with a single-out MZI.

\[ \alpha_1 = \cos \phi \text{ and } \alpha_2 = \sin \phi \]

\[ \alpha_1^2 + \alpha_2^2 = 1 \quad (\text{energy conservation}) \]
Any $N$-input subtree with arbitrary $\alpha$’s satisfying energy conservation

$$\sum_{i=1}^{N} \alpha_i^2 = 1, -1 \leq \alpha_i \leq 1, i = 1, \ldots, N$$

can be implemented by cascading $(N-1)$ single-out MZIs.

Energy conservation embedded in training.
For unitary network $U$ satisfying $UU^* = I$, add the regularization

$$\text{reg} = \|UU^* - I\|_F$$

Training loss function

$$\text{Loss} = \text{Data Loss} + \text{Regularization Loss}$$

leading to a near-implementable ONN with high accuracy

Trained $U_t \sim$ unitary but only true unitary is implementable by MZIs
Unitary Network in Implementation

- Approximate $U_t$ by a true unitary $U_a$
- SVD-decompose $U_t = PSQ^* \rightarrow U_a = PQ^*$

- **Claim.** Minimize the regularization $\Leftrightarrow$ find the best approximation

$$\text{Min. } \text{reg} \Leftrightarrow \text{Min. } \| U_t - U_a \|_F$$
## Simulation Results

- Implemented in TensorFlow for various ONN setup

<table>
<thead>
<tr>
<th>N1: (14 × 14)-100-10</th>
<th>N4: (14 × 14)-150-150-10</th>
<th>N7: (14 × 14)-150-150-150-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>N2: (14 × 14)-150-10</td>
<td>N5: (28 × 28)-400-400-10</td>
<td>N8: (28 × 28)-400-400-200-10</td>
</tr>
<tr>
<td>N3: (28 × 28)-400-10</td>
<td>N6: (28 × 28)-600-300-10</td>
<td>N9: (28 × 28)-600-600-300-10</td>
</tr>
</tbody>
</table>

- Tested it on Intel Core i9-7900X CPU and an NVIDIA TitanXp GPU

- Performed on the handwritten digit dataset MNIST
Simulation Results

- N1~N9: network configurations
- Our architecture uses 15%-38% less MZIs
- Similar accuracy (~0 accuracy loss)
- Maximum loss is 0.0088
- Average is 0.0058
Noise Robustness

- Better resilience due to less cascaded components
Converged in 300 epochs
Balance of the accuracy and the unitary approximation
Contributions of This Work

♦ An new architecture for ONN
  › Area-efficiency
  › ~0 accuracy loss
  › Better robustness to noise

♦ Hardware and software co-design methodology
  › Software-embedded hardware parameters
  › Hardware constraints guaranteed by software
Future Work

- Better MZI pruning methods
  - ~0 phase MZI \rightarrow \text{pruned} + accuracy recover
  - MZI-sparse unitary matrix

- Design for robustness
  - Adjust noise distribution in training

- Online training

- ONN for other neural network architectures
  - CNN, RNN, etc.
Thanks

Q&A