

LightSim: A Flexible, Fast Modeling Framework for Holistic Photonic Computing Systems

YUXUAN LIN, The Hong Kong University of Science and Technology (Guangzhou), China
GUOHAO DAI, The Hong Kong University of Science and Technology (Guangzhou), China
SHANSHI HUANG, The Hong Kong University of Science and Technology (Guangzhou), China
YUZHMA, The Hong Kong University of Science and Technology (Guangzhou), China

Photonic computing presents a promising alternative for accelerating neural network (NN) computation. However, the effective evaluation and design of these next-generation systems necessitate a thorough understanding of the complex interplay between their electronic and photonic components. In this work, we propose *LightSim*, a holistic and flexible modeling framework aimed at guiding the design and optimization of photonic computing systems. *LightSim* features a hierarchical and modular structure that provides users with a customizable and user-friendly interface. Its photonic-specific models offer in-depth insights into critical aspects like dataflow, mapping, and system performance. Furthermore, its fast modeling capability facilitates rapid exploration of the design space. *LightSim* supports co-design and fair architectural comparisons across all levels, thereby fostering the agile, iterative design process crucial for the development of high-performance and efficient photonic computing hardware. Validation across both incoherent and coherent photonic computing designs demonstrates a high correlation between our results and the reference data.

Additional Key Words and Phrases: Photonic design automation

ACM Reference Format:

Yuxuan Lin, Guohao Dai, Shanshi Huang, and Yuzhe Ma. 2026. LightSim: A Flexible, Fast Modeling Framework for Holistic Photonic Computing Systems. *ACM Trans. Des. Autom. Electron. Syst.* 1, 1 (January 2026), 26 pages. <https://doi.org/10.1145/3818624>

1 Introduction

In recent years, the rapid advancement of artificial intelligence (AI) has imposed significant demands on computational infrastructure. Traditional electronic accelerators, such as Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs), while highly powerful, are increasingly encountering limitations in terms of speed, power consumption, and thermal management. Photonic computing is emerging as a promising alternative, offering transformative potential for machine learning applications [1–3]. Optical computing platforms promise substantial advantages, including ultra-low latency, reduced energy consumption, and high computational density [4]. Operations conducted in the optical domain are inherently high-speed, nearly free from resistive energy dissipation, and facilitate massive parallelization, which are crucial capabilities for the efficient

Yuxuan Lin and Guohao Dai contributed equally to this work. This work is supported in part by the Nansha District Key Area S&T Scheme (No. 2024ZD007). We thank Weilong Guan for helpful discussions and technical assistance with thermal modeling.

Authors' Contact Information: Yuxuan Lin, The Hong Kong University of Science and Technology (Guangzhou), China, ylin440@connect.hkust-gz.edu.cn; Guohao Dai, The Hong Kong University of Science and Technology (Guangzhou), China, gdai767@connect.hkust-gz.edu.cn; Shanshi Huang, The Hong Kong University of Science and Technology (Guangzhou), China, shanshihuang@hkust-gz.edu.cn; Yuzhe Ma, The Hong Kong University of Science and Technology (Guangzhou), China, yuzhema@hkust-gz.edu.cn.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2026 Copyright held by the owner/author(s).

ACM 1557-7309/2026/1-ART

<https://doi.org/10.1145/3818624>

Table 1. COMPARISON OF PHOTONIC COMPUTING SYSTEM MODELING FRAMEWORKS

Methods	Interconnect-Aware System Analysis	Photonic Effects Modeling	Flexible Architecture Exploration	Fast System-Level Simulation
Circuit-level Simulation [11, 12]	✗	✓	✗	✗
Manual Scaling [13–15]	✗	✗	✗	✓
CiMLoop [16], SCALE-Sim [17]	✓	✗	✓	✓
SimPhony [18]	✗	✓	✓	✓
This Work	✓	✓	✓	✓

execution of complex AI tasks. Consequently, the research community has focused extensively on developing the materials, devices, and circuits that are integral to constructing integrated optical processors [5–7]. This drive is motivated by the need for scalable, reliable, and energy-efficient computing [8, 9], leading to innovative proposals that integrate diverse photonic components and explore hybrid optoelectronic architectures that combine photonic and electronic elements, leveraging the strengths of both technologies [10].

The development of such hardware is characterized by a continuous, iterative design loop. This loop encompasses architectural definition, component selection, performance modeling, and optimization. This iterative process supports an agile design approach, allowing for swift adaptation to new requirements and promoting continuous improvement in system performance and efficiency. Consequently, accurate and efficient modeling and simulation are crucial for evaluating these hybrid optoelectronic designs and effectively exploring the vast design space.

However, precisely evaluating these complex hybrid systems remains a significant challenge. Table 1 compares the capabilities of our simulator to prior modeling frameworks in addressing the following challenges. (1) *Interconnect-Aware System Analysis*: Evaluating hybrid optoelectronic systems necessitates the modeling of key electronic components [1, 19], which can significantly impact overall system performance [14, 20–22]. In photonic accelerators, the interconnect is pivotal, connecting photonic units with electronic peripherals like memory and accumulation logic [1, 20, 23]. Consequently, capturing these interconnect overheads is indispensable for holistic full-system evaluation. (2) *Photonic Effects Modeling*: Accurate evaluation requires a tool that fully accounts for the unique characteristics of photonic computing, such as Electrical-to-Optical-to-Electrical (EOE) conversion [24], weight programming overheads [25], and critical phenomena like insertion loss [26] and thermal modeling in thermo-optic photonic integrated circuits (PICs). Quantifying these factors demands their integration into the system-level execution flow. (3) *Flexible Architecture Exploration*: Photonic systems are undergoing rapid evolution, characterized by diverse components and emerging architectures [27, 28]. This trajectory creates an urgent demand for a tool with sufficient flexibility to define and customize complex component configurations and novel hybrid architectures. (4) *Fast System-Level Simulation*: The large design space of photonic systems necessitates a modeling tool that is fast enough to support efficient design space exploration (DSE).

Existing modeling efforts for photonic computing remain fragmented. While circuit-level simulation offers high fidelity for isolated photonic elements and limited-scale circuits [11, 12], they are prohibitively resource- and time-intensive for system-scale analysis. In photonic accelerators, electronic components are necessary alongside the photonic computation units to provide system-level operations such as data movement, partial sum accumulation, and data storage [20, 29]. Some photonic computing works tend to manual scaling from individual device/circuit data, providing only rough estimations since they concentrate solely on the core photonic computation

units [13–15], thus the estimates are overly optimistic due to the failure to consider the entire system holistically. To bridge these gaps, researchers have leveraged structural analogies to adapt electronic deep neural network (DNN) simulators, such as CiMLoop [30] and SCALE-Sim [31], for photonic architectures [16, 17]; however, these approaches inherently omit critical photonic-specific factors. Even dedicated photonic simulator SimPhony [18], which considers unique optical aspects such as wavelength-level parallelism and the photonic link budget, lacks comprehensive interconnects modeling, precluding true full-system evaluation. Collectively, the limitations of existing approaches underscore the urgent need for a system-level photonic simulation framework.

To address these challenges, we introduce *LightSim*, an end-to-end modeling tool for photonic computing systems aimed at bridging the gap between rough estimates and more accurate performance evaluations that consider practical limitations. The main contributions of *LightSim* are as follows:

- We propose a holistic modeling framework that explicitly and accurately models all system-level overheads. Crucially, this includes interconnects, enabling high-fidelity calculation of hardware performance costs.
- Our modeling tool is meticulously designed to integrate the unique characteristics of photonic accelerators. We implement dedicated workload mapping and dataflow mechanisms that are fully aware of the weight programming overhead and the constraints of the limited capabilities of photonic accelerators. Furthermore, we introduce a flexible and adaptable EOE conversion interface and incorporate insertion-loss modeling with corresponding laser power analysis, as well as thermal modeling in PICs to inform crucial design decisions, enabling more accurate evaluation of photonic accelerator designs.
- We provide a flexible, modular system model with extensive parametric components, including interconnects, buffers, and adders. Furthermore, a hierarchical configuration system allows users to effortlessly define and customize novel architectures, ensuring broad applicability across diverse designs.
- Our methodology enables fast and efficient modeling for exploring the vast photonic design space. By abstracting system-level complexity while maintaining high fidelity at the circuit level, our tool significantly reduces the modeling time by avoiding time-consuming circuit simulations. This enables rapid iteration and evaluation of numerous architectural configurations.
- Two system-level modeling tasks were executed to establish the fidelity of our methodology, with results exhibiting a high degree of concordance with the published literature. Building upon this accuracy, *LightSim* explores the design space of photonic computing systems, providing early-stage results that guide improved system design.
- We quantitatively substantiate the fast modeling capability of *LightSim* through a runtime comparison against SimPhony [18], which is the only available open-source photonic architecture modeling tool to the best of our knowledge. Overall, *LightSim* demonstrates a speedup of 6.24 $\times$ across various DNN workloads. This proves *LightSim*’s exceptional computational efficiency for exploring photonic architectures.

2 Preliminary

2.1 Basic Components for Photonic Computing

Photonic computing relies on a variety of devices and circuits to function. Lasers, either off-chip or on-chip, generate the optical signals necessary for computation and communication in photonic circuits [32, 33]. Off-chip lasers offer high light-emitting efficiency and good temperature stability; however, they incur significant coupling losses and higher packaging costs. On-chip lasers

promise better energy efficiency and higher integration density, but are extremely challenging to develop on silicon due to silicon's low emission efficiency and its sensitivity to chip-scale thermal violations [34].

Waveguides carry optical signals by confining them via total internal reflection within a silicon core and silicon dioxide cladding [35]. However, propagation signals suffer from various losses (e.g., propagation loss and bending loss), which must be limited to ensure the reliability of photonic processing systems.

Modulator devices integrated along waveguides are essential components that couple, remove, and manipulate optical signals, enabling photonic information processing and communication [36]. The two most common modulator topologies are the Mach-Zehnder interferometer (MZI) and the microring resonator (MRR) [34]. MZIs utilize phase shifters and directional couplers to tune optical signals, while MRRs use a ring-shaped waveguide and tuning circuits to selectively couple or manipulate light at a specific resonant wavelength. Separately, tuning circuits are integrated with modulator devices to precisely modify characteristics like phase or amplitude of an optical signal [37].

Photo-detectors (PDs) are essential for converting modulated optical signals into a proportional electrical current [38]. Designing an effective PD involves navigating a crucial trade-off. Maximizing bandwidth requires miniaturization, which lowers power efficiency. Conversely, maintaining high efficiency necessitates providing adequate input laser power well above the required minimum to reliably drive the devices despite system losses.

Furthermore, digital-to-analog converters (DACs) and analog-to-digital converters (ADCs) handle the necessary conversion between digital data and analog electrical signals, but their high latency and energy costs make them a major performance bottleneck in silicon-photonic systems [35].

2.2 Matrix Computation with Photonic Components

Owing to its inherent parallel processing capabilities and rapid data transmission speed, photonic computing is an ideal choice for accelerating NN computations. The photonic implementations for matrix computation can be categorized into two distinct types: incoherent and coherent implementations.

2.2.1 Incoherent Implementations. Typically, in an incoherent processor, a single resonant element such as a MRR can execute scalar multiplication by modulating the light's amplitude [10]. By arranging multiple such resonators into a block, a weight bank is formed, which serves as the physical basis for matrix operations [39]. Input optical signals interact with this weight bank, where each signal is weighted by its wavelength-matched ring. Incoherent devices often exhibit wavelength-selectivity, enabling the integration of wavelength-division multiplexing (WDM) for enhanced computational throughput. This permits configurations in which weights at different wavelengths are interleaved along a shared waveguide path. In such configuration, multi-wavelength optical input signals are injected into the system. Each wavelength component interacts only with the resonators of the same wavelength while bypassing resonators of other wavelengths with negligible interference. Ultimately, weighted elements across multiple wavelengths are summed by the PD, which aggregates the power of all signals to perform the final summation.

2.2.2 Coherent Implementations. The MZI serves as the fundamental building block of coherent computing, as it is adept at realizing $U(2)$ matrix transformations [17, 40]. Constructed using two 3-dB directional couplers (DCs) and two integrated phase shifters, the MZI unit has an ideal transfer

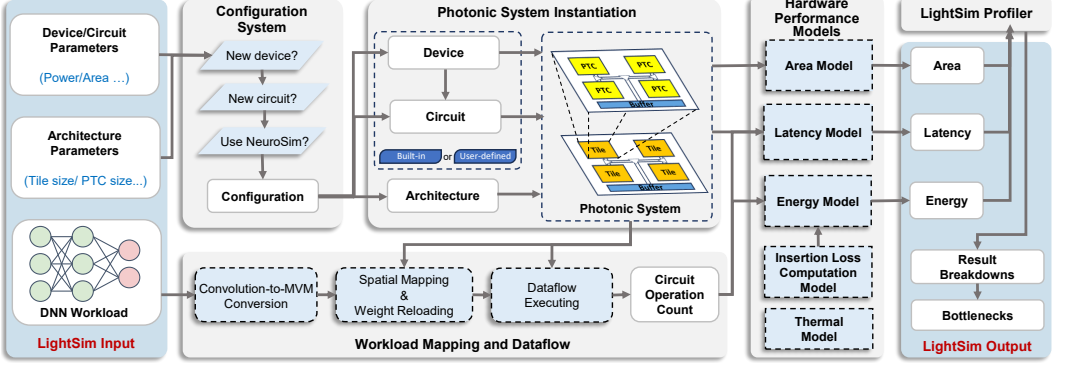


Fig. 1. *LightSim* overview.

matrix (T_{MZI}) defined by the product of its components as

$$T_{\text{MZI}}(\theta, \phi) = T_{\text{DC2}} \cdot T_{\theta} \cdot T_{\text{DC1}} \cdot T_{\phi} = \begin{pmatrix} \frac{e^{i\phi}}{2} (e^{i\theta} - 1) & \frac{i}{2} (e^{i\theta} + 1) \\ \frac{ie^{i\phi}}{2} (e^{i\theta} + 1) & -\frac{1}{2} (e^{i\theta} - 1) \end{pmatrix}. \quad (1)$$

To implement larger linear algebra operations, basic MZI units are cascaded using algorithms such as Reck [41] or Clement [42] schemes to form MZI mesh networks. Since these meshes are inherently limited to unitary matrices, implementing a general real-valued matrix M requires decomposition. This is achieved using the singular value decomposition (SVD) algorithm, which factors the weight matrix as $M = U\Sigma V^{\dagger}$. Here, the unitary matrices (U and $V^{\dagger}$) are implemented by the MZI meshes. The remaining diagonal matrix Σ is then implemented by various devices, including an array of dedicated optical attenuators or amplifiers [24], or a simpler array of MZIs configured as variable attenuators by terminating one input and one output port per MZI [43].

3 LightSim Modeling Platform

3.1 LightSim Overview

LightSim is a comprehensive modeling platform designed for end-to-end evaluation of photonic computing systems. Figure 1 is a high-level depiction of *LightSim*. *LightSim*'s workflow includes a photonic system instantiation module, a workload mapping module, a hardware performance modeling module, and a result analysis module.

In *LightSim*, users instantiate the specific photonic accelerator by defining configuration parameters for the hierarchical and modular system, alongside the input DNN workload. These inputs are managed by a configuration system. The performance metrics for fundamental devices and circuits are derived from either predefined internal resources or user-defined specifications. For instance, *LightSim* incorporated NeuroSim [44] as a built-in engine, enabling users to directly leverage its validated models to characterize electronic peripherals, such as adders, buffers, and interconnects. Crucially, by incorporating these established electronic-domain simulation backends into our unified modular system, *LightSim* establishes a holistic electro-optic co-simulation framework.

Concurrently, the workload mapping module bridges the software workload with this instantiated photonic system. It commences with transforming convolutional operations into general matrix vector multiplications (MVMs). Subsequently, it implements spatial mapping and weight reloading to assign computations to specific hardware units. Finally, the module analyzes the circuit operation count according to the dataflow. After the above steps, the instantiated photonic system and circuit

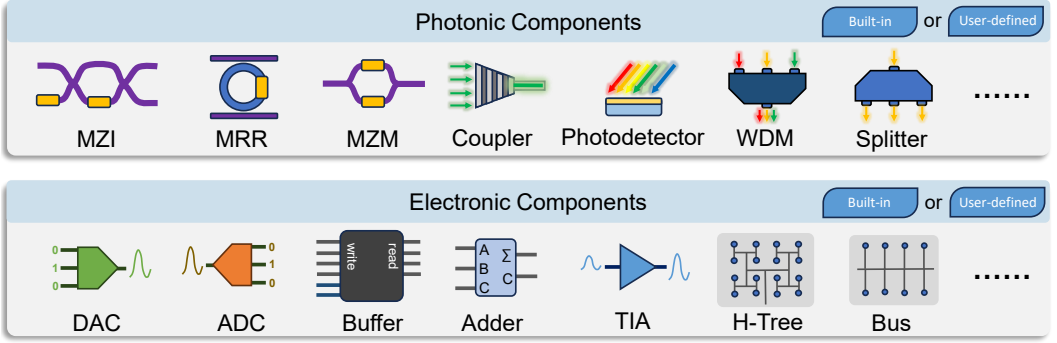


Fig. 2. The supported photonic and electronic components in *LightSim*.

operation counts are fed into *LightSim*'s hardware performance models for quantitative evaluation. Dedicated models compute the aggregate area, latency, and energy of the entire photonic system. Notably, considering insertion loss is a vital factor for photonic systems in terms of reliability and power, an insertion loss computation model is included as well. Furthermore, the thermal model is integrated to capture thermal crosstalk effects.

LightSim provides not only numerical reporting of the total area, latency, and energy metrics but also the granular result breakdowns and critical bottlenecks. These outputs offer vital insights for architectural optimization and design iterations.

3.2 Photonic Computing System Instantiation

To model a wide range of photonic systems, *LightSim* instantiates systems using a hierarchical approach. This subsection details this process, from the fundamental device and circuit components to their integration into a holistic system. For reference, the representative parameters are outlined in Table 2.

Table 2. SYSTEM CONFIGURATION PARAMETERS OF *LIGHTSIM* (IC REPRESENTS INTERCONNECT)

Modeling Level	Representative Parameters		
Component	Power	Area	Precision
PTC	Type	Size	Comp. Inst. [†]
Tile	Size	IC Bus Width	Buffer Config.
Chip	Size	IC Bus Width	Buffer Config.

[†] Comp. Inst.: Component Instantiation. PTC components can be instantiated using either built-in or user-defined devices and circuits defined at the component level.

3.2.1 Device and Circuit Components. *LightSim* constructs the photonic computing system from a collection of configurable photonic devices and electronic circuits, which serve as the fundamental building blocks for photonic systems. A key feature of our framework is its flexibility: users can either utilize the extensive set of pre-defined blocks provided by *LightSim* or introduce their own custom-defined devices and circuits to explore architectural ideas. These devices and circuits are categorized into photonic and electronic domains. For every building block, characterization is performed as outlined in Table 2: static components are defined by direct hardware metrics (e.g., power,

area, and precision), whereas parametric component behaviors utilize configuration inputs (e.g., frequency, precision, and technology node) to estimate the corresponding hardware performance costs. By capturing component behaviors through their key hardware metrics, *LightSim* achieves high circuit-level fidelity while bypassing time-consuming physical simulation.

In the photonic domain, *LightSim* provides models for fundamental devices such as MZIs, MRRs, MZMs, couplers, photodetectors, WDM, light splitters and etc. These devices can be composed into complex photonic circuits that serve as a computation core for matrix multiplication, such as MZI arrays and MRR weight banks. Additionally, the framework supports the definition of new photonic components.

In the electronic domain, *LightSim* supports a wide range of parametric components required for data conversion, storage, and communication. Through the utilization of the NeuroSim simulator [44], *LightSim* offers comprehensive built-in components, including ADCs, adders, buffers, interconnects such as H-Tree networks, etc. Unlike static models, these components are modeled analytically, where hardware performance metrics are dynamically estimated based on configuration parameters such as precision, frequency, and technology node. Specifically, interconnects are modeled using parameterized bus and H-tree structures. Each interconnect link is represented as a sequence of wire segments and inserted repeaters, where repeater insertion and sizing are automatically determined to capture realistic long-distance signal propagation. The analytical formulation takes as inputs the array dimension, unit footprint, bus width, and technology-dependent unit-length wire resistance and capacitance. Based on these hardware parameters, the physical structure, wire lengths, and repeater configuration are determined. The interconnect area is then computed from the physical footprint of all instantiated wire segments and inserted repeaters. The latency of a transfer is estimated using an RC-based delay model by accumulating the delay contributions of the traversed wire segments according to their wire lengths, while the dynamic energy of a transfer is computed using a switched-capacitance model that accounts for both wire capacitance and inserted repeater capacitance. This abstraction focuses on system-level interconnect performance modeling rather than detailed cycle-accurate network simulation.

To ensure cross-domain consistency, we enforce strict parameter alignment as the default configuration, where key operating parameters (e.g., technology node and clock frequency) are directly derived from the photonic accelerator’s specifications [45]. This ensures that the modeled electronics are physically consistent with and capable of supporting the photonic accelerator. While *LightSim* provides built-in parametric options, it also maintains extensibility by allowing users to define custom components.

3.2.2 Hierarchical System Integration. To facilitate efficient DSE across diverse hardware configurations, *LightSim* adopts a highly configurable three-level hierarchy. As shown in the Figure 3, the photonic system is abstracted into three levels: the Photonic Tensor Core (PTC), the tile, and the chip. This hierarchical design was chosen for two primary reasons: (1) The large footprint of photonic devices places significant limitations on the size of a single PTC. A PTC can typically handle a weight matrix ranging from 4×4 [24] to 128×128 [1]. To perform the large-scale matrix multiplications required by modern neural networks, we introduce the tile level to integrate multiple PTCs for collaborative computation. (2) Unlike many prior studies that often focus on isolated PTCs and overlook system-level overheads, this three-level hierarchy facilitates a holistic system-level evaluation, accounting for critical practical costs such as data movement and storage.

The components at each level are defined as follows: (1) **PTC**: As the fundamental execution unit for MVMs, the PTC is modeled by integrating the previously described photonic and electronic components for conversion and computation. *LightSim* allows users to configure critical PTC parameters, including the PTC type (e.g., MZI-based or MRR-based) and PTC size. Furthermore,

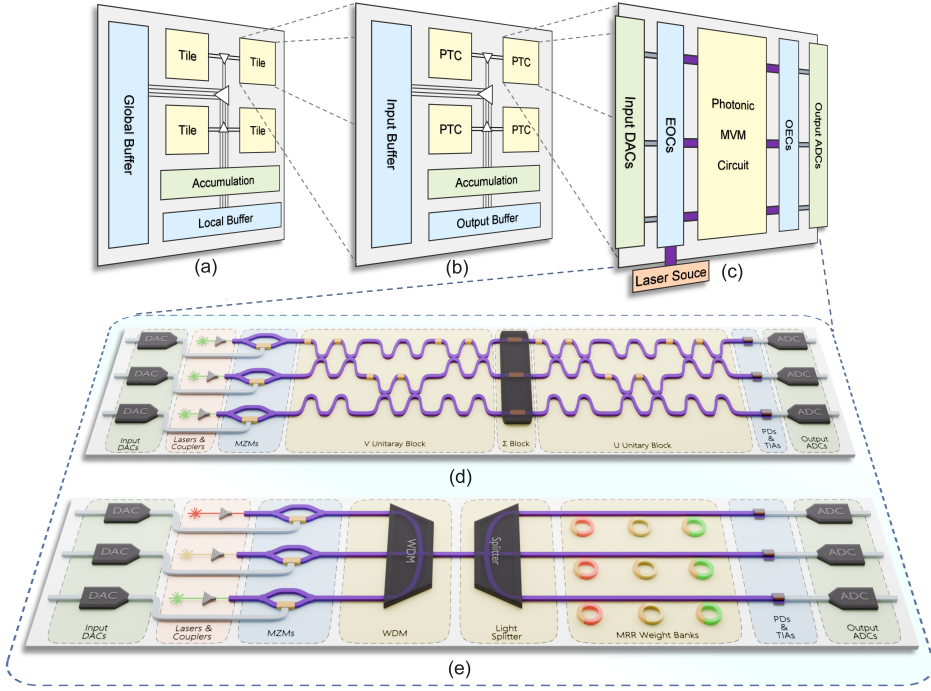


Fig. 3. The architecture of the photonic system in *LightSim*. The illustrated chip contains 4 tiles, with each tile containing 4 PTCs and each PTC responsible for a 3×3 MVM. (a) Chip. (b) Tile. (c) PTC. (d) Coherent PTC example. (e) Incoherent PTC example.

it supports flexible component selection, including DACs, electrical-to-optical converters (EOCs) like MZMs, a photonic computation circuit, optical-to-electrical converters (OECs) like PDs, trans-impedance amplifiers (TIAs), and ADCs. (2) **Tile**: a tile is a modular unit that contains multiple PTCs. It includes local input/output buffers for data staging and adders to aggregate the partial sums generated by its PTCs. For the interconnect that routes data between buffers and PTCs, *LightSim* provides built-in options for H-Tree and bus topologies, and also allows users to customize and integrate other interconnects. (3) **Chip**: the top-level chip consists of multiple tiles, designed to handle the high computational demands of neural networks. It contains a large global buffer to store the input features and output features for the entire network and extra adders to aggregate results from the tiles. When a network layer’s workload exceeds the chip’s parallel processing capacity, temporal reuse of the hardware is required, which will be discussed in detail in the following subsection. To mitigate the resulting data transfer overhead, a local buffer is positioned adjacent to the adders to accumulate intermediate results without repeated access to the global buffer. Similar to the tile level, the global interconnect circuit that connects the multiple tiles is also flexible and can be defined by users.

Across this hierarchical structure, *LightSim* establishes a practical default configuration designed to sustain massive parallelism, while retaining flexibility for user-defined customization. Specifically, all buffers at each level are sized to accommodate the fully scheduled workload, ensuring data availability for parallel execution. The modeling of memory performance explicitly accounts for bus width constraints, where the buffer access latency is bounded by the interconnect bus width. To prevent communication bottlenecks, the interconnect network is logically partitioned into multiple

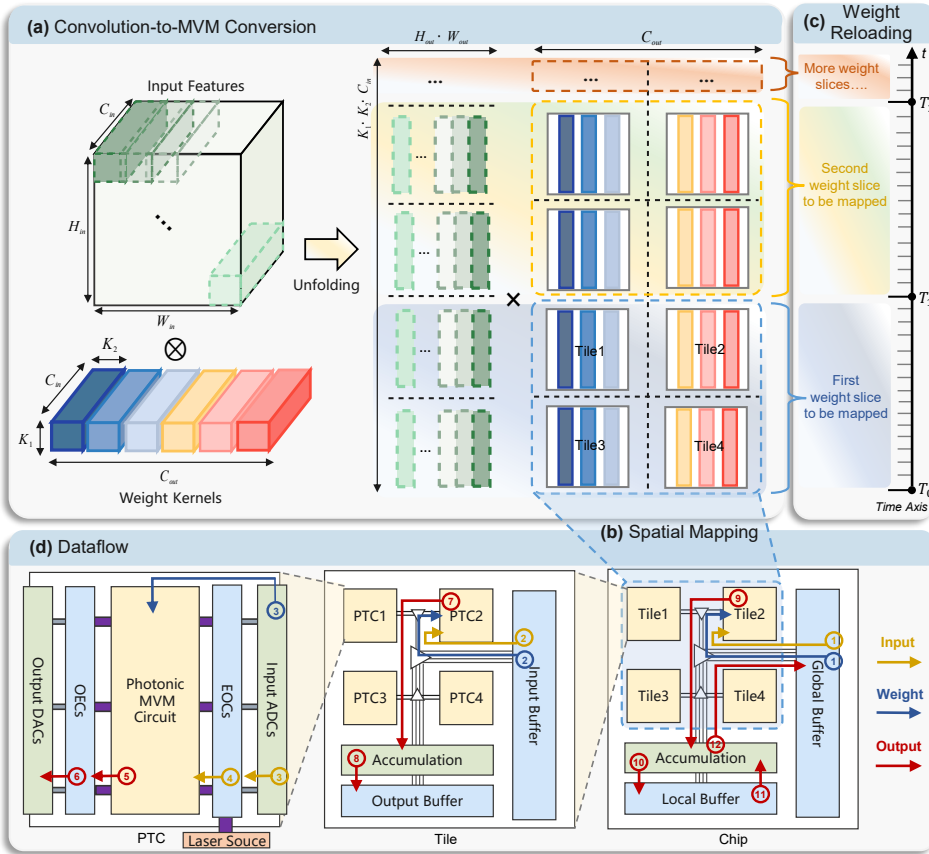


Fig. 4. Workload mapping and dataflow.

channels to serve distributed units simultaneously. Furthermore, the number of adders is configured to match the number of corresponding computing units, thereby minimizing the data read/write overhead associated with partial sum aggregation.

Beyond this baseline configuration, a distinguishing feature of *LightSim* is its flexibility, allowing for customization of the dataflow patterns and architectural hierarchy. Users can selectively mask or activate specific dataflow links or components at any level. For example, to model an architecture where input data is fetched directly from the tile-level eDRAM as in HolyLight [23], users could simply disable the input dataflow from the chip-level global buffer to the tiles. This architectural configurability empowers *LightSim* to extend the default template, thereby supporting the exploration of a broad range of novel photonic designs.

3.3 Workload Mapping and Dataflow

This subsection details the workload mapping and dataflow in *LightSim*. As illustrated in Figure 4, the workload mapping process consists of three stages: convolution-to-MVM conversion, spatial mapping, and weight reloading. The resulting dataflow is described thereafter.

3.3.1 Convolution-to-MVM Conversion. DNNs frequently employ convolutional layers, which are not inherently structured as standard MVM operations. To enable these layers to be mapped onto

a general-purpose MVM photonic computing system, a convolution-to-MVM step is performed, as shown in the Figure 4(a). Consider a convolutional layer with C_{out} kernels, each of dimension $C_{in} \times K_h \times K_w$. Each kernel is flattened into a column vector of length $C_{in} \cdot K_h \cdot K_w$, and the C_{out} vectors are stacked to form the weight matrix. The input feature map of size $H_{in} \times W_{in} \times C_{in}$ is similarly unrolled into a 2D matrix, where each column corresponds to the flattened receptive field associated with one output spatial location and therefore has height $C_{in} \cdot K_h \cdot K_w$. The unfolded input matrix contains $H_{out} \cdot W_{out}$ columns, where H_{out} and W_{out} are the height and width of the output feature maps (OFMs). Since the computation is executed as MVMs, the computing system processes the input matrix one column at a time. As a result, generating the full OFMs requires iterating over all $H_{out} \cdot W_{out}$ input columns. This convolution-to-MVM step converts the convolution operation into a standard MVM. For fully connected layers, which are already in a standard MVM format, no additional transformation is required.

3.3.2 Spatial Mapping and Weight Reloading. Our mapping strategy employs a weight-stationary approach, which is particularly well-suited for photonic computing systems as it avoids the significant modulation overhead associated with frequent weights re-encoding. The DNN model is mapped onto the photonic accelerator on a layer-by-layer basis. We assume a photonic system composed of T tiles, each containing $H_T \times W_T$ PTCs. Each PTC is characterized by dimensions of $H_P \times W_P$. *LightSim* executes mapping the workload of the convolutional layer defined in the last paragraph onto this system by considering three primary aspects: spatial mapping for distributing weights across the photonic system’s physical resources, weight reloading for managing layers that exceed the photonic system’s capacity, and weight duplication for optimizing the processing of smaller layers. We detail the chip-level mapping onto tiles. For brevity, the analogous mapping process at the tile level is not elaborated here.

Spatial mapping determines how the layer’s weight matrix is physically placed across the computational hardware, as illustrated in Figure 4(b). The full MVM operation is first partitioned along its row and column dimensions into tile-sized sub-matrices whose shapes match the $(H_T \cdot H_P) \times (W_T \cdot W_P)$ compute capacity of each tile. Each such sub-matrix is then mapped to one tile. The total number of tile-level sub-matrices is given by

$$\#tile = \left\lceil \frac{K_h \cdot K_w \cdot C_{in}}{H_T \cdot H_P} \right\rceil \cdot \left\lceil \frac{C_{out}}{W_T \cdot W_P} \right\rceil. \quad (2)$$

Among these $\#tile$ sub-matrices, at most T can be scheduled on the chip concurrently. Inside each tile, the mapped tile-level sub-matrix is further partitioned and distributed among the individual PTCs to match their $H_P \times W_P$ compute dimensions.

When the weight matrix of this layer exceeds the total capacity of the entire photonic chip ($\#tile > T$), weight reloading (Figure 4(c)) becomes necessary. In this scenario, the hardware resources are reused temporally by partitioning the weight matrix into $D_T = \lceil \#tile / T \rceil$ slices over time. Each time slice is sequentially loaded onto the available tiles and processed. Partial sums generated from these slices are buffered and subsequently accumulated in adders to form the complete result for the layer.

Conversely, if this layer’s MVM dimension is smaller than the chip’s maximum MVM capacity ($\#tile < T$), weight duplication is employed to maximize hardware utilization. In this case, the weight matrix is duplicated $D_{dup} = \lceil T / \#tile \rceil$ times across different tiles, enabling the system to process multiple input vectors simultaneously.

3.3.3 Dataflow. After mapping, *LightSim* analytically computes the operation counts for each circuit based on the dataflow. The Figure 4(d) provides a high-level schematic of these dataflow paths. For simplicity, the data paths through one representative PTC within a single tile are depicted.

Based on these defined paths, *LightSim* analytically determines the total number of operations required for each device and circuit. These operation counts serve as direct inputs to our device and circuit models, which then derive the corresponding energy consumption and latency.

At the chip level (rightmost inset in Figure 4d), the dataflow begins with input and weight data being fetched from the global buffer. This data is then distributed via the global interconnect to multiple tiles (labeled ①), which operate in parallel to perform computations simultaneously. Each tile produces a set of partial sums as its output. These partial sums are routed back through the interconnect to a chip-level accumulation unit (labeled ⑨) for aggregation. As discussed under weight reloading, a large weight matrix often requires processing over multiple time windows. To manage this, the aggregated partial sums from the accumulation unit are temporarily stored in an adjacent local buffer (labeled ⑩). In the next time window, as new partial sums from the tiles arrive, the previously stored partial sums are fetched from the local buffer (labeled ⑪) and accumulated to the new ones. This accumulation loop between labels ⑩ and ⑪ continues until all time windows for the layer have been processed. After partial sums of all time windows are accumulated, the final out results of a layer are obtained. Subsequently, they are transferred back to the global buffer (labeled ⑫) for storage, concluding the layer’s computation.

The tile-level dataflow (middle inset in Figure 4d) mirrors the chip-level principles but at a smaller scale. Within a tile, input and weight data are supplied from the tile’s input buffer and distributed to multiple PTCs through an internal interconnect (labeled ②). The PTCs then perform their MVM computations in parallel. The output results generated by each PTC are collected and aggregated by the tile’s dedicated accumulation unit (labeled ⑦). This process yields the tile’s final output, *i.e.*, a set of partial sums, which is then stored in the output buffer (labeled ⑧).

At the PTC level (leftmost inset in Figure 4d), the dataflow begins by encoding the matrix weights into the optical devices of the photonic computation circuit (labeled ③). Subsequently, input vectors, as digital signals, are converted to analog signals by DACs, which then drive EOCs like MZMs to modulate light emitted from the light source (labeled ③). Thus, the input vectors are encoded as modulated optical signals. This modulated light propagates through the photonic computation circuit, where the MVM is performed (labeled ④). The resulting output optical signals are then converted back to analog electrical signals by OECs like photodetectors (labeled ⑤) and finally digitized by ADCs for further processing (labeled ⑥).

3.4 Hardware Performance Modeling

The hardware performance model is explained in detail, highlighting the specific impacts of workload mapping and dataflow. To make this formulation more concrete, we first use a simple interconnect example to illustrate how mapped data movement is translated into interconnect transfer count and, ultimately, into latency, dynamic energy, and area. As an illustrative example, consider input data distribution from a global buffer to a 2×2 tile array through an H-tree interconnect. In this example, the H-tree structure is determined by the 2×2 tile organization and the chip layout footprint of $11.16 \times 11.16 \text{ mm}^2$, which are taken as inputs to *NeuroSim* [44]. The model in *NeuroSim* accounts for wire lengths and repeater insertion, yielding an H-tree area of $A_{ic} = 5.12 \text{ mm}^2$, a per-transfer dynamic energy of $e_{ic} = 2.52 \text{ nJ}$, and a per-transfer latency of $\tau_{ic} = 7.95 \text{ ns}$. For the first convolution layer of ResNet-18 [46], the dimension of the input feature map $W_{in} \times H_{in} \times C_{in}$ is $224 \times 224 \times 3$. The kernel dimension $K_h \times K_w \times C_{in} \times C_{out}$ is $7 \times 7 \times 3 \times 64$, with a stride of $S = 2$ and a padding of $P = 3$. With input activation precision $b_{in} = 12$ and batch size $B = 1$, the mapped input data volume in bits is

$$B_{ic-in} = b_{in} \cdot B \cdot K_h \cdot K_w \cdot C_{in} \cdot \left(\left\lceil \frac{H_{in} + 2 \cdot P - K_h}{S} \right\rceil + 1 \right) \cdot \left(\left\lceil \frac{W_{in} + 2 \cdot P - K_w}{S} \right\rceil + 1 \right). \quad (3)$$

Then the required number of interconnect transfers is $N_{\text{ic-in}} = \left\lceil \frac{B_{\text{ic-in}}}{W_{\text{bus}}} \right\rceil$, where the interconnect bus width of W_{bus} is 4,096. Accordingly, the total interconnect energy and latency for the mapped input-activation movement are

$$E_{\text{ic-in}} = N_{\text{ic-in}} \cdot e_{\text{ic}}, \quad T_{\text{ic-in}} = N_{\text{ic-in}} \cdot \tau_{\text{ic}}.$$

With all the values available, the total interconnect energy and latency are calculated as 13.61 μJ and 42.94 μs , respectively. This example shows how mapped data movement is translated into interconnect transfer count, latency, dynamic energy, and area, while the detailed formulations are presented below.

3.4.1 Energy Modeling. The sample device/circuit hardware models provided in *LightSim* encompass distinct consumption metrics, utilizing approaches based on energy per operation and others relying on continuous supply power. To ensure accurate evaluation, users are encouraged to integrate either these provided models or the specific hardware models corresponding to their proprietary photonic designs.

The energy consumed by the MVM operations at the PTC level is primarily attributed to the electrical control and EOE conversion components. Consequently, the energy cost for MVM operations (E_{MVMs}) is calculated by summing the operational energy and power costs of these key devices. For the MRR-based or MZI-based PTCs detailed in the preceding subsection, E_{MVMs} can be modeled as:

$$E_{\text{MVMs}} = (e_{\text{dac}} \cdot H_P + e_{\text{adc}} \cdot W_P) \cdot N_{\text{MVMs}} + (P_{\text{ps}} \cdot N_{\text{ps}} + P_{\text{tia}} \cdot W_P + P_{\text{pd}} \cdot W_P) \cdot T_{\text{MVMs}}. \quad (4)$$

Here, T_{MVMs} is the latency cost of the N_{MVMs} MVM operations, which will be explained later. e_{dac} and e_{adc} represent the energy required per conversion operation for the DAC and ADC, respectively. The power consumption of a single phase shifter, TIA, and PD are denoted by P_{ps} , P_{tia} , and P_{pd} . It should be noted that P_{ps} represents the static power consumed by the phase shifter to maintain its configured state. For a PTC of dimension (H_P , W_P), N_{ps} is the number of phase shifters (PSs). For instance, in an MZI-based PTC, this quantity is calculated as $(H_P(H_P - 1) + W_P(W_P - 1))/2$, assuming one PS per MZI.

The dynamic energy consumption for weight programming in PICs (E_{we}) is determined by both the selected material and the operational mechanism of the tuning devices or circuits. Importantly, E_{we} is conceptually and temporally distinct from E_{MVMs} . Nano-opto-electro-mechanical (NOEM) phase shifters use voltage-driven mechanical movement to change the refractive index, making E_{we} essentially a one-time energy, since maintaining the programmed state requires minimal static power [47]. Conversely, thermo-optic (TO) shifters require a continuous power supply to maintain the elevated temperature necessary for MVM inference. Other mechanisms, like the free-plasma effect, also exist, and users customize their E_{we} calculation based on the device's specific physics.

DNN workloads that involve operations beyond MVM are mapped onto the entire photonic computing system, which integrates various supporting components. These system overheads (latency and energy) are characterized by modeling their data movements and computations as a sequence of operations within these components. Each energy component, E_i , is calculated as the product of operation number and average energy cost per corresponding operation, i.e., $E_{\text{read-in}} = N_{\text{read-in}} \cdot e_{\text{mem}}$ ($N_{\text{read-in}}$ is the number of input data read operations and e_{mem} is the average energy cost per memory read operation). The overall energy consumption, denoted as E_{total} , is estimated by aggregating the energy consumed across the sequence of operations, as defined by:

$$E_{\text{total}} = \sum_i E_i = E_{\text{read-in}} + E_{\text{ic-in}} + E_{\text{we}} + E_{\text{MVMs}} + E_{\text{accum}} + E_{\text{ic-out}} + E_{\text{write-out}}, \quad (5)$$

where E_i represents the energy consumption for each operational stage. Specifically, these stages correspond to: input data read-in from memory ($E_{\text{read-in}}$); input interconnect transfer ($E_{\text{ic-in}}$); partial

sum accumulation (E_{accum}); output interconnect transfer ($E_{\text{ic-out}}$), and output data write-out to memory ($E_{\text{write-out}}$).

3.4.2 Latency Modeling. MVM operations in the PTC are executed using high-speed photonic computing. Once the weights are configured in the photonic circuits, the interaction between input vectors and weights occurs at extremely high speed owing to the nature of light inference.

The total execution time at the PTC level, denoted as T_{MVMs} , primarily depends on the numerical range and expressivity of the computation devices. For instance, when an 8-bit input value is represented by eight sequential 1-bit input signals, eight processing cycles are required to preserve full 8-bit precision. In our modeling, the PTC data processing rate is assumed to be fixed. After the NN workload is mapped and scheduled onto hardware resources, the execution time for the workload, T_{MVMs} , can be estimated as

$$T_{\text{MVMs}} = \frac{C_{\text{MVM}}}{f}, \quad (6)$$

considering C_{MVM} is the total number of cycles to execute the workload and f is the rate at which input vectors are processed in a PTC for MVM operations. This rate, measured in the number of input vectors successfully modulated, transmitted, computed, and converted per second, is determined by the aforementioned dataflow of the PTC.

The overall latency is the sum of all operation latency costs as $T_{\text{total}} = \sum_i T_i$. The latency of a specific stage, T_i , is calculated as the product of its required processing cycles, C_i , and the cycle duration (or latency per cycle), τ_i : $T_i = C_i \cdot \tau_i$. The number of required processing cycles, C_i , for N_i operations is dependent on two workload mapping and dataflow factors: (1) Weight reloading: In this scenario, the cycle count is scaled by the partitioning factor: $C_i = N_i \cdot D_T$. (2) Weight duplication: This factor quantifies the cycle count as $C_i = N_i / D_{\text{dup}}$ if D_{dup} copies of the weight are concurrently mapped onto the hardware. It must be noted that the weight encoding latency, T_{we} , is incurred only once per weight load and is therefore not subject to duplication (D_{dup}). For instance, the overall latency for executing a small neural network where the duplication strategy is applied to the majority of stages is given by:

$$T_{\text{total}} = T_{\text{we}} + (T_{\text{read-in}} + T_{\text{ic-in}} + T_{\text{MVMs}} + T_{\text{accum}} + T_{\text{ic-out}} + T_{\text{write-out}}) / D_{\text{dup}}. \quad (7)$$

The stage latencies T_i (e.g., $T_{\text{read-in}}$, $T_{\text{ic-in}}$, etc.) are defined by their corresponding stages listed previously.

3.4.3 Area Modeling. The total chip area is computed as the sum of all constituent component areas, including not only the PTCs but also system overheads such as interconnect, buffers, and adders. In addition, the next subsection presents a more detailed PTC-level area modeling approach, which goes beyond simple component-wise summation. Different PTC-level area modeling methods can be adopted depending on the desired modeling granularity.

3.4.4 PTC-Level Physical Constraints and Loss Modeling. PTC-level modeling accounts for physical non-idealities and loss effects in PICs that influence system performance. In this subsection, we consider several key factors in PICs, including thermal behavior under thermo-optic modulation, insertion loss, and the resulting laser power analysis, and PTC-level area modeling.

Thermal Modeling. Thermal crosstalk is a well-known challenge in thermo-optic PIC systems [14, 48, 49]. To capture this effect, COMSOL simulations are used to generate detailed temperature profiles around individual microheaters, which are then used to construct compact thermal models for PTC-level analysis. The temperature distribution is described in two dimensions using the radial distance d from the heater source and the angular position θ . To represent the temperature

distribution as a function of both distance and angle, two complementary approaches are adopted, including a two-dimensional lookup table (LUT) for high accuracy and a compact polynomial model for improved computational efficiency.

In the LUT-based approach, temperature values at discrete sampling points (d, θ) are precomputed from the COMSOL simulations. During runtime, the temperature rise at a given location is obtained through interpolation between these stored values, as

$$\Delta T(d, \theta) \approx \text{Interpolate}(\Delta T_{\text{LUT}}(d_1, \theta_1), \Delta T_{\text{LUT}}(d_2, \theta_2), \dots). \quad (8)$$

Although this method captures fine spatial variations in the temperature profiles, it requires relatively large memory and can become computationally demanding for large circuits. To improve computational efficiency, a compact model is introduced in which the temperature rise generated by the microheater j with applied power P_j is expressed as $\Delta T_j(d, P_j) = f(d, \theta)P_j$. The spatial temperature distribution $f(d, \theta)$ is approximated using Chebyshev polynomials along the radial dimension, $f(d, \theta) \approx \sum_{n=0}^N b_n(\theta)T_n(d)$, where $T_n(d)$ are Chebyshev polynomials and $b_n(\theta)$ are angle-dependent coefficients obtained by fitting the temperature profiles. To model thermal crosstalk among multiple microheaters, the temperature contributions from individual heaters are superimposed according to their spatial arrangement and input powers. Each microheater is characterized by its position (x_i, y_i) and applied power P_j , while the thermal interaction between heaters i and j is described by the function $f(d_{ij}, \theta_{ij})$, where d_{ij} is the Euclidean distance and θ_{ij} is the relative angle determined by the circuit layout. The total temperature rise as heater i can therefore be written as

$$\Delta T_i = \sum_{j \in \mathcal{N}(i)} f(d_{ij}, \theta_{ij})P_j, \quad (9)$$

where $\mathcal{N}(i)$ denotes the set of heaters within the interaction range, and the term $f(0, 0)$ represents the self-heating distribution. This formulation enables efficient evaluation of temperature variation in PTC.

Insertion Loss Modeling. In PICs, as the optical signal generated by a laser travels towards the PD, it undergoes passive propagation. During this transit, the optical power is attenuated due to the insertion loss introduced by individual components. The scalability of most photonic computing applications is constrained by limited laser output power and wall-plug efficiency, as well as significant on-chip insertion loss [34]. Thus, it is important to provide an accurate insertion loss model to offer an accurate insertion loss estimation and identify the critical path.

To facilitate a flexible and granular estimation of insertion loss, we model the optical signal path within the PTC as a directed acyclic graph (DAG). In this representation, each node corresponds to a photonic device along the optical path and is annotated with its insertion loss. Directed edges represent the optical connectivity between devices and define the possible propagation paths of optical signals. Under this abstraction, the insertion loss of any optical path can be obtained by accumulating the insertion losses of the devices along that path. This graph-based approach allows us to compute the critical path loss efficiently, an example of which is depicted in Figure 5.

Laser Power Analysis. To ensure the accuracy of the system, it is necessary to compensate for the insertion loss by increasing the input laser power. The minimum required laser power can be expressed as $P = \frac{(2^{b_{\text{out}}})^2 \cdot (q\Delta f/4)}{\eta_{\text{laser}} \cdot \eta_{\text{IL}} \cdot \eta_{\text{det}}}$, with b_{out} being the bit precision of the output ADC, q being the elementary charge, Δf being the bandwidth of the detector, η_{laser} , η_{IL} , and η_{det} representing the wall-plug efficiency of the laser, the total transmissivity corresponding to the critical insertion loss along the optical path from the laser to detector, and the efficiency of the PDs [45].

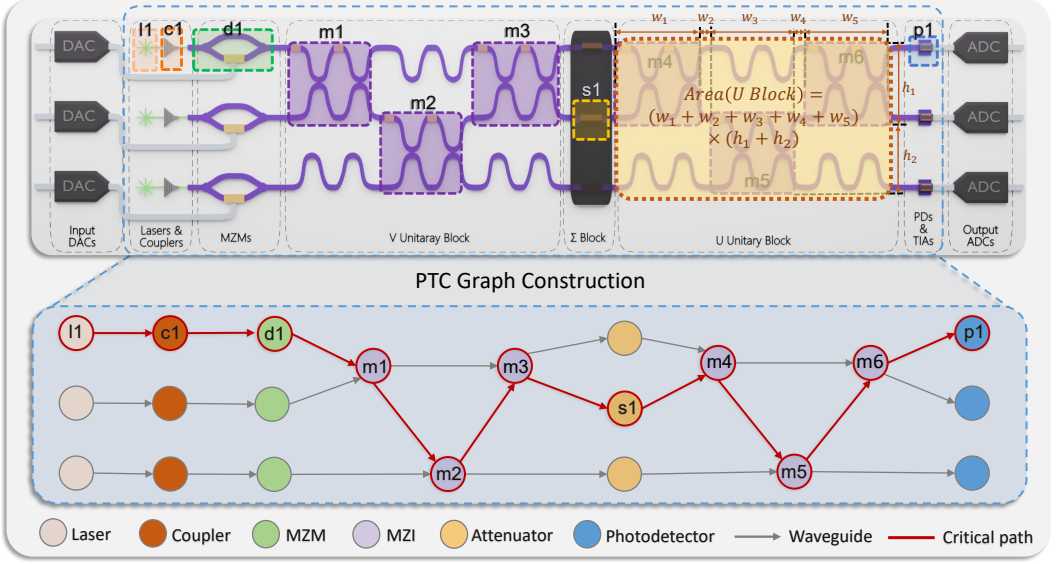


Fig. 5. PTC graph construction and critical path identification. For visual clarity, only area modeling for the U block is illustrated in the graph. The same underlying principles of area estimation apply to the entire graph.

PTC-Level Area Modeling. Beyond insertion loss estimation, the same graph abstraction is also leveraged for PTC-level area modeling. In this model, the width of the PTC is estimated as the total horizontal span of a device path, calculated as the sum of the widths of the photonic devices along the path plus the layout spacing between adjacent devices. The height of the PTC is determined by the maximum accumulated vertical height among the devices aligned along the same horizontal path. For example, a 3×3 PTC configuration illustrated in Figure 5, the vertical dimension of the 3×3 MZI mesh is typically larger than that of other components such as modulators. As a result, the height of the PTC is determined by the MZI mesh in this case. This graph-based estimation only considers photonic devices along the optical path. An example of area modeling for the U block is also shown in Figure 5. Electronic components such as ADCs and DACs are incorporated separately when estimating the overall PTC area.

4 Experimental Results

This section presents a comprehensive evaluation of *LightSim*. First, we validate the accuracy of our framework by modeling two typical photonic computing systems: the coherent ADEPT [45] and the incoherent HolyLight-M [23]. Subsequently, we provide a runtime analysis to demonstrate the fast modeling capability and high computational efficiency of *LightSim*. At last, we conduct a DSE to showcase the usefulness of *LightSim*. The source code is available at <https://github.com/OEDA/LightSim>.

4.1 Validation with Coherent Photonic Computing System ADEPT

To validate *LightSim* for coherent accelerator architectures, we simulated the ResNet-50 workload with a batch size of 58 on the ADEPT system [45] of a 128×128 photo-core operating at 10 GHz. Most device and circuit parameters are adopted directly from ADEPT work and its corresponding references to maintain alignment. Given the scarcity of detailed SRAM and non-general matrix multiplication (Non-GEMM) component information (which includes element-wise, non-linear,

and data rearrangement operations) within ADEPT, we made specific substitutions. We employed the NeuroSim-modeled SRAM and modeled a key part of the digital processing unit. While ADEPT models a digital electronic processing unit, its specification for Non-GEMM functionality does not explicitly detail the component for partial sum accumulation, which is a necessary digital operation. Therefore, we substituted the combination of the Non-GEMM component and the unmodeled partial sum accumulation logic with a NeuroSim-modeled adder. We acknowledge that this adder only covers a subset of the total digital processing functionality, which results in a smaller footprint but higher energy consumption compared to the original digital processing unit. The simulation results demonstrate a strong correlation with published data.

4.1.1 Throughput Validation. The simulated throughput of *LightSim* is 31,430, which is marginally smaller than the published ADEPT data of 35,698. This discrepancy is primarily attributed to a divergence in execution modeling assumptions. The ADEPT work derived its latency model from SCALE-Sim [31], a simulator that abstracts certain microarchitectural details. In contrast, our model accounts for the detailed mapping of weights onto the hardware, which results in a slight difference in execution time. Specifically, these refined execution considerations lead to a larger simulated latency for *LightSim* relative to the reference data.

4.1.2 Area Validation. The area breakdown comparison is shown in Figure 6. The observed minor difference from the reported data are due to the approximation of certain area metrics from visual figures in the reference papers. This introduces an unavoidable estimation error, which is the cause of the observed differences in the optical and EOE area metrics.

4.1.3 Power Validation. The comparison of power breakdown is detailed in Figure 6. With the increased latency discussed above, this slower execution results in a smaller overall power consumption measurement. This behavior arises from our energy modeling, in which the total energy is calculated as the sum of all components contributions. Some components, such as modulators and detectors, operate in an energy-per-operation mode, where energy consumption is modeled as the product of a fixed per-operation energy cost and the total number of operations required by the workload. Since the total number of these operations is fixed, their total energy is amortized over a longer time, resulting in lower average power.

4.2 Validation with Incoherent Photonic Computing System HolyLight-M

To validate the modeling accuracy of *LightSim*, we first reproduce the incoherent photonic computing system, HolyLight-M [23]. Furthermore, we introduce another modeling tool, SimPhony[18], to reproduce HolyLight-M as an additional baseline. We then compare our modeling results against those reported in the reference. HolyLight-M features a four-level hierarchy comprising 28 tiles, where each tile contains one processing element (PE), and each PE contains eight PTCs. Although our framework was previously introduced with a three-level hierarchy, it is also designed to fully support four-level architectures. The device, circuit and architecture parameters in our modeling are configured to match those of HolyLight-M. The input data resolution is set to 16 bits and the weight resolution is 16 bits similarly, with each MRR representing 2 bits. The system clock frequency is set as 1.28 GHz. The ADC and DAC resolutions are set to 8 bits and 1 bit, respectively. Furthermore, HolyLight-M utilizes a network-on-chip (NoC) for inter-tile communication. We have modeled this NoC based on the provided specifications in HolyLight-M and integrated it as a new circuit module within *LightSim*. Following the experiment setting of the HolyLight-M [23], we employ LeNet-5 [50], MCDNN [51], AlexNet [52], and ResNet-18 [46] as evaluation workloads. These neural network models are simulated in *LightSim* with a batch size of 10.

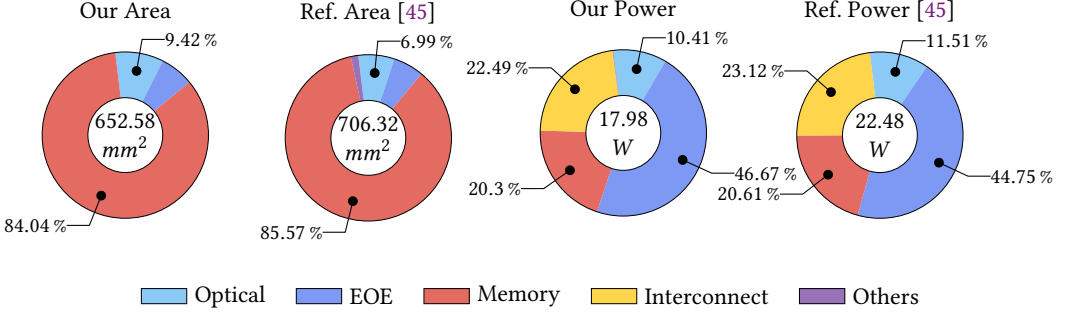


Fig. 6. Area and power breakdown comparison. The "EOE" category encompasses metrics derived from detectors, modulators, ADCs and DACs, whereas the "Others" category accounts for overhead from adders and non-GEMM components. The shared legend describes all components considered. Note that the interconnection area was not modeled, so it is omitted from the area breakdown. Only the reference area metric explicitly includes a significant contribution for the "Others" category. For all other metrics (simulated area, simulated power, and reference power), the "Others" category is excluded from the visual presentation of the breakdown, as its contribution is negligible (less than 1%). This minimal component is, however, accounted for in the overall totals..

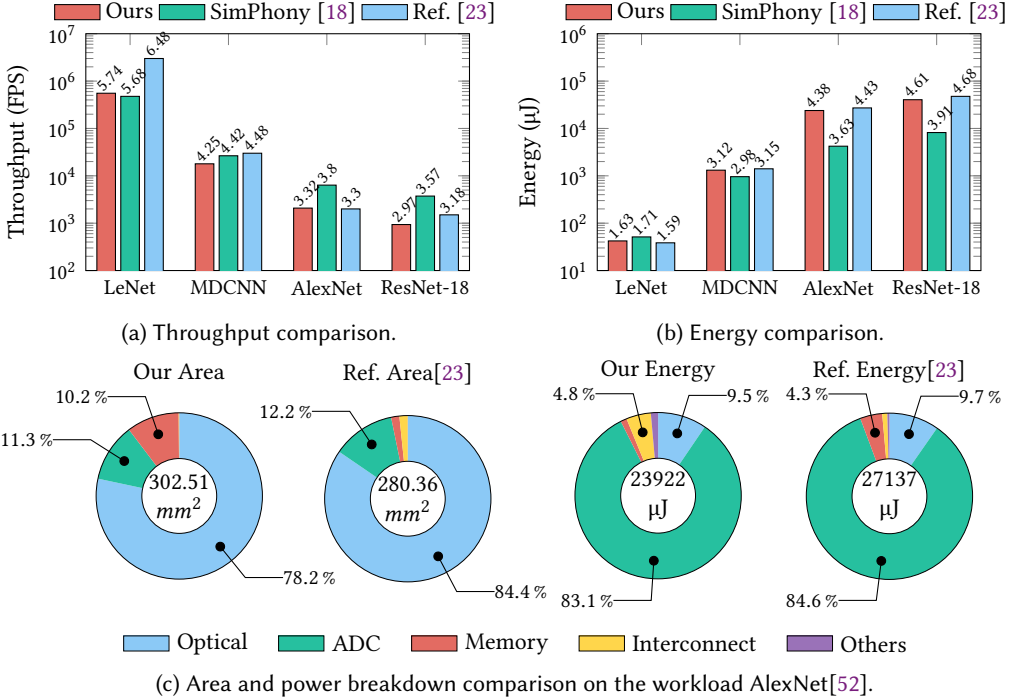


Fig. 7. Comparison between simulation results from *LightSim*, *SimPhony* [18] and reported data from the HolyLight-M reference [23]. (a) Throughput and (b) energy consumption comparison across various neural network workloads. (c) Detailed breakdown of area and energy for AlexNet. The "Others" category in the breakdown includes the overhead from adders, DACs, and circuits for sigmoid and ReLU activations.

4.2.1 Throughput Validation. Figure 7a presents a comparison of throughput, measured in frames per second (FPS), across the different neural network models. The numerical value above each bar in Figure 7a is $\log_{10}(\text{FPS})$. It should also be noted that the reference FPS results were extracted from a bar chart in the original paper and may therefore have limited precision. For larger workloads, including MCDNN, AlexNet, and ResNet-18, our simulated results closely align with the reference data, falling within the same order of magnitude, which validates the modeling accuracy of *LightSim*. However, a notable discrepancy is observed in the results for LeNet-5. This deviation stems from LeNet-5’s small model size, which occupies only 9 of the 28 available tiles on the HolyLight-M system. In the original HolyLight-M implementation, the entire LeNet-5 model is mapped directly onto the system, enabling a deeply pipelined execution that yields very high throughput. In contrast, *LightSim* is designed with a more general approach, mapping one layer at a time with weight duplication as discussed in section 3.3.2. This layer-by-layer mapping strategy incurs additional weight reconfiguration overhead, resulting in lower throughput for small networks like LeNet-5 and explaining the deviation from the HolyLight-M results.

Furthermore, this deviation for LeNet-5 is identically observed in the SimPhony results. While *LightSim*, SimPhony, and the reference data exhibit the exact same relative throughput trend across all evaluated workloads, *LightSim* achieves significantly higher accuracy, with its absolute FPS values aligning much more closely with the reference data than SimPhony.

4.2.2 Area Validation. As shown in Figure 7c, the area breakdowns are largely analogous between our modeling and the reference results. In both breakdowns, optical components account for the largest proportion of the total chip area. The total area estimations are also closely matched, with *LightSim* yielding 302.51 mm^2 compared to the reference area of 280.36 mm^2 . Minor discrepancies between them are primarily attributed to our use of the NeuroSim component for modeling memory circuits and bus interconnects. We additionally evaluated the area using SimPhony [18], which reported a total area of 323.82 mm^2 . Notably, our estimated area is closer to the reference data than that of SimPhony.

4.2.3 Energy and Power Validation. In our validation against HolyLight-M [23], we first evaluate the total energy consumption. Since the original HolyLight-M framework assumes a constant power draw across all circuits regardless of the executed DNN model, it reports a fixed power of 66.88W rather than workload-specific energy metrics. Therefore, to establish a baseline “reference energy” for comparison, we perform a component-wise calculation: we multiply the static power of each component reported in [23] by its corresponding execution latency, and then sum the resulting values.

Because the HolyLight-M reference is not open-source and lacks a detailed latency breakdown, i.e., the execution time of each individual component, we utilize the component-wise latencies measured by *LightSim* to compute this reference energy. This substitution is highly justified by the rigorous throughput cross-validation in Section 4.2.1, which firmly established the reliability of *LightSim*’s timing model. Figure 7b illustrates the energy consumption comparison across the evaluated neural networks, with the numerical value above each bar representing $\log_{10}(\text{Energy})$. As shown in Figure 7b, our simulated energy demonstrates high fidelity with the reference energy across all four DNN models. In contrast, the energy estimated by SimPhony diverges significantly from the reference. This strongly validates the accuracy of *LightSim*’s energy modeling. To provide deeper insights, we present an energy breakdown for the AlexNet workload [52] in Figure 7c. This figure demonstrates that the proportional energy breakdown from *LightSim* is highly consistent with the HolyLight-M reference, correctly identifying ADCs as the dominant energy contributor in both cases. Note that for a fair comparison, we excluded the laser energy, as it was also neglected in the original HolyLight-M evaluation.

To explicitly decouple our validation from the aforementioned timing assumptions, we further conduct a direct power validation. In *LightSim*, the total power of a network is calculated dynamically: we divide the accumulated energy of each individual component by its respective runtime, and then sum these component-wise power values. While the reference power reported in the HolyLight paper is a static 66.88W, *LightSim* captures workload-specific variations, simulating the dynamic power consumption for LeNet, MCDNN, AlexNet, and ResNet-18 at 70.43W, 59.71W, 61.11W, and 57.54W, respectively. Despite the static nature of the reference, our simulated absolute power aligns well with the 66.88W result, and identically confirms that the ADC remains the most significant power bottleneck. This close alignment independently and robustly validates the accuracy of *LightSim*'s power modeling.

4.3 Runtime Analysis

To quantitatively substantiate the fast modeling capability of *LightSim*, we evaluate its simulation speed and hardware scaling sensitivity against an open-source baseline simulator, SimPhony [18]. All evaluations are executed on the same host machine to ensure a fair comparison. To guarantee a fair head-to-head comparison, we isolate the hardware modeling process in both tools. Specifically, we configure SimPhony to bypass its software accuracy evaluation, ensuring that both simulators only compare the runtime strictly required to estimate energy, latency, and area.

4.3.1 Simulation Speed Comparison. To validate the simulation efficiency of *LightSim*, we compare its speed against SimPhony [18] across a diverse set of neural network workloads. Specifically, we use a benchmark suite consisting of six workloads, including AlexNet[52], MCDNN[51], LeNet-5[50], ResNet-18[46], MobileNet-v3-Large[53], and Inception-v4[54]. For each workload, we run the simulation five times and average the execution time. We then sum the average runtimes for all six workloads to calculate the total runtime. *LightSim* achieves an average total runtime of 1.77s, whereas SimPhony requires 11.06s. This demonstrates that *LightSim* achieves a 6.24 $\times$ overall speedup over SimPhony. Notably, the majority of the total execution time in both simulators is dominated by Inception-v4 and MobileNet-v3-Large, which collectively account for approximately 74% of SimPhony's total runtime and 72% of *LightSim*'s runtime. This concentration of simulation time is expected, as these two models feature significantly more complex network topologies compared to other benchmarks, necessitating more intensive hardware mapping and evaluation overhead. Nonetheless, even on the most demanding workload, Inception-v4, *LightSim* completes the evaluation in just 0.91s compared to SimPhony's 6.03s, demonstrating a 6.59 $\times$ speedup. By consistently maintaining this substantial performance advantage across all scales of networks, *LightSim* solidly proves its superior efficiency.

4.3.2 Runtime Sensitivity to Hardware Scaling. For system-level modeling frameworks, maintaining a stable simulation speed when scaling up hardware architectures is critical for efficient design space exploration (DSE). To analyze this, we ran the ResNet-18 workload while exponentially scaling up the hardware configurations from 2×2 to 1024×1024 . The Figure 8 illustrates the runtime sensitivity of both simulators. In the Figure 8a, we scale the number of PTCs while keeping the tile count fixed; in the Figure 8b, we scale the tile counts while fixing the PTC configuration.

As shown in both subfigures, *LightSim*'s runtime remains perfectly flat (maintaining roughly 10^{-2} seconds) and is strictly independent of the scale of the underlying hardware components. This proves that *LightSim* features an $O(1)$ scaling simulation time with respect to hardware size, avoiding the dimension explosion and exponential slowdowns. Consequently, the DSE sweep cost in *LightSim* is negligible. Sweeping through thousands of photonic architectures can be completed in minutes (e.g., evaluating 10,000 distinct configurations takes only ~ 8 minutes), making *LightSim* a highly agile and practical tool for architectural explorations.

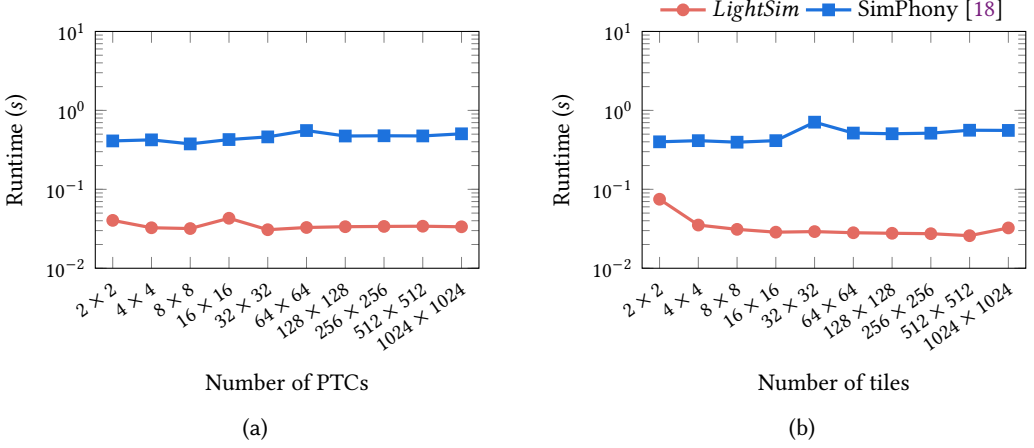


Fig. 8. Runtime sensitivity of *LightSim* and SimPhony [18] to varying (a) PTC counts and (b) tile counts.

4.4 Design Space Exploration

In this section, we demonstrate a DSE to show the usefulness of *LightSim*. The exploration is based on an photonic computing system that integrates the coherent PTC from the ADEPT [45] within a three-level hierarchy using H-Tree and bus interconnects. We conduct this DSE at both circuit and architectural levels, showcasing how *LightSim* can inform the design process of photonic system. Although some of the observed trends are expected, they still help confirm that the simulator correctly captures known scaling behaviors and can therefore support broader architectural exploration.

4.4.1 Circuits. We explore circuit-level design parameters from the perspectives of the electrical interface (ADC/DAC precision), interconnects, and PTC layout.

Data Precision. To analyze the impact of data precision, we perform a DSE on the ResNet-18[46] workload. In this study, the photonic computing system is composed of 4 tiles, where each tile contains 4 PTCs. We independently vary the precision for inputs, weights, and outputs from 4 to 10, while keeping the other two precisions fixed at 10 bits.

Figure 9 illustrates the trends in normalized energy and latency when the precision of inputs, weights, and outputs are varied. Based on the results in Figure 9, we analyze how the precision of inputs, weights, and outputs affects system energy and latency. (1) Latency is predominantly sensitive to input precision, showing a nearly 10x increase as the bit width scales from 4 to 10. This sensitivity is attributed to the fact that the total volume of input data for ResNet-18 is substantially larger than that of the model’s weights and outputs. Therefore, an increase in input precision incurs greater data movement through the interconnects, leading to a significant rise in overall latency. (2) Energy is predominantly influenced by weight precision, which causes a 10x rise in energy consumption as the weight bit width scales from 4 to 10. This sharp increase is attributed to the power of the DACs used for programming the MZI weights, which scales exponentially with the number of bits. (3) In contrast, output precision has a more moderate impact on both latency and energy. This is because the overheads tied to the output—the ADC energy consumption and data transfer latency—constitute a smaller fraction of the total system cost compared to the dominant overheads from input data transfer and weight programming.

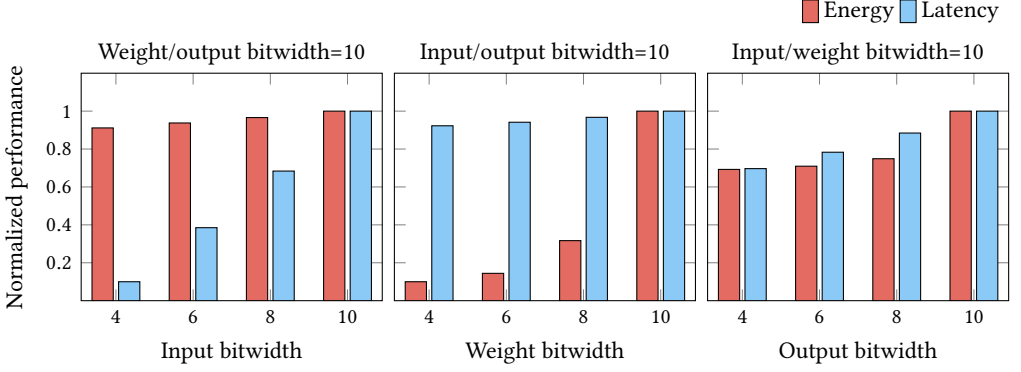


Fig. 9. Design space exploration of input, weight and output precision.

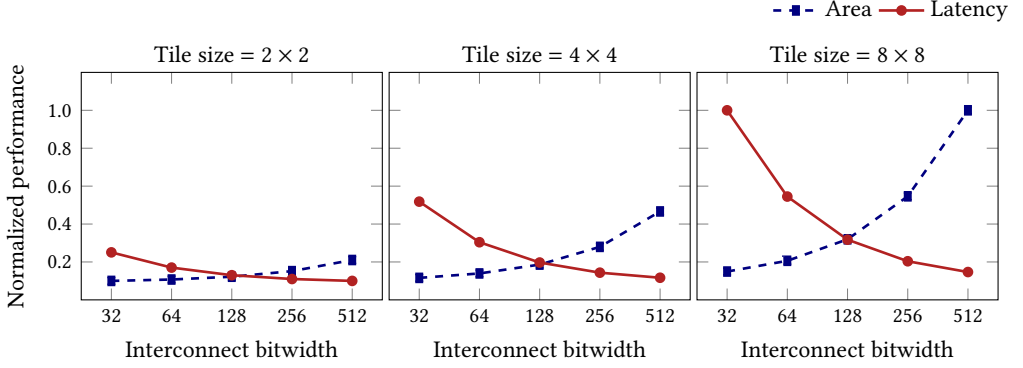


Fig. 10. Design space exploration of interconnection.

This exploration reveals a critical design insight for MZI-based photonic systems executing large, input-heavy models like ResNet-18: input precision serves as the primary lever for controlling system latency, while weight precision is the dominant factor determining energy efficiency.

Interconnect. We evaluate the design space of the interconnect network by varying its bitwidth across different tile sizes (2×2 , 4×4 , and 8×8). In this experiment, we deploy the ResNet workload on the HolyLight architecture and measure the normalized area and inference latency of interconnects. As illustrated in Figure 10, increasing the interconnect bitwidth effectively alleviates communication bottlenecks, leading to a noticeable reduction in latency. However, this performance gain comes at the cost of requiring more physical interconnect area. This trend clearly demonstrates a fundamental trade-off between area overhead and latency. Furthermore, this trade-off becomes increasingly pronounced at larger tile sizes, highlighting the capability of *LightSim* to help designers identify the optimal interconnect configurations under varying system constraints.

PTC-Level Spacing-Ratio. To investigate the PTC-level trade-off between thermal crosstalk and footprint, we perform a DSE on the spacing ratio of the MZI mesh. In this experiment, a thermo-optic MZI mesh following the Clements topology is adopted as the representative structure. The baseline center-to-center distances between adjacent MZIs are $400\mu\text{m}$ in the horizontal direction and $70\mu\text{m}$ in the vertical direction, and the spacing ratio is applied to these distances in the MZI

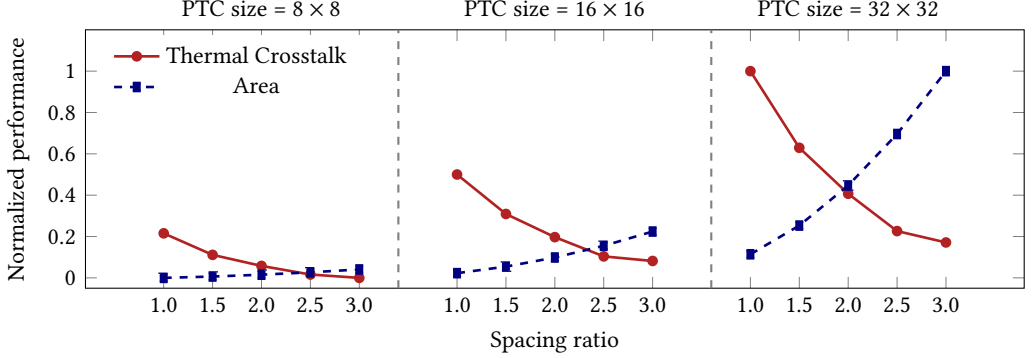


Fig. 11. Design space exploration of MZI mesh spacing ratio.

mesh. Other key distances between components are defined as follows: $150\mu\text{m}$ between the coupler and the modulator, $150\mu\text{m}$ between the modulator and the MZI mesh, and $30\mu\text{m}$ between the MZI mesh and the PD [14]. All non-layout design parameters, including the DNN workload and system-level configuration, remain consistent with those used in the previous DSE case. In this case, we specifically adopt PTC-level area modeling with layout information to enable a more accurate evaluation of the footprint under different spacing ratios and PTC sizes. Fig. 11 reports the normalized thermal crosstalk and normalized area under different spacing ratios and PTC sizes.

We observe that thermal crosstalk increased substantially with PTC size. For example, under the same spacing ratio, the 32×32 PTC consistently exhibits much higher thermal crosstalk than the 16×16 and 8×8 cases, suggesting that thermal management becomes increasingly critical as the photonic tile scales up. On the other hand, increasing the spacing ratio reduces thermal crosstalk for all PTC sizes, since larger device spacing weakens thermal coupling among adjacent MZIs. However, this reduction is clearly nonlinear: the benefit is significant at small spacing ratios but gradually saturates as the spacing ratio continues to increase. Meanwhile, the area cost grows monotonically, and the increase becomes more pronounced at larger spacing ratios. Therefore, simply enlarging the MZI spacing is not always cost-effective. Instead, the results suggest that a moderate spacing ratio may provide a better balance between thermal crosstalk mitigation and area efficiency.

4.4.2 Architecture. To understand the scalability and performance trade-offs of photonic computing systems, we conduct an architectural DSE with the ResNet-18 [46] workload on the same previously mentioned system. This DSE investigates two primary scaling dimensions: the size of the PTCs and the total number of tiles composing the system.

In the first scaling DSE study, we fixed the system architecture to consist of 2×2 tiles, with each tile containing 2×2 PTCs. We then varied the size of each PTC from 10×10 to 120×120 to study the impact of scaling up the computational core. The normalized area, energy and latency are shown in Figure 12a. We analyze the trends in area, energy, and latency, explaining the underlying reasons for each. (1) Total area grows with PTC size, as the optical core and its driving DACs, which dominate the area budget, scale quadratically with the PTC’s dimensions. (2) Total energy consumption follows a U-shaped curve, reaching an optimal point at a mid-range PTC size. Initially, energy decreases because larger PTCs can process more data at once, which reduces the significant overhead from repeatedly activating DACs during the weight reloading required by smaller PTCs. However, for excessively large PTCs, the large number of the DACs required to program the quadratically-scaling

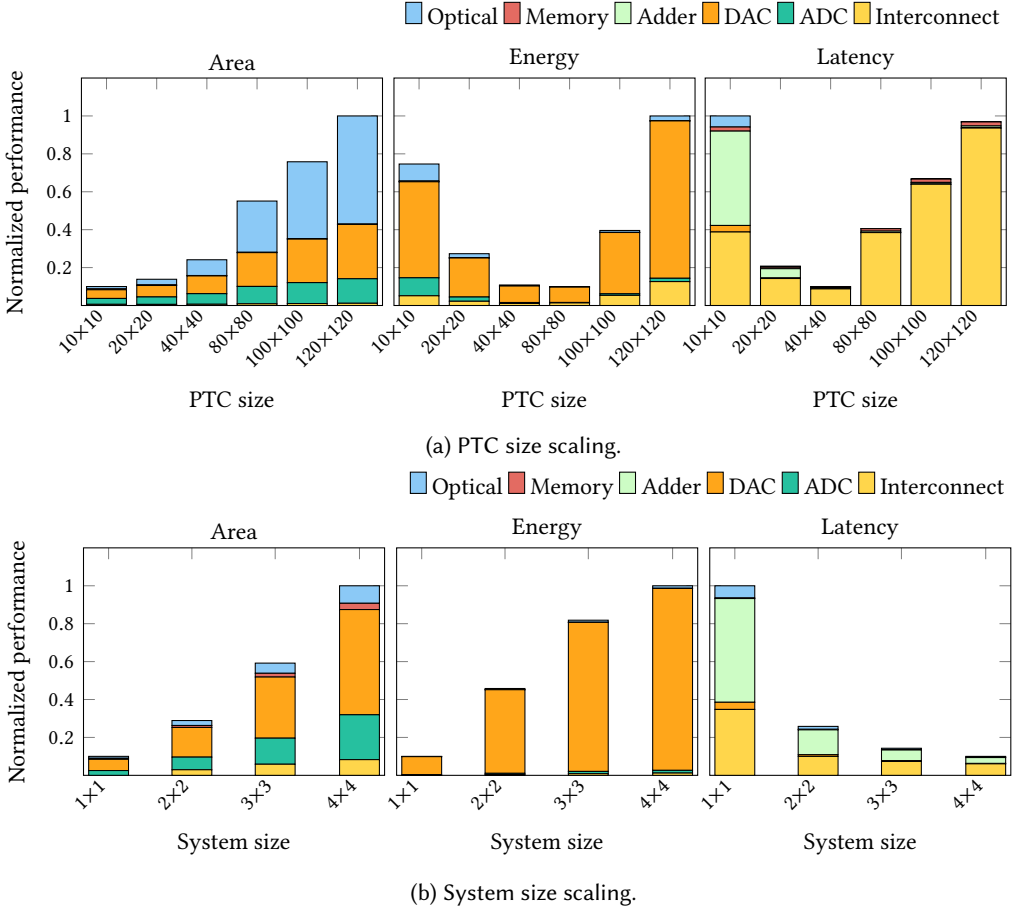


Fig. 12. Design space exploration of architecture size.

PTCs become the dominant factor, causing the total energy to skyrocket. (3) Latency also exhibits a U-shaped trend. The initial latency reduction comes from using larger PTCs to reduce the times of weight reloading, which reduces the interconnect traffic and accumulation overhead previously required between many small Tiles. Conversely, once the PTC becomes physically oversized, the delay on the larger interconnects between different PTCs becomes the new primary bottleneck, causing the total latency to increase again.

In the second scaling DSE study, we fixed the tile configuration containing 2×2 PTCs with the PTC size at 10×10 . We then varied the number of tiles in the system from a 1×1 array to a 4×4 array to study the impact of scaling the system size. The normalized area, energy and latency are shown in Figure 12b. We analyze the trends in area, energy, and latency, explaining the underlying reasons. (1) Total area and energy scale nearly linearly with the system size. The component breakdown remains proportionally consistent across different system sizes, with DACs being the primary energy consumer and a major contributor to area. (2) Total latency decreases significantly as the system size increases. This is a direct result of increased spatial parallelism. A larger system can process more sections of the workload simultaneously, drastically reducing the need for weight reloading. The breakdown clearly shows that the latency contributions from

adder and interconnect components, which are associated with aggregating partial results over time, diminish as more parallel hardware becomes available.

Our architecture DSE reveals two primary bottlenecks inherent in photonic computing system design. On the energy front, the necessity of frequent, high-speed E-O-E conversions requires power-hungry DACs and ADCs, which our results confirm consistently dominate the total system energy budget. Meanwhile, for latency, the large physical footprint of photonic devices makes the interconnect delay a dominant performance limiter. This finding highlights a critical need: since interconnects are a major bottleneck, accurately modeling them is essential for realistic performance estimation, which is a core capability that *LightSim* provides.

5 Conclusion

In this article, we propose *LightSim*, an end-to-end simulation framework for photonic computing systems. *LightSim* is designed to address the critical challenges in photonic computing system simulation by incorporating key photonic-specific characteristics, providing a flexible system to describe novel architectures, accurately modeling electronic interconnects, and enabling fast modeling for rapid design iteration. By providing a unified simulation environment for both photonic and electronic components, *LightSim* empowers designers to rapidly evaluate photonic system performance and conduct extensive design space exploration. To demonstrate the accuracy of *LightSim*, we validated our framework against prior works. Our modeling results show close agreement with their published data. Furthermore, we quantitatively validated our fast modeling capability through a rigorous runtime comparison against a baseline photonic simulator. To demonstrate the usefulness of *LightSim*, we conduct design space exploration at both the circuit and architectural levels. In the future, we will extend *LightSim* with more mapping strategies and support more novel AI workloads.

References

- [1] S. R. Ahmed, R. Baghdadi, M. Bernadskiy, N. Bowman, R. Braid, J. Carr, C. Chen, P. Ciccarella, M. Cole, J. Cooke *et al.*, “Universal photonic artificial intelligence acceleration,” *Nature*, vol. 640, no. 8058, pp. 368–374, 2025.
- [2] S. Hua, E. Divita, S. Yu, B. Peng, C. Roques-Carmes, Z. Su, Z. Chen, Y. Bai, J. Zou, Y. Zhu *et al.*, “An integrated large-scale photonic accelerator with ultralow latency,” *Nature*, vol. 640, no. 8058, pp. 361–367, 2025.
- [3] Z. Xu, T. Zhou, M. Ma, C. Deng, Q. Dai, and L. Fang, “Large-scale photonic chiplet taichi empowers 160-tops/w artificial general intelligence,” *Science*, vol. 384, no. 6692, pp. 202–209, 2024.
- [4] P. L. McMahon, “The physics of optical computing,” *Nature Reviews Physics*, vol. 5, no. 12, pp. 717–734, 2023.
- [5] M. Wei, K. Xu, B. Tang, J. Li, Y. Yun, P. Zhang, Y. Wu, K. Bao, K. Lei, Z. Chen *et al.*, “Monolithic back-end-of-line integration of phase change materials into foundry-manufactured silicon photonics,” *Nature Communications*, vol. 15, no. 1, p. 2786, 2024.
- [6] S. Lischke, D. Knoll, C. Mai, L. Zimmermann, A. Peczek, M. Kroh, A. Trusch, E. Krune, K. Voigt, and A. Mai, “High bandwidth, high responsivity waveguide-coupled germanium pin photodiode,” *Optics express*, vol. 23, no. 21, pp. 27 213–27 220, 2015.
- [7] W. Zhou, B. Dong, N. Farmakidis, X. Li, N. Youngblood, K. Huang, Y. He, C. David Wright, W. H. Pernice, and H. Bhaskaran, “In-memory photonic dot-product engine with electrically programmable weight banks,” *Nature Communications*, vol. 14, no. 1, p. 2887, 2023.
- [8] D. Pérez-López and L. Torrijos-Morán, “Large-scale photonic processors and their applications,” *npj Nanophotonics*, vol. 2, no. 1, p. 32, 2025.
- [9] M. A. Nahmias, T. F. De Lima, A. N. Tait, H.-T. Peng, B. J. Shastri, and P. R. Prucnal, “Photonic multiply-accumulate operations for neural networks,” *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, no. 1, pp. 1–18, 2019.
- [10] N. Farmakidis, B. Dong, and H. Bhaskaran, “Integrated photonic neuromorphic computing: opportunities and challenges,” *Nature Reviews Electrical Engineering*, vol. 1, no. 6, pp. 358–373, 2024.
- [11] M. Orlandin, A. Cem, V. Curri, A. Carena, F. Da Ros, and P. Bardella, “Thermal crosstalk effects in a silicon photonics neuromorphic network,” in *2023 International Conference on Numerical Simulation of Optoelectronic Devices (NUSOD)*. IEEE, 2023, pp. 43–44.

- [12] A. Alam, G. Wang, A. Liu, J. Klein, and D. McGuire, "Modeling thermal crosstalk in silicon photonics," in *Integrated Photonics Research, Silicon and Nanophotonics*. Optica Publishing Group, 2016, pp. ITu2B-4.
- [13] S. Fei, A. Eldebiky, G. L. Zhang, B. Li, and U. Schlichtmann, "An efficient general-purpose optical accelerator for neural networks," in *Proceedings of the 30th Asia and South Pacific Design Automation Conference*, 2025, pp. 1070–1076.
- [14] S. Bandyopadhyay, A. Sludds, S. Krastanov, R. Hamerly, N. Harris, D. Bunandar, M. Streshinsky, M. Hochberg, and D. Englund, "Single-chip photonic deep neural network with forward-only training," *Nature Photonics*, vol. 18, no. 12, pp. 1335–1343, 2024.
- [15] S. Banerjee, M. Nikdast, and K. Chakrabarty, "Modeling silicon-photonic neural networks under uncertainties," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021, pp. 98–101.
- [16] T. Andrulis, G. I. Chaudhry, V. M. Suriyakumar, J. S. Emer, and V. Sze, "Architecture-level modeling of photonic deep neural network accelerators," in *2024 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 2024, pp. 307–309.
- [17] D. Sturm and S. Moazeni, "Scalable coherent optical crossbar architecture using pcm for ai acceleration," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2023, pp. 1–6.
- [18] Z. Yin, M. Zhang, N. Gangi, R. Huang, J. Zhang, and J. Gu, "Symphony: A device-circuit-architecture cross-layer modeling and simulation framework for heterogeneous electronic-photonic ai system," in *2025 62nd ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2025, pp. 1–7.
- [19] L. Liu, S. R. Kari, X. Xin, N. Youngblood, Y. Zhang, and J. Yang, "Lightml: A photonic accelerator for efficient general purpose machine learning," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, 2025, pp. 18–33.
- [20] C. Demirkiran, F. Eris, G. Wang, J. Elmhurst, N. Moore, N. C. Harris, A. Basumallik, V. J. Reddi, A. Joshi, and D. Bunandar, "An electro-photonic system for accelerating deep neural networks," *ACM journal on emerging technologies in computing systems*, vol. 19, no. 4, pp. 1–31, 2023.
- [21] C. Cole, "Optical and electrical programmable computing energy use comparison," *Optics Express*, vol. 29, no. 9, pp. 13 153–13 170, 2021.
- [22] Z. Zhong, M. Yang, J. Lang, C. Williams, L. Kronman, A. Sludds, H. Esfahanizadeh, D. Englund, and M. Ghobadi, "Lightning: A reconfigurable photonic-electronic smartnic for fast and energy-efficient inference," in *Proceedings of the ACM SIGCOMM 2023 Conference*, 2023, pp. 452–472.
- [23] W. Liu, W. Liu, Y. Ye, Q. Lou, Y. Xie, and L. Jiang, "Holylight: A nanophotonic accelerator for deep learning in data centers," in *2019 design, automation & test in europe conference & exhibition (DATE)*. IEEE, 2019, pp. 1483–1488.
- [24] Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, S. Zhao, H. Larochelle, D. Englund *et al.*, "Deep learning with coherent nanophotonic circuits," *Nature photonics*, vol. 11, no. 7, pp. 441–446, 2017.
- [25] S. Ning, H. Zhu, C. Feng, J. Gu, Z. Jiang, Z. Ying, J. Midkiff, S. Jain, M. H. Hlaing, D. Z. Pan *et al.*, "Photonic-electronic integrated circuits for high-performance computing and ai accelerators," *Journal of Lightwave Technology*, 2024.
- [26] A. Shafiee, S. Banerjee, K. Chakrabarty, S. Pasricha, and M. Nikdast, "Analysis of optical loss and crosstalk noise in mzi-based coherent photonic neural networks," *Journal of lightwave technology*, vol. 42, no. 13, pp. 4598–4613, 2024.
- [27] B. Dong, F. Brücknerhoff-Plückelmann, L. Meyer, J. Dijkstra, I. Bente, D. Wendland, A. Varri, S. Aggarwal, N. Farmakidis, M. Wang *et al.*, "Partial coherence enhances parallelized photonic computing," *Nature*, vol. 632, no. 8023, pp. 55–62, 2024.
- [28] Y. Chen, M. Nazhamaiti, H. Xu, Y. Meng, T. Zhou, G. Li, J. Fan, Q. Wei, J. Wu, F. Qiao *et al.*, "All-analog photoelectronic chip for high-speed vision tasks," *Nature*, vol. 623, no. 7985, pp. 48–57, 2023.
- [29] H. Zhu, J. Gu, H. Wang, Z. Jiang, Z. Zhang, R. Tang, C. Feng, S. Han, R. T. Chen, and D. Z. Pan, "Lightening-transformer: A dynamically-operated optically-interconnected photonic transformer accelerator," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2024, pp. 686–703.
- [30] T. Andrulis, J. S. Emer, and V. Sze, "Cimloop: A flexible, accurate, and fast compute-in-memory modeling tool," in *2024 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 2024, pp. 10–23.
- [31] A. Samajdar, Y. Zhu, P. Whatmough, M. Mattina, and T. Krishna, "Scale-sim: Systolic cnn accelerator simulator," *arXiv preprint arXiv:1811.02883*, 2018.
- [32] C. Chen and A. Joshi, "Runtime management of laser power in silicon-photonic multibus noc architecture," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 19, no. 2, pp. 3 700 713–3 700 713, 2013.
- [33] M. S. Nezami, T. F. De Lima, M. Mitchell, S. Yu, J. Wang, S. Bilodeau, W. Zhang, M. Al-Qadasi, I. Taghavi, A. Tofini *et al.*, "Packaging and interconnect considerations in neuromorphic photonic neural accelerators," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 29, no. 2: Optical Computing, pp. 1–11, 2022.
- [34] S. Shekhar, W. Bogaerts, L. Chrostowski, J. E. Bowers, M. Hochberg, R. Soref, and B. J. Shastri, "Roadmapping the next generation of silicon photonics," *Nature Communications*, vol. 15, no. 1, p. 751, 2024.
- [35] S. Pasricha, "Optical computing for deep neural network acceleration: Foundations, recent developments, and emerging directions," *arXiv preprint arXiv:2407.21184*, 2024.

- [36] V. Bangari, B. A. Marquez, H. Miller, A. N. Tait, M. A. Nahmias, T. F. De Lima, H.-T. Peng, P. R. Prucnal, and B. J. Shastri, "Digital electronics and analog photonics for convolutional neural networks (deap-cnns)," *IEEE journal of selected topics in quantum electronics*, vol. 26, no. 1, pp. 1–13, 2019.
- [37] Z. Lu, K. Murray, H. Jayatilaka, and L. Chrostowski, "Michelson interferometer thermo-optic switch on soi with a 50- μ w power consumption," *IEEE Photonics Technology Letters*, vol. 27, no. 22, pp. 2319–2322, 2015.
- [38] H.-T. Peng, M. A. Nahmias, T. F. De Lima, A. N. Tait, and B. J. Shastri, "Neuromorphic photonic integrated circuits," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 24, no. 6, pp. 1–15, 2018.
- [39] A. N. Tait, A. X. Wu, T. F. De Lima, E. Zhou, B. J. Shastri, M. A. Nahmias, and P. R. Prucnal, "Microring weight banks," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 22, no. 6, pp. 312–325, 2016.
- [40] S. Pai, B. Bartlett, O. Solgaard, and D. A. Miller, "Matrix optimization on universal unitary photonic devices," *Physical review applied*, vol. 11, no. 6, p. 064044, 2019.
- [41] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, "Experimental realization of any discrete unitary operator," *Physical review letters*, vol. 73, no. 1, p. 58, 1994.
- [42] W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, and I. A. Walmsley, "Optimal design for universal multiport interferometers," *Optica*, vol. 3, no. 12, pp. 1460–1465, 2016.
- [43] S. Banerjee, M. Nikdast, S. Pasricha, and K. Chakrabarty, "Pruning coherent integrated photonic neural networks," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 29, no. 2: Optical Computing, pp. 1–13, 2023.
- [44] J. Lee, A. Lu, W. Li, and S. Yu, "Neurosim v1. 4: Extending technology support for digital compute-in-memory toward 1nm node," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 4, pp. 1733–1744, 2024.
- [45] C. Demirkiran, F. Eris, G. Wang, J. Elmhurst, N. Moore, N. C. Harris, A. Basumallik, V. J. Reddi, A. Joshi, and D. Bunandar, "An electro-photonics system for accelerating deep neural networks," *ACM journal on emerging technologies in computing systems*, vol. 19, no. 4, pp. 1–31, 2023.
- [46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [47] R. Baghdadi, M. Gould, S. Gupta, M. Tymchenko, D. Bunandar, C. Ramey, and N. C. Harris, "Dual slot-mode noem phase shifter," *Optics Express*, vol. 29, no. 12, pp. 19 113–19 119, 2021.
- [48] A. Marchisio, F. Da Ros, V. Curri, A. Carena, and P. Bardella, "Comprehensive model of mzi-based circuits for photonic computing applications," *Communications Physics*, vol. 8, no. 1, p. 277, 2025.
- [49] S. Pai, Z. Sun, T. W. Hughes, T. Park, B. Bartlett, I. A. Williamson, M. Minkov, M. Milanizadeh, N. Abebe, F. Morichetti *et al.*, "Experimentally realized in situ backpropagation for deep learning in photonic neural networks," *Science*, vol. 380, no. 6643, pp. 398–404, 2023.
- [50] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 2002.
- [51] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3642–3649.
- [52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [53] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.
- [54] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.