PIC^{2}O-Sim: A Physics-Inspired Causality-Aware Dynamic Convolutional Neural Operator for Ultra-Fast Photonic Device Time-Domain Simulation

Pingchuan Ma,[1] Haoyu Yang,[2] Zhengqi Gao,[3] Duane S. Boning,[3] and Jiaqi Gu[1]

[1)] *School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281, USA*

[2)] *11001 Lakeline Blvd Suite #100 Bldg. 2, NVIDIA Inc, Austin, TX 78717, USA*

[3)] *Department of Electrical Engineering and Computer Science , Massachusetts Institute of Technology, Boston, MA 02139, USA*

(*Electronic mail: jiaqigu@asu.edu.)

(Dated: 11 February 2025)

Optical simulation plays an important role in photonic hardware design flow. The finite-difference time-domain (FDTD) method is widely adopted to solve time-domain Maxwell equations. However, FDTD is known for its prohibitive runtime cost as it iteratively solves Maxwell equations and takes minutes to hours to simulate a single device. Recently, AI has been applied to realize orders-of-magnitude speedup in partial differential equation (PDE) solving. However, AI-based FDTD solvers for photonic devices have not been clearly formulated. Directly applying off-the-shelf models to predict the optical field dynamics shows unsatisfying fidelity and efficiency since the model primitives are agnostic to the unique physical properties of Maxwell equations and lack algorithmic customization.

In this work, we thoroughly investigate the synergy between neural operator designs and the physical property of Maxwell equations and introduce a physics-inspired AI-based FDTD prediction framework `PIC²O-Sim`. `PIC²O-Sim` features a causality-aware dynamic convolutional neural operator as its backbone model that honors the space-time causality constraints via careful receptive field configuration and explicitly captures the permittivity-dependent light propagation behavior via an efficient dynamic convolution operator. Meanwhile, we explore the trade-offs among prediction scalability, fidelity, and efficiency via a multi-stage partitioned time-bundling technique in autoregressive prediction. Multiple key techniques have been introduced to mitigate iterative error accumulation while maintaining efficiency advantages during autoregressive field prediction. Extensive evaluations on three challenging photonic device simulation tasks have shown the superiority of our `PIC²O-Sim` method, showing 51.2% lower roll-out prediction error, 23.5 times fewer parameters than state-of-the-art neural operators, providing 133-310$\times$ or 31-89$\times$ higher simulation speed than an open-source single-process or 8-process parallel FDTD numerical solver.

I. INTRODUCTION

Photonics has demonstrated significant potential for high-performance, energy-efficient computing, communication, and sensing, owing to the fast propagation speed, high bandwidth, and high degree of freedom of photons. Finite-difference time-domain (FDTD) simulation is a widely used numerical method for simulating the spectral response of photonic structures. This method solves the time-dependent Maxwell equations on a discrete mesh to simulate electromagnetic (EM) wave propagation. However, FDTD simulations are highly time-consuming because they require iterative updates of the EM field distribution across a fine space-time grid to ensure convergence. The computational complexity of 2D FDTD is $\mathcal{O}(N^2T)$, where N represents the spatial grid dimensions and T the simulation timesteps. For instance, simulating a small 2D photonic device measuring $28\mu m \times 10.5\mu m$ over a 1.3 ps timespan can take around one minute on powerful CPUs, as illustrated in Fig. 1(a). Simulating larger devices or circuits with intricate structures and resonant responses requires hours of computation, significantly slowing down the design process and making iterative simulation-in-the-loop optimization prohibitively expensive. This highlights the pressing need for an ultra-fast surrogate simulation model that can provide rapid feedback during the early stages of design.

There is a growing trend toward using AI to accelerate PDE solving for physical simulations. Physics-informed neural networks (PINNs)[3] and neural operators (NOs) have demonstrated the ability to learn nonlinear, high-dimensional mappings from PDE observations to solutions, achieving orders-of-magnitude speed improvements over traditional numerical solvers. AI-based surrogate models have been successfully applied in a variety of scientific domains, including fluid dynamics and field simulations[1,4–9], weather forecasting[10], and hardware simulation and inverse design[11–15]. In the realm of AI for optics, previous work has introduced physics-augmented and data-driven neural networks[11,12,14] for finite-difference frequency-domain (FDFD) optical simulations. Some research has also explored time-domain field prediction, replacing FDTD solvers with PINNs[16,17], and physics-driven recurrent neural networks (RNNs)[18] that embed iterative FDTD update rules into the model architecture to directly simulate light propagation.

The potential of AI to solve time-domain Maxwell equations and directly generate optical field dynamics as a video remains underexplored. In this work, we address several key questions related to AI-based optical FDTD tasks: (1) **What neural operator architecture honors the physics constraints of time-domain Maxwell equations?** State-of-the-art neural operators designed for flow dynamics or EM wave simulations often use global-view operators, such as multi-scale convolution[19,20] and Fourier-domain operators[1],

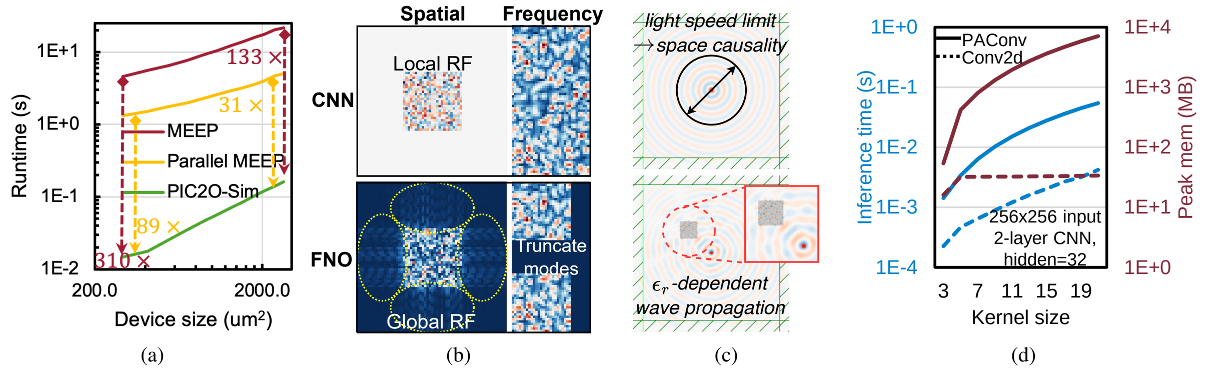


FIG. 1: (a) Our ML-based model shows orders-of-magnitude speedup over FDTD solvers. (b) Global-view FNO[1] with truncated modes cannot learn local convolution. (c) (*Top*) Due to light speed limitation, space causality implies local receptive fields; (*Bottom*) Wave propagation depends on local permittivity distributions. (d) Expanding convolutional kernel sizes for long-term prediction is not scalable in runtime and memory, especially for dynamic convolution PAConv[2].

to capture long-range correlations in the solution domain, as shown in Fig. 1(b). However, this approach can violate the space-time causality constraints of Maxwell's equations, where the speed of light limits the maximum distance over which information can propagate, as illustrated in Fig. 1(c) (*Top*). Moreover, Fourier operators, with truncated frequency modes, are not ideal for representing local interactions, making convolution with a limited receptive field a more suitable candidate. (2) **How should PDE variables be effectively represented in the model architecture?** Many existing approaches treat all PDE variables as multi-channel tensors fed into the neural operator, without considering their distinct physical properties. However, Maxwell's equations imply a permittivity-dependent wave propagation behavior, as shown in Fig. 1(c)(*Bottom*), which cannot be adequately captured by static matrix multiplication or convolution. (3) **What are the trade-offs among scalability, long-term prediction fidelity, and speedup?** Since FDTD requires generating a video of field dynamics, it is essential to determine how to formulate this video generation task in a scalable, efficient, and high-fidelity manner. A straightforward approach is to predict each frame in the next timestep with high fidelity. However, this sacrifices speed benefits due to the large number of iterations and leads to error accumulation over time, resulting in significant rollout errors. On the other hand, predicting the entire video in one shot avoids error accumulation but greatly increases the learning difficulty and is not scalable for long simulation timespans, as computing and memory costs grow quadratically with larger receptive fields (kernel sizes), as depicted in Fig. 1(d).

Building on the unique characteristics of optical FDTD simulations, we address the three key unresolved questions and propose a physics-inspired, data-driven ML-based photonic FDTD simulation framework, dubbed `PIC`^{2}`O-Sim`. Inspired by the space-time locality and permittivity dependency of light propagation, we draw an analogy between optical FDTD simulations and dynamic convolution. The `PIC`^{2}`O-Sim` framework employs a dynamic convolutional neural operator model to predict high-quality light fields in a time-bundled manner. To generate long timespan light dynamics, we use an autoregressive, multi-iteration approach to maintain scalability and speed advantages while minimizing the long-term prediction error.

The main contributions of this work are threefold:

- We thoroughly investigate the unique properties and unresolved challenges in AI-based time-domain Maxwell PDE solving and introduce `PIC`^{2}`O-Sim`, a physics-inspired, causality-constrained FDTD simulation framework for photonic devices that balances scalability, fidelity, and speed.

- We identify the analogy between FDTD and dynamic convolutions, designing a convolutional neural operator model with causality-aware receptive fields and dilated, position-adaptive dynamic convolutions[2,21,22] to predict permittivity-aware light propagation.

- We propose an autoregressive framework for scalable long-term video generation, mitigating temporal error accumulation through cross-iteration error correction techniques.

- Extensive evaluations demonstrate superior prediction fidelity and efficiency across three types of complex photonic devices, achieving **51.2%** lower prediction errors, **23.5**× higher parameter efficiency, and hla speedup of **133**∼**310**× compared to the single-core open-source FDTD solver Meep and **31**∼**89**× compared to 8-core parallel Meep. For details regarding the speed-up, please refer to Appendix A 3.

II. BACKGROUND: AI FOR PDE SOLVING

Recently, scientific machine learning algorithms have been widely explored to accelerate the solution of fundamental PDE problems, offering orders-of-magnitude speed improvements. PINNs and data-driven neural operators represent two branches of research where physics is either added as a hard constraint or ignored to remove domain knowledge requirements. In the field of AI for optics, physics-informed models, e.g., WaveTorch[23], directly embed the PDE updating rules in the recurrent neural network (RNN) cells to leverage the GPU-accelerated inference engine for faster iteration. With a small enough spatiotemporal resolution, these methods have a theoretical guarantee on the solved fields, while their speedup is rather limited due to a large number of iterations. Also, oversimplified equations in the RNN cells make it hard to match the golden results from commercial tools.

Physics-augmented models, such as MaxwellNet[12] and WaveYNet[11], use a standard U-Net architecture and incorporate Maxwell residual loss into the training objective to learn optical fields that respect physical constraints. More recently, SineNet[19] was introduced to mitigate temporal misalignment caused by skip connections between multi-scale features in U-Net, using a cascading U-Net structure.

However, the synergy between model architecture and the underlying physical constraints of Maxwell equations remains under-explored. SoTA neural operators may not be well-suited for optical FDTD simulations due to the unique properties of Maxwell's equations. Besides, prior work has largely focused on single-iteration prediction tasks without handling the error accumulation effects that arise in autoregressive prediction.

III. PROPOSED `PIC²O-SIM` FRAMEWORK

A. Understanding the FDTD Simulation for Light Propagation in Photonic Devices

First, we formulate the FDTD method for photonic device simulation. FDTD starts by injecting an eigenmode light source into the device and simulating the light propagation via sequential time-stepping. To capture the response across multiple wavelengths in one shot, The incident source typically has a Gaussian-shaped envelope centered at frequency f_c with a frequency width of f_w, enabling broadband simulation by covering a wide range of wavelengths. FDTD method discretizes the time-domain Maxwell equation and iteratively updates the electric fields. For the details of the electric field updating rule, please refer to Appendix A 1

Considerable computational complexity of FDTD. FDTD is computationally intensive, as its convergence relies on fine-grained space-time resolutions (Δ_t, Δ_x, Δ_y) to accurately capture light-speed signal propagation, e.g., a typical timestep is $\Delta_t = 0.167fs$, and a space resolution is around 1/15 of the wavelength, resulting in high computational complexity of $\mathcal{O}(\frac{N_xN_yT}{\Delta_x\Delta_y\Delta_t})$, where N_x, N_y are the domain dimensions and T is the simulation timespan. To further improve convergence, electric and magnetic fields are updated alternately on an interleaved 2D Yee grid, increasing the computational cost by a factor of four. Consequently, a fast prediction method capable of bypassing numerous FDTD time-marching steps and directly reconstructing the spatiotemporal fields would dramatically accelerate time-domain photonic device simulations.

Causality-constrained space-time locality. In near-future light field prediction tasks, wave propagation is constrained to a local spatial region due to the finite speed of light, as illustrated by the light cone in Fig. 2. This means that any fields outside the light cone are non-causal to the central field, implying a limited theoretical spatial receptive field (RF) for any prediction model. A model with an RF that is too small

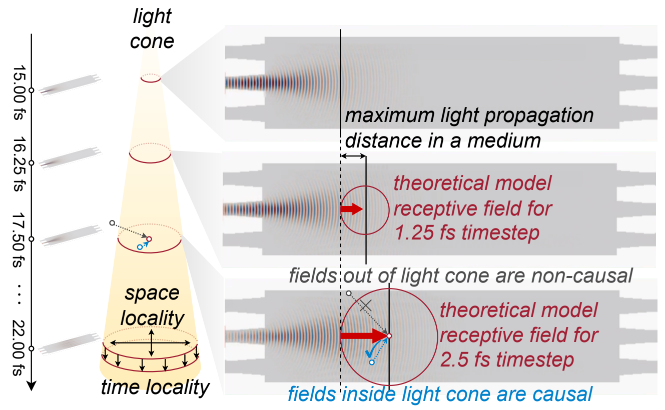

light cone
15.00 fs
16.25 fs
17.50 fs
...
22.00 fs
space locality
time locality
maximum light propagation distance in a medium
theoretical model receptive field for 1.25 fs timestep
fields out of light cone are non-causal
theoretical model receptive field for 2.5 fs timestep
fields inside light cone are causal

FIG. 2: Illustration of the causality-constrained space-time locality and the theoretical receptive field.

lacks sufficient information to accurately reconstruct the central field. Conversely, a model with overly large RF or global views extending beyond the light cone risks learning non-physical mappings by incorporating irrelevant, non-causal information. Besides spatial locality limited by light speed, the light field also exhibits temporal locality Based on Maxwell equations, the field distribution at timestep t solely depends on the electromagnetic wave in the previous timestep $t-1$. This indicates that capturing a long temporal context, as is often preferred in other sequence or time-series modeling tasks, is unnecessary in this case. Therefore, it is crucial to design a model with a carefully selected space-time receptive field that respects causality.

Permittivity-dependent light propagation. The causality-constrained space-time locality suggests that a convolution-based neural operator with a carefully selected spatial receptive field is well-suited for propagating light waves from neighboring causal regions to the central location. However, static convolution operations may not be sufficient to model the complex light-matter interaction in the photonic device as the local wave propagation behavior at coordinate $[m,n]$ is a function of material permittivity $\varepsilon[m,n]$ at that specific location, as illustrated in Fig. 1(c). Such a property implies that the convolutional filter, which simulates the wave propagation mechanism, should dynamically adjust its values based on the local material permittivity. This insight leads us to propose a dynamic permittivity encoding in the convolutional filters.

B. Proposed `PIC²O-Sim` Framework

1. Framework Overview

Figure 3 illustrates our autoregressive time-bundled `PIC²O-Sim` framework. To potentially handle prediction over a long time horizon, `PIC²O-Sim` autoregressively predicts the future optical field dynamics based on previous fields E_{in} and corresponding light source J. Time-bundling is employed to predict multiple timesteps/frames of fields at each iteration.

`PIC²O-Sim` formulates a single-iteration FDTD prediction task as a functional mapping from given initial condition $\mathscr{A} \in \mathbb{R}^{\Omega\times d_a}$, including previous light fields, light source, and device permittivity distribution, to the electrical field solution $\mathscr{U} \in \mathbb{R}^{\Omega\times d_u}$. For details on why we chose to predict the fields

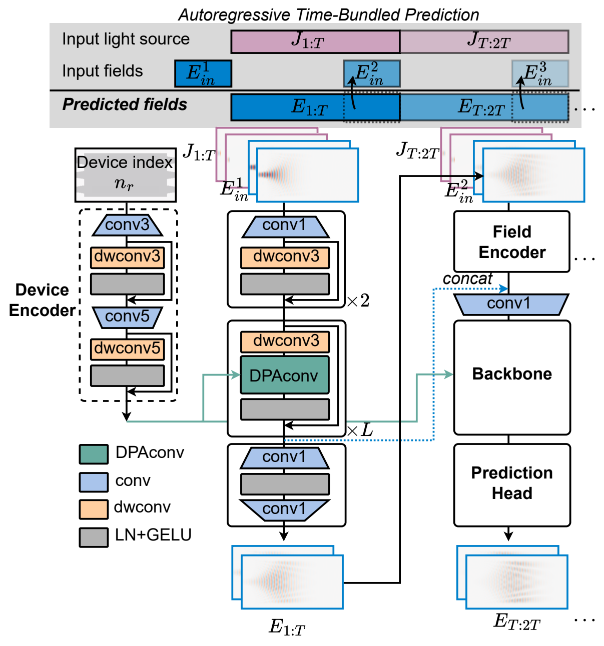


FIG. 3: PIC^{2}O-Sim framework overview.

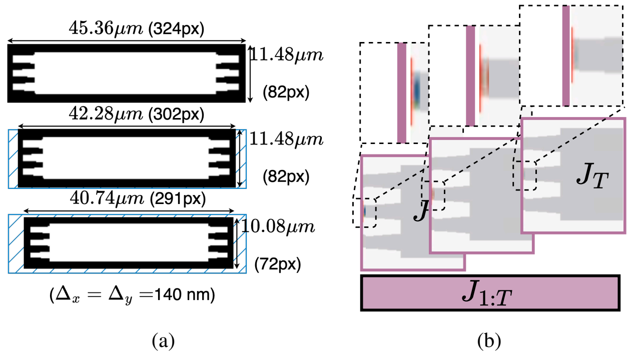


FIG. 4: (a) Devices in a mini-batch are replicate-padded to the same size with a fixed spatial resolution. (b) Light source representation.

instead of specific quantities, such as the S-parameter at a particular port, please refer to Appendix A 2. Ω is the 2-D field domain with a size of $[N_x, N_y]$. PIC^{2}O-Sim takes E_{in} from previous T_{in} timesteps and $J_{1:T}$ in the future T timesteps as input and passes through a convolutional neural operator Ψ_θ to predict the future T-step light fields $E_{1:T}$. The neural operator Ψ_θ consists of a field encoder, a dynamic convolutional backbone, and a prediction head. Permittivities/refractive indices of the device are explicitly encoded by a dedicated device encoder and fed into all dynamic position-adaptive convolutional layers in the model backbone, guiding the light propagation process.

2. *Resolution-Preserved Shift-Invariant Domain Discretization*

To handle varying device sizes, prior work[14] employed a scale-adaptive domain discretization approach that resizes devices of different physical dimensions to a fixed image size, as Fourier neural operators require a function mapping in a fixed domain Ω. In contrast, PIC^{2}O-Sim preserves the spatial resolution (Δ_x, Δ_y) and uses padding, if necessary, to maintain shift-invariance. This design offers two key advantages: (1) **Shift-invariant is more scalable and generalizable than resolution-invariant discretization in our problem**. The prediction model is trained on ground-truth fields simulated with sufficiently high resolution to ensure FDTD accuracy, typically where a wavelength contains 15-20 pixels. Increasing the resolution beyond this point provides no practical benefit. Moreover, downsizing introduces the challenge of handling large, unseen devices, where excessive downsampling leads to significant information loss. By leveraging the shift-invariant property of convolution, our model can predict light propagation across arbitrarily large domains without the loss of information caused by downsampling, as long as the spatial resolution remains consistent. (2) **Training efficiency benefit**. To avoid downsampling-induced information loss, the domain-adaptive method tends to scale all devices to a large image size, which causes high costs during training and inference. In contrast, PIC^{2}O-Sim scales all devices to the same pixel resolution, e.g., $\Delta_x = \Delta_y = 140nm$ and pads devices to the maximum image size only in this mini-batch for parallel batched processing as shown in Fig. 4(a). In this way, we can avoid information loss due to the downsampling of large devices and improve speeds on small devices.

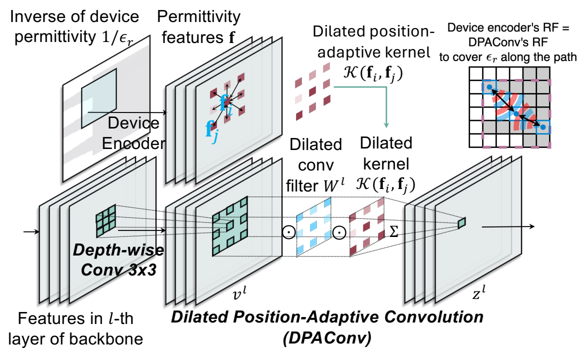
3. *Model Input/Output Definition: Permittivity, Input Fields, and Sources*

At one iteration, the model takes the permittivity ϵ_r and T_{in}-frame input fields $E_{in} \in \mathbb{R}^{T_{in}\times H\times W}$before the target prediction timestep, and T-frame light sources $J_{1:T}$ as the PDE variables. For 2D simulation, the line-shaped Gaussian eigenmode source $J_{1:T}$ is injected at the port center. The existence of a source in the system is the fundamental difference and also the challenging part compared to other source-free PDE systems. The right-hand side of the Maxwell equation is not zero but a time-varying function $J(t)$, such that each frame of field is potentially impacted by all previous injected light sources. We formulate the variable light source within the prediction time horizon as a T-frame video, where all fields are masked to zero except the line-shaped region of light source at the input port, as shown in Fig. 4(b). In this way, the FDTD prediction task is translated to a masked video restoration task, given that previous frames and the video patches at the source location are unmasked hints.

C. Efficient Physics-Inspired Dynamic Convolutional Neural Operator Architecture

Building on the space-time causality principles outlined in Section III A, our PIC^{2}O-Sim model, Ψ_θ, employs local-view convolutions to constrain the receptive field. The detailed architecture is described below.

Convolutional field encoder. PIC^{2}O-Sim starts with a convolutional encoder to project the previous light fields and incident light sources to a D-dimensional latent space: $a_E^\dagger(\boldsymbol{r}) \to v_0(\boldsymbol{r}), \forall \boldsymbol{r} \in \Omega$, where $a_E^\dagger = \{E_{in}; J_{1:T}\} \in \mathbb{R}^{(T_{in}+T)\times M\times N}$ and



Inverse of device permittivity $1/\epsilon_r$
Permittivity features $\mathbf{f}$
Dilated position-adaptive kernel $\mathcal{K}(\mathbf{f}_i,\mathbf{f}_j)$
Device encoder's RF = DPAConv's RF to cover ϵ_r along the path
Device Encoder
Dilated conv filter W^l
Dilated kernel $\mathcal{K}(\mathbf{f}_i,\mathbf{f}_j)$
Depth-wise Conv 3x3
v^l
z^l
Features in l-th layer of backbone
Dilated Position-Adaptive Convolution (DPAConv)

FIG. 5: Illustration of the proposed dilated position-adaptive convolution (DPAConv).

$v_0(\boldsymbol{r}) \in \mathbb{R}^{D\times M\times N}$. The encoder has two blocks, each containing a point-wise convolution followed by a residual block of 3×3 depthwise convolution, layer normalization, and the GELU activation function.

Causality-constrained permittivity-aware convolutional backbone. The backbone of PIC^{2}O-Sim consists of L-layer residual blocks, each including a depthwise convolution, dilated position-adaptive convolution (DPAConv), layer normalization, and GELU activation. A dedicated convolutional device encoder shown in Fig. 3 takes the inverse of permittivity map $1/\epsilon_r(r)$ as input and extracts a shared local geometry information $\boldsymbol{f}(\epsilon_r^{-1})$ for guiding all L DPAConv layers in the backbone to dynamically adapt their kernels through which we explicitly model the permittivity-dependent wave propagation dynamics.

Figure 5 illustrates a $K\times K$ DPAConv module with dynamic permittivity-adaptive kernels. Each DPAConv operation within a size-K window $\Omega(i)$ at pixel position i is formulated as $z^l(i) = \sum_{j\in\Omega(i)}\left(\mathscr{K}(\mathbf{f}_i,\mathbf{f}_j)(\epsilon_r)\mathbf{W}^T(j)\right)\cdot v^l(j)$. Inspired by dynamic convolution (PAConv)[2], the convolutional filter weights applied to a sliding window on the feature map is the Hadamard product of a statically-learned convolutional filter $W\in\mathbb{R}^{D\times D\times K\times K}$ and a dynamic permittivity-adaptive kernel $\mathscr{K}(\mathbf{f}_i,\mathbf{f}_j)(\epsilon_r)\in\mathbb{R}$, shown in Fig. 5. The dynamic kernel $\mathscr{K}$ projects the permittivity features $\mathbf{f}$ into high-dimensional space via a Gaussian kernel function $\mathscr{K}(\mathbf{f}_i,\mathbf{f}_j)(\epsilon_r) = \exp\left(-\frac{1}{2}(\mathbf{f}_i-\mathbf{f}_j)^\top(\mathbf{f}_i-\mathbf{f}_j)\right)$, which helps the model understand light-matter interaction and learn how light wave propagates dynamically through a path with heterogeneous material permittivities.

This dynamic convolution demonstrates strong modeling capabilities for capturing wave propagation principles. However, the standard PAConv[2] presents practical challenges due to its substantial memory and runtime costs, especially during training. As the prediction frames T increase, the required receptive field and convolutional kernel size increase linearly. When a large-kernel convolution requires dynamic position-specific kernels, the memory cost is bottlenecked by the largest intermediate tensor $\mathscr{K}\odot W\in\mathbb{R}^{D\times H\times W\times K\times K}$, e.g., when $D=96$, $H=W=256$, and $K=21$, a single tensor can exceed 10 GB. To alleviate the computational and memory burden, we modify this approach by introducing a dilated position-adaptive convolution (DPAConv), preceded by a depthwise convolution to aggregate local features and prevent information loss from dilation. The kernel size K is a crucial design parameter for DPAConv, as it determines the model's receptive field. As shown in Figure 6, insufficient receptive field size significantly degrades performance. Empirically, we recommend using **a receptive field that is 30 pixels larger than the theoretical value** for optimal results. For detailed kernel size selection, please see Appendix A 5

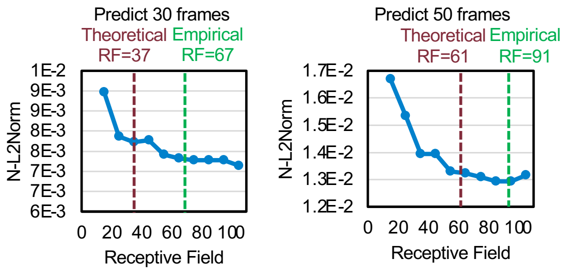


FIG. 6: 30 pixels larger RF provides optimum fidelity

Prediction head. At the end of the model, we simply use two point-wise convolutions with layer normalization and GELU in between to project it back to the required prediction frames.

D. Autoregressive Prediction with Multi-Stage Partitioned Time-Bundling

Time-bundling significantly accelerates predictions by generating multiple frames in one step, reducing the number of iterations needed during autoregression[24]. However, we argue that **bundling too many timesteps can negatively impact scalability and prediction fidelity**. (1) Scalability: As the number of output frames increases, the required receptive field expands linearly, resulting in a *quadratic increase in parameter count and computation cost*, along with *greater optimization difficulty* due to larger convolutional kernels. Moreover, predicting more frames demands greater model capacity, meaning the hidden dimension D must also grow. Therefore, bundling too many frames per iteration is *not scalable* for efficient model inference. (2) Prediction Fidelity: Time-bundling essentially shares the learned encoder and backbone between different timesteps for efficiency consideration and inevitably correlates different output frames. Since the correlation between frames gets reduced with a longer timespan, such sharing can deteriorate the overall prediction fidelity. Besides, bundling too many timesteps in a one-shot prediction also *breaks the temporal causality* in light waves, as output field $E[t]$ strictly should not see any information from future sources $J_{t+1:T}$. Hence, reducing bundled timesteps by partitioning the frames into multiple stages can largely relax the scalability and causality issue while still benefiting from the speed advantages in parallel time-bundled prediction.

Figure 3 illustrates our multi-stage time-bundling approach for autoregressive prediction. The total predicted fields $E_{1:NT}$ consist of NT frames, which we partition into N stages. In standard autoregressive prediction, the model Ψ_θ is trained in a single iteration and recurrently generates future fields based on predictions from the previous stage, encouraging the model to learn consistent mapping functions across timesteps. However, this approach assumes ideal prediction fidelity at each stage. In reality, the first-stage prediction $E_{1:T}$ may deviate from the ground truth $E^*_{1:T}$, leading to *accumulated roll-out errors* as the inputs for later iterations suffer from *distribution shifts* due to earlier prediction inaccuracies. To address this, we propose two methods to mitigate temporally accumulated errors.

Stage-dedicated prediction models to mitigate distribution shift. We use N independent submodels $\Psi_{\theta_1}, \cdots, \Psi_{\theta_N}$ in the partitioned N stages to mitigate the distribution shift issue. Stage 2 to stage N will learn how to align the non-ideal input fields from their previous stage to the ground-truth fields. In a later evaluation section, we will show the advantages of independent models in reducing prediction errors while maintaining high parameters and runtime efficiency.

Cross-stage hidden state propagation to facilitate error mitigation. While the time locality of Maxwell's equations suggests that light fields depend only on recent past fields and future sources, the non-ideal field predictions from Ψ_{θ_i} often lack sufficient information for $\Psi_{\theta_{i+1}}$ to fully correct the errors from the i-th stage. Drawing inspiration from the State-Space Model, we introduce an additional information path by propagating the encoded hidden states from the backbone of the i-th stage to the input of the next stage backbone. A pointwise convolution is used as a lightweight adaptor to compress the concatenated hidden states back to dimension D.

Light field normalization. Normalization is critical for model convergence and generalization. Empirically, we find that using the maximum field intensity to normalize the input fields and source provides the best roll-out error. For a detailed ablation study, please refer to Appendix A 5.

IV. EVALUATION

A. Evaluation Setup

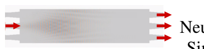
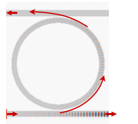
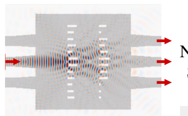
Benchmarks. We evaluate different methods on three representative and challenging photonic device types, including tunable multi-mode interference (MMI) with complicated interference patterns, micro-ring resonator (MRR) with sensitive coupling and resonance effects, and Metaline with highly discrete permittivity distributions and fine-grained structures. Those practical devices pose significant challenges and have not been evaluated in the literature. We use the open-source FDTD software package MEEP[25] to generate the simulation videos. All videos are resized to have a spatial resolution of ($\Delta_x = \Delta_y = 140nm$, $\Delta_t = 1$ fs). Details are in the Appendix V.

Training settings and evaluation metrics. Since all frames have the same importance in FDTD, we use averaged per-frame normalized L2-Norm as the training loss function and also evaluation metric, i.e., N-L2Norm $= \frac{1}{T}\sum_{t=1}^{T} \|\Psi_\theta(E_{in}, \epsilon_r, J_{1:T})[t] - E^*[t]\|_2 / \|E^*[t]\|_2$. We use frames per second (FPS) to evaluate the prediction speed. Detailed training settings can be found in Appendix A 6.

B. Main Result

We compare seven models in Table I, including: (1) *global-view Fourier-domain neural operators*: FNO[1], its factorized variant F-FNO[4], the state-of-the-art optical FDFD neural network surrogate NeurOLight[14], and a Koopman neural operator (KNO) designed for time-marching in the linear Koopman space[26]; and (2) *local-view convolution-based neural operators*: a 16-layer SimpleCNN with static 2D convolutions, SineNet[19], which uses a cascaded multi-stage U-Net for temporal modeling, and our proposed dynamic convolutional neural operator, PIC^{2}O-Sim. For fairness, all Fourier-domain models are configured to use full modes, allowing them to learn local spatial operations. The video simulations span 160 fs (i.e., 160 frames), with 10 frames of past input fields ($T_{in} = 10$) serving as initial conditions. Detailed model configurations can be found in Appendix A 7. It is noteworthy that while PIC^{2}O-Sim incorporates a strong inductive bias through the use of dynamic convolutions and controlled receptive fields inspired by time-domain light propagation principles, it remains a purely data-driven model. This is because no Maxwell residual loss is utilized during training. In other words, PIC^{2}O-Sim is a physics-agnostic model. From this perspective, the comparison between PIC^{2}O-Sim and the strong baseline models is fair.

TABLE I: Compare different models on 3 benchmarks in terms of #params, inference speed (FPS), training, and test error (N-L2Norm). The predicted light fields have 160 frames.

Dataset	Model	#Params ↓	FPS ↑	Train error ↓	Test error ↓
MMI 	FNO[1]	340M	8147	0.035	0.122
	F-FNO[4]	4.5M	4359	0.039	0.070
	KNO[26]	171.8M	1251	0.188	0.193
	NeurOLight[14]	2.2M	8180	0.157	0.140
	SimpleCNN	3.8M	17524	0.066	0.075
	SineNet[19]	38M	3414	0.071	0.085
	PIC^{2}O-Sim	2.4M	15701	0.042	0.052
MRR 	FNO[1]	340M	8147	0.033	0.423
	F-FNO[4]	4.5M	4376	0.028	0.138
	KNO[26]	171.8M	1252	0.138	0.179
	NeurOLight[14]	2.2M	8190	0.102	0.151
	SimpleCNN	7.3M	7646	0.038	0.088
	SineNet[19]	38M	3282	0.044	0.109
	PIC^{2}O-Sim	4.4M	1906	0.025	0.085
Metaline 	FNO[1]	146.4M	20047	0.062	0.173
	F-FNO[4]	3.3M	9713	0.053	0.089
	KNO[26]	74.6M	2889	0.278	0.268
	NeurOLight[14]	1.6M	18413	0.213	0.185
	SimpleCNN	3.8M	26920	0.112	0.117
	SineNet[19]	30M	4484	0.114	0.122
	PIC^{2}O-Sim	2.4M	7348	0.077	0.086
Avg Improv. w/ FNO/SimpleCNN		95.74%	92.92%	53.33%	51.16%
Avg Improv. w/o FNO/SimpleCNN		40.47%	156.5%	43.47%	42.89%

Compared to these baselines, on average, our PIC^{2}O-Sim achieved 51.2% less normalized L2-norm error with 95.7% fewer parameters.

Result in tunable MMI. On tunable MMI, PIC^{2}O-Sim achieves the most accurate prediction result with only 2.4M parameters, less than 1% of full-mode FNO. Without the inductive bias of the local spatial receptive field, FNO requires full frequency modes to learn a causality-aware local window. This comes at the cost of hundreds of millions of parameters and optimization challenges in large kernel learning. SineNet, although it is a U-Net-based model, and the receptive field is too large due to its multi-scale feature fusion that introduces too many irrelevant features, shows 60% more error compared to PIC^{2}O-Sim. Both SimpleCNN and PIC^{2}O-Sim outperform the Fourier-domain and U-Net-based models, which we attribute to their causality inductive bias. However, Sim-

TABLE II: Compare different task partitioning when predicting 160 frames. Kernel sizes (*KS*) are adjusted to match the suitable receptive field for the predicted frames (T) per step. Partitioning into 2 stages gives the best results. Dilation is set to 4 by default

#stages	T per stage	KS	#Params ↓	FPS ↑	Train error ↓	Test error ↓
1	160	29	3.5M	13793	5.44e-2	6.26e-2
2	80	17	2.3M	6865	4.20e-2	5.13e-2
4	40	11	1.9M	5056	4.70e-2	5.38e-2
8	20	9	3.7M	2603	8.14e-2	8.71e-2

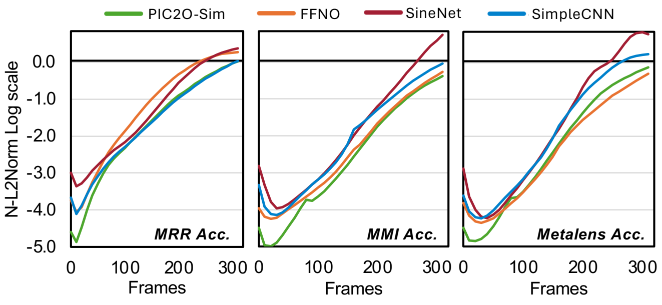


FIG. 7: In a 320-frame prediction task across three benchmarks, $\texttt{PIC}^2\texttt{O-Sim}$ achieves the lowest cumulative error compared to all other models.

pleCNN, which uses content-agnostic static kernels, demonstrates slightly lower fidelity than $\texttt{PIC}^2\texttt{O-Sim}$, likely due to the homogeneous permittivity in most parts of the MMI.

Result in MRR. Compared with MMI, MRR is more difficult as it includes light coupling, feedback loop, and resonance. Considering that the ϵ_r in MMR is lower than that in MMI, which means faster light propagation, we adjusted the receptive field for the CNN-based networks accordingly. Still, there remains a performance gap of approximately 50% between the Fourier-domain models, U-Net-based models, and CNN-based models. Among the CNN-based models, $\texttt{PIC}^2\texttt{O-Sim}$ achieved the lowest error while using one of the fewest parameters.

Result in Metaline. Compared to MMI, Metaline features highly discrete permittivity distributions with strong scattering effects, making dynamic convolution essential for accurately capturing wave propagation behavior. Consequently, we observe a significantly higher prediction error from static convolutional networks.

On average, our $\texttt{PIC}^2\texttt{O-Sim}$ achieved 51.2% less test error with 95.7% fewer parameters, showing the advantages of our dedicated device encoder and DPAConv-based backbone.

C. Ablation Study and Discussion

Input field frames T_{in} and dilation rate s. As a key hyperparameter, we select 10 frames of input fields E_{in}, and we choose a dilation rate of 4 to balance fidelity, speed, and parameter efficiency. For details, please refer to Appendix A 5.

Multi-stage partitioning in time-bundled prediction. As shown in Table II, we explored various multi-stage partitioning strategies for time-bundled prediction across 160 frames. Each stage utilizes an independent Ψ_θ model. In the single-stage approach, the model attempts to predict all 160 frames in one shot, but this lacks sufficient model capacity to maintain high fidelity and is not scalable for long-range predictions.

We found that bi-partitioning strikes the best balance between accuracy and parameter efficiency for generating 160 frames. Too many partitions increase the number of iterations, leading to significant distribution shifts and error accumulation. Further ablation studies are provided in Appendix A 5.

Cross-stage model sharing and hidden state propagation. Table III compares different methods of partition, highlighting that the best strategy involves passing historical information, along with the prediction result, between adjacent independent submodels, from $\Psi_{\theta_{i-1}}$ to Ψ_{θ_i}. Simply iterating twice, CNN will suffer from a severe distribution shift. By jointly optimizing two weight-sharing CNNs, the model is aware of the error accumulation, thus showing better roll-out errors. Once we relax them to two independent CNNs, combined with the hidden state that passes more information to the downstream submodels, we achieve the lowest error in 160 frames of prediction.

TABLE III: Stage-dedicated models with hidden state propagation give the best 160-frame fidelity.

	Hidden state	Out frames	#Iter	Single test err	Roll-out test err
A single CNN	N/A	80	2	2.13e-2	7.30e-2
2 weight-sharing CNNs	×	80+80	1	5.98e-2	5.98e-2
2 independent CNNs	✓	80+80	1	4.86e-2	4.86e-2

To further demonstrate $\texttt{PIC}^2\texttt{O-Sim}$'s superior performance in mitigating error accumulation during auto-regressive prediction, we evaluated the average normalized L2-norm per frame over a 320-frame prediction task across three benchmark datasets. As illustrated in Fig. 7, $\texttt{PIC}^2\texttt{O-Sim}$ consistently exhibits the lowest error compared to the baseline models overall, highlighting its effectiveness in long-term sequence modeling.

Key components in $\texttt{PIC}^2\texttt{O-Sim}$ architecture. Table IV shows the performance of different network settings to predict 80 frames of the light field, where we progressively transform to our proposed $\texttt{PIC}^2\texttt{O-Sim}$. We start with a simple CNN, using the lifting layer from FNO[1] as the encoder. By replacing this with our custom convolutional encoder, we observe a 29% reduction in test error. Next, introducing a dilation factor of 4 and adding a depth-wise convolution to aggregate local information reduces the parameter count by a factor of 10 despite only increasing the error by 2.3%. Further improvements are achieved by partitioning the model into two independent submodels, each with a halved kernel size, resulting in a 17.4% reduction in prediction error. To further refine the model, we propagate an additional hidden state to the second network for error compensation, which enhances fidelity by another 6.1%. Finally, replacing the dilated CNN with our DPAConv layer introduces permittivity awareness, yielding the best performance with just 1M parameters.

TABLE IV: Ablation study on `PIC`^{2}`O-Sim`. Starting from a SimpleCNN with a Lifting field encoder and Conv2d backbone, we progressively add/modify one component. We adopted 72 channels here instead of 36 channels in Table I

	#Params ↓	FPS ↑	Train err ↓	Test err ↓
SimpleCNN (Lifting Enc.+Conv Backbone)	12M	3293	2.52e-2	3.00e-2
+Convolutional Field Encoder	12M	3235	1.74e-2	2.13e-2
+Dilated Conv Backbone	1.1M	7058	1.79e-2	2.18e-2
+Bi-part. w/ Stage-dedicated Models ($\Psi_{\theta_1}, \Psi_{\theta_2}$)	0.9M	5125	1.55e-2	1.80e-2
+Cross-stage Hidden State Propagation	0.9M	5065	1.52e-2	1.69e-2
+Device Enc.+DPAConv Backbone (Final)	1M	4560	1.43e-2	1.59e-2

V. CONCLUSION AND LIMITATION

In this work, we present a physics-inspired causality-aware AI-accelerated FDTD solving framework `PIC`^{2}`O-Sim` for ultra-fast photonic device simulation. We thoroughly analyze and integrate physical constraints into our model design, respecting space-time causality and permittivity-dependent wave propagation principles. Cross-iteration error mitigation techniques have been proposed to compensate for the distribution shift issue during time-bundled autoregressive prediction with balanced scalability, long-term prediction fidelity, and efficiency. Compared to SoTA Fourier-based and convolutional neural operators on three challenging photonic device types, our `PIC`^{2}`O-Sim` outperforms them with 49.1% less prediction error and 23.5 times fewer parameters. Additionally, 133∼310× or 31∼89× speedup has been demonstrated over open-source single-process or 8-process parallel FDTD solvers on average. One potential limitation is that our framework still observes rapidly accumulated errors with large-iteration rollout even though we have various error suppression methods. As a future direction, error suppression during auto-regressive prediction or even a completely new formulation beyond auto-regression will be further investigated to support long timespan optical FDTD simulation.

VI. DATA AVAILABILITY

The data that support the findings of this study are openly available in PIC^{2}O-Sim, reference number[27].

Appendix A: Appendix

1. Optical Simulation Details

For TM^z polarized electric field, the FDTD updating rule is shown as follows:

$$\begin{aligned}
H_x^{q+\frac{1}{2}}\left[m,n+\frac{1}{2}\right] &= H_x^{q-\frac{1}{2}}\left[m,n+\frac{1}{2}\right] - \frac{\Delta_t}{\mu[m,n+\frac{1}{2}]\Delta_y}\left(E_z^q[m,n+1]-E_z^q[m,n]\right)\\
H_y^{q+\frac{1}{2}}\left[m+\frac{1}{2},n\right] &= H_y^{q-\frac{1}{2}}\left[m+\frac{1}{2},n\right] + \frac{\Delta_t}{\mu[m+\frac{1}{2},n]\Delta_x}\left(E_z^q[m+1,n]-E_z^q[m,n]\right)\\
E_z^{q+1}[m,n] &= E_z^q[m,n] + \frac{\Delta_t}{\varepsilon[m,n]\Delta_x}\left\{H_y^{q+\frac{1}{2}}\left[m+\frac{1}{2},n\right]-H_y^{q+\frac{1}{2}}\left[m-\frac{1}{2},n\right]\right\}\\
&\quad - \frac{\Delta_t}{\varepsilon[m,n]\Delta_y}\left\{H_x^{q+\frac{1}{2}}\left[m,n+\frac{1}{2}\right]-H_x^{q+\frac{1}{2}}\left[m,n-\frac{1}{2}\right]\right\},
\end{aligned} \tag{A1}$$

in which the m, n, q represent discrete counterpart of x, y, and t in continuous domain. The $\frac{1}{2}$ shown in the index refers to the points at the middle point of edges in Yee's grid.

2. Motivation for Full-wave Prediction

As depicted in Section. III B 1 and III B 3, `PIC`^{2}`O-Sim` predicts the fields instead of specific quantities. The reasons are three-fold:

- **Enhanced Generalization and Versatility.** Indeed, some prior work has demonstrated neural network prediction of specific device responses (e.g., power splitting ratio for splitters). This is a much simpler task than EM field prediction but highly ad-hoc, not generalizable, and not interpretable. It requires retraining if the target metric or design variable changes. In contrast, we want to explore the capability of AI for scientific PDE-solving tasks for more generic tasks, i.e., learn to solve a family of parametric time-domain Maxwell PDEs. Specifically, our proposed model is designed to predict the entire field, enabling seamless application to a wide range of tasks without the need for retraining. This holistic approach ensures the model's utility across diverse scenarios, making it a more efficient, versatile, and explainable solution in the long run.

- **Critical for Adjoint Inverse Design Applications.** Predicting the entire field is indispensable for adjoint inverse design tasks as the adjoint gradient requires electrical fields of the forward and adjoint simulation. If a model only predicts fields on an output port, the adjoint method cannot be employed for inverse design. To embrace and enable the power of AI-assisted adjoint optimization in the future, we are motivated to push toward full wave prediction capability in the entire domain.

- **Improved Model Explainability.** Comprehensive field prediction enhances the interpretability of the model. Visualizing the electric field's propagation throughout the device allows designers to gain valuable insights into the physical mechanisms underpinning device operation. This understanding is instrumental in refining and optimizing device designs, fostering a more informed and iterative development process.

In summary, the ability to predict the entire field is not only foundational for generalization and task versatility but also crucial for supporting advanced applications such as adjoint inverse design and improving the interpretability of simulation outcomes.

3. Speedup Justification

When evaluating the speed-up, we did not count the time on data generation and model training; instead, the speed-up reported is the comparison between the inference time of `PIC`^{2}`O-Sim` and simulation time of MEEP for the same simulation tasks. To be specific, the task is to simulate/predict 160 fs of fields and then calculate the DFT field of the central frequency out of the simulated/predicted fields.

The trained model is expected to perform inference tasks on many different simulation instances as it is a neural operator that learns to solve a family of parametric Maxwell PDE, not a single simulation instance. Hence, the dataset generation time and model training time can be largely amortized and thus is not included in runtime comparison.

It is a very good suggestion to include the time cost for dataset collection and model training. We used MEEP for FDTD simulation. As a rough estimation, it took 1 hour to generate the MMI dataset, where 100 FDTD simulation instances were performed, and 10 hours to train our model on a single NVIDIA A6000 GPU.

As shown in Fig. 1(a), our model inference is at least 31$\times$ faster than MEEP FDTD simulation. Assume we use our model to predict 3200 simulation instances for small devices, it totally takes around 320-480s for `PIC`^{2}`O-Sim` and 800 min (above 13 hours) for MEEP. The saved time already exceeds the dataset generation and model training cost, which proves that data-driven neural operator training will pay off.

4. Dataset Generation

For MMI, we randomly generate 20 devices to train and 5 devices to test. For MRR, as it is longer than that of MMI and Metaline, we generate 6 devices to train and 5 to evaluate. For Metaline, 32 devices to train and 8 devices to test, based on the variable settings and distributions in Table V. Each device has an individual simulation for each input port. MMIs sweep over 3 ports and generate a total of 75 simulation videos; MRRs only have 1 input port and generate a total of 11 simulation videos (much longer); Metalines sweep over 3 ports and generate a total of 120 simulation videos. The time interval between two frames is 1 fs, i.e., $\Delta_t = 1$ fs.

How to sample video patches as training/validation/test dataset. First, according to different device types, we select different numbers of devices for train and testing, as demonstrated above. During training, for each example, we randomly select one device and one port and slice a video segment. The video segment has a randomly sampled starting frame index of i. $E_{i;i+T_{in}}$ will be the input fields. When sampling the starting frame index of i, considering the imbalanced temporal distribution of the source, which means that the source only exists within the first 560 frames approximately, we attribute more probability to sample the starting frames before 560 frames. To be specific, for MMI and Metaline, starting frames that contain a source have twice the probability of being sampled, and for MRR, since the video is much longer than that of MMI and Metaline, the starting frames have 6 times the probability of being sampled. Sources $J_{T_{in}:T_{in}+T}$ will be extracted from $E_{T_{in}:T_{in}+T}$ at the input port region. $E_{T_{in}:T_{in}+T}$ serves as the target fields. Examples across epochs are totally randomly sampled.

For validation and inference, we uniformly slice the videos with an offset of 16 frames, i.e., $i = 0, 16, 32, \cdots$. We do not resample the starting frames with sources in validation and test. All video slices have bilinear interpolated to have the same spatial resolution ($\Delta_x = \Delta_y = 140nm$). Since our mini-batch size during training and inference is, no padding is added.

TABLE V: MMI, MRR, Metaline device configurations in dataset generation.

Variables	Value/Distribution			Unit
	Tunable MMI 3×3	MRR	Metaline 3×3	
Length	$\mathscr{U}(20, 30)$	-	$\mathscr{U}(8, 10)$	μm
Width	$\mathscr{U}(5.5, 7)$	-	Length	μm
Radius	-	$\mathscr{U}(5, 15)$	-	μm
Port Length	3	1	3	μm
Port Width	$\mathscr{U}(0.8, 1.0)$	$\mathscr{U}(0.5, 0.8)$	$\mathscr{U}(0.8, 1.0)$	μm
Taper Length	2	-	2	μm
Taper Width	Port Width+0.3	-	Port Width+0.3	μm
Ring Bus Width	-	$\mathscr{U}(0.5, 0.8)$	-	μm
Bus Waveguide Gaps	-	$\mathscr{U}(0.1, 0.15)$	-	μm
#Slots	-	-	1.4*Length	μm
Spacing	-	-	Length/3	μm
w_{slot}	-	-	$\mathscr{U}(0.1,1)$	μm
h_{slot}	-	-	$\mathscr{U}(0.2,0.25)$	μm
Border Width	1	1	1	μm
PML Width	2	2	2	μm
Wavelength range	[1.4, 1.65]	[1.4, 1.65]	[1.4, 1.65]	μm
$\varepsilon_{\text{cladding}}, \epsilon_r$	$\{2.07, 12.11\}$	$\{1, 6\}$	$\{2.07, 12.11\}$	-
Video frames	833	[800*Radius/3]	600	fs

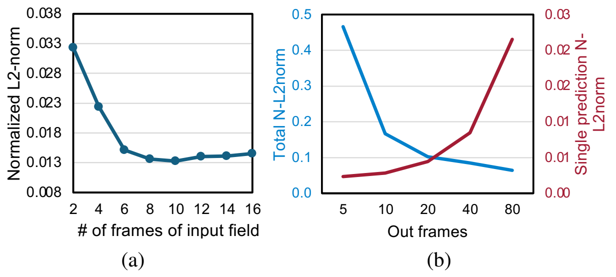


FIG. 8: (a) Input fields with $T_{in} = 10$ gives the lowest prediction error on MMI. (b) The total roll-out error dominates one single prediction.

5. Hyper-Parameter Selection

How to determine kernel size K. Kernel size K is the key design parameter for DPAConv that determines the receptive fields of the model.

To predict the fields after T timesteps, aware of space-time causality, we estimate the furthest distance the wave can propagate in the medium as $R = \sum_0^T \frac{c_0 \times t}{\sqrt{\epsilon_r}}$. Since most light fields are confined in the waveguide region, we use the relative permittivity of the waveguide ϵ_r^{wg} to calculate the theoretical receptive field $R \approx \frac{Tc_0}{\sqrt{\epsilon_r^{wg}}}$. Empirically, we recommend a 30 pixels larger receptive field than the theoretical value $R \approx \frac{Tc_0}{\sqrt{\epsilon_r^{wg}}} + 30$ to obtain the best fidelity as shown in Figure 9 in which we sweep the receptive field for different timesteps to predict. Then, each PAConv is assigned to have a receptive field of $\lceil (R-5)/L \rceil$, where 5 is the RF of the field encoder. With a dilation factor of s, the kernel size of the DPAConv is set to $K = \lceil \frac{R-5}{sL} \rceil$, and the kernel size of the precedent depth-wise convolution is set to $s+1$. The device encoder should extract features of the permittivity map along the optical path toward the center pixel, as illustrated in Figure. 5. Hence, we set the receptive field of the device encoder to K.

Frames of input light field E_{in}. `PIC`^{2}`O-Sim` takes the light

TABLE VI: Compare different dilation factors s on Conv2d layers in a SimpleCNN and MMI dataset in terms of parameter count, runtime, and prediction error (N-L2Norm). K_{DWConv}, K_{Conv} represents kernel size for depthwise convolution and Conv2d. The receptive field roughly remains the same (17∼23). We select s=4 to balance efficiency and fidelity.

K_{DWConv}	K_{Conv}	s	#Params ↓	FPS ↑	Train error ↓	Test error ↓
N/A	17	1	12M	3235	1.74e-2	2.13e-2
3	9	2	3.5M	3415	1.65e-2	2.04e-2
5	5	4	1.1M	7058	1.79e-2	2.18e-2
9	3	8	0.5M	9417	1.91e-2	2.29e-2

Model	PIC2O Sim		
Norm in device encoder	train error	test error	batch size
None	1.42E-02	1.61E-02	1
Batch norm	1.43E-02	1.57E-02	
Layer norm	1.43E-02	1.59E-02	
None	1.47E-02	1.64E-02	2
Batch norm	1.50E-02	1.67E-02	
Layer norm	1.44E-02	1.60E-02	
None	5.15E-02	5.33E-02	16
Batch norm	4.95E-02	5.14E-02	
Layer norm	5.41E-02	5.69E-02	

TABLE VII: The smaller the batch size, the better the performance, there is no obvious difference between different normalization techniques

fields E_{in} from previous timesteps as the initial condition for field prediction. Given the time locality that we analyzed in Section III A, theoretically, two frames ($T_{in} = 2$) should provide sufficient information to obtain the current field distribution to calculate the time derivative $\partial E/\partial t$ using first-order finite difference in the Maxwell equation, which indicates the light propagation direction. Fig. 8(a) investigates the impacts of input frames on the prediction error. We find out that a small number of frames fail to provide enough information for the neural operator to capture the effective initial condition. At least 8-10 frames are required for the model to deliver low prediction errors. Note that more timesteps in the input light fields are harmful since the provided information from the further past is irrelevant and useless due to time locality.

Multi-stage partitioning in time-bundling. Time bundling is preferred to reduce the iteration times during the autoregression. As shown before, there is a trade-off between the speed and memory for different output frames since the output frames are directly related to the kernel size. In addition, another trade-off is between the single prediction accuracy and roll-out prediction accuracy. Fig 8(b) shows the average normalized L2-Norm of the entire 160 frames prediction. Fewer output frames make it an easier task and, hence, a smaller single prediction error. However, the fidelity benefit obtained from a smaller prediction frame count vanished quickly. For the 160-frame prediction task, the 80 output frames model achieves the best rollout error.

Efficiency-fidelity trade-off in convolution dilation factor.

As convolution kernel size increases, especially for dynamic PAConv, the computation and memory cost increase quadratically. We explore the efficiency-fidelity trade-off with different dilation factors s in Table VI. Dilated convolution could speed up the training process dramatically, and combined with a depth-wise local information aggregation convolution, and it also achieves high fidelity. TableVI shows different dilation strategies with the same equivalent receptive field. Considering the trade-off between fidelity, speed, and parameter efficiency, we choose 4 as the dilation rate.

Light field normalization. To increase the generalization and convergence of the model, irrelevant information related to field intensity (light brightness) needs to be normalized. Besides, proper normalization is crucial to minimize error accumulation in autoregressive prediction. A common method used in PDE learning tasks is to normalize the input field to standard normal distribution per channel and de-normalize the model output using the calculated statistics. However, this method does not apply to optical FDTD problems with a Gaussian light source where the resultant light field distribution has *high spatial and temporal sparsity*. We also observe huge intensity discrepancies across space, time, and data samples. We evaluate different normalization methods on a 160-frame roll-out test in Fig. 10. Subtraction of the mean value is not involved in all the normalization methods since the mean of the wave is already zero. To be specific, the first one, 15*std, means that instead of using one standard deviation, we use 15 times of standard deviation to normalize the field since the field are sparse in the devices especially in MRR. Using one standard deviation will make the range to be too large. Second, in regional standard deviation, we first determine the wave propagating region by the average energy and then only calculate the standard deviation within the region for normalization. The avgpool max refers to the method where we first pick the frame among all the input frames that contains the maximum electric field values. And then reduce the spatial dimension by 8 times. Finally pick the maximum value in the reduced frame. The avgpool 99.95% max is almost the same as the avgpool max except that instead of pick the maximum electric field value in the reduced frame, in this method, we pick the 99.5% quantile maximum value to leave some margin for outliers. The method called max is straightforward, just pick the maximum absolute electric field value in the input frames to normalize. The 99.95% max refers to the method that is almost the same with the former one except instead of using the maximum absolute value to normalize, the 99.95% employs the 99.95% quantile to leave some margin for outliers. Statistics based on standard deviation or 99.95% quantile show high roll-out error. Normalizing by the maximum absolute field intensity (**max**) gives the best roll-out fidelity.

Normalization in Device Encoder. In the device encoder, we adopt the LayerNorm as the normalization technique. The rationale of using layernorm is that (1) layer norm is often used

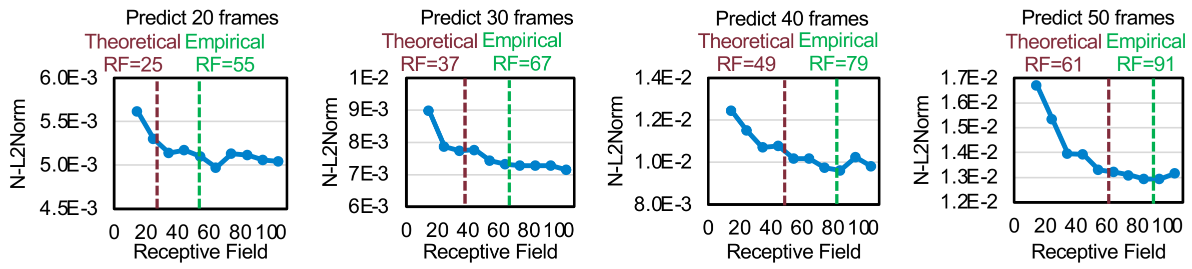


FIG. 9: Prediction error (normalized L2-Norm) with different receptive fields (RF) given various output frames. A 5-layer CNN with various kernel sizes is trained on the MMI dataset. Empirically, using an $\sim 30+$ larger RF than the theoretical one gives the best fidelity and efficiency.

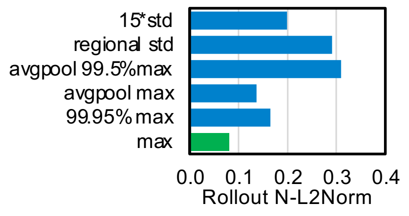


FIG. 10: Using max abs. value to normalize and restore the prediction shows best N-L2norm in one-step roll out on MMI.

when different samples have distinct statistics (e.g., light intensity here), especially in dense prediction tasks, while batchnorm is often used in classification models to capture group statistics). (2) BatchNorm has stability issues with small batch sizes in the literature, as the running statistics (mean/std) are based on a mini-batch. Since we have a relatively small amount of data and the data has relatively large spatial resolution, we use a small mini-batch size for memory efficiency consideration during training and improve generalization due to more stochasticity in small-batch gradients. Hence, we choose to use layer norm by default. To give a more thorough investigation, as shown in Table. VII, we conduct an ablation study on the normalization layer types used in the device encoder, with different normalization layers and batch sizes. Note that the learning rate is proportionally scaled as batch size increases. Compared to larger batch sizes (such as 16), smaller batch sizes (1 or 2) provided better performance. LayerNorm shows the best results, but there is no significant difference between different normalization options. For our model, the encoder is robust for different normalization layer type selections.

Choice of Time Step between Frames. One of the key advantages of `PIC`^{2}`O-Sim` over the FDTD method is that the convergence of FDTD heavily depends on a high-resolution Yee grid and adherence to the Courant-Friedrichs-Lewy (CFL) condition. In contrast, `PIC`^{2}`O-Sim` can directly predict field values on a given mesh without being constrained by the CFL condition, leading to significantly faster computations. Although `PIC`^{2}`O-Sim` is not constrained by the CFL condition, this does not mean that the time step between adjacent frames can be chosen arbitrarily. Beyond the Nyquist sampling limit, there is **a trade-off between efficiency and accuracy**. A smaller time step simplifies the learning task by limiting the wave's propagation distance between frames but reduces efficiency, whereas a larger time step enhances efficiency at the cost of accuracy, as the wave travels a greater distance per step. To investigate this trade-off, we conducted an ablation study on the influence of time step selection. In our experiment settings, the period of light is 5.2 fs, meaning that the maximum allowable time step is 2.6 fs according to the Nyquist sampling criterion. As shown in Table VIII, smaller time steps yield higher accuracy compared to larger ones.

Model	Time step (fs)	# frames	Pred. range (fs)	Train err	Test err
PIC2O-Sim	1	160	160	0.042	0.052
	2	160	320	0.117	0.119

TABLE VIII: Ablation study on time step on benchmark MMI, smaller time step shows better accuracy

6. Training/Inference Settings

We adopt Adam optimizer with an initial learning rate of 2e-3, following a cosine learning rate decay schedule and a minimum learning rate of 1e-5 for all the baselines except for the SineNet for which we use the suggested initial learning rate 2e-4 by its author and ended at 1e-6 All models are trained and evaluated on two servers with 8 NVIDIA A6000 GPUs. The runtime for all neural network models is averaged across 5 runs per photonic device in the test dataset. The runtime for the CPU numerical FDTD solver MEEP is evaluated on a 64-core AMD EPYC 7763 64-Core Processor.

7. Model Architecture Details

We compare our method with SoTA Fourier-domain neural operators and CNN models. We set a full mode for Fourier-domain neural operators to enable them to learn local window operations. For CNN models, we maintain similar layers and the number of parameters for fair comparison. And for Fourier kernel integral operation models, we try to choose as many modes as possible to capture the local wave behavior.

FNO[1]: We construct a 4-layer FNO with Fourier modes of (128, 128), hidden channel of 36. The total parameters are 340M for FNO and MRR. For Metaline, the data are padded to 168, so the full mode is (68, 68), with a hidden channel of 36; the total parameters are 146M.

F-FNO[4]: We construct a 12-layer F-FNO whose modes, for

MMI and MRR that padded to 256×256 is (128, 129) and for metaline which is padded to 168×168, is (84, 85). The number of parameters is 4.5M and 3.3M, respectively.

KNO[26]: We construct a 2-layer KNO whose modes, for MMI and MRR that padded to 256×256 is (128, 129) and for Metaline which is padded to 168×168, is (84, 85). The number of parameters is 171.8M and 74.6M, respectively.

NeurOLight[14]: We construct a 6-layer NeurOLight whose modes, for MMI and MRR that padded to 256×256 is (128, 129) and for Metaline which is padded to 168×168, is (84, 85). The number of parameters is 2.2M and 1.6M, respectively.

SimpleCNN: We construct a 16-layer SimpleCNN. The kernel size is set to be 15 for MMI, and Metaline has the same ε_r, and hence the same required receptive field and the kernel is set to be 21 for MRR due to the high-speed light because of the relatively small ε_r. The number of channels is set to 32 so that the number of parameters is within a reasonable range, which is 3.8M for MMI and Metaline and 7.3M, respectively.

SineNet[19]: We construct a SineNet with 8 waves for MMI and MRR whose was padded to 256×256, the number of downsampling and upsampling blocks in each wave is 4, and the initial hidden channel is 24 so that the number of parameters keeps reasonable. For Metaline, to cooperate with its size, which is 168×168, we changed the number of downsampling and upsampling blocks to 3 in each block, and to compensate, we set the initial channels to 42, and the number of parameters is 30M.

`PIC`^{2}`O-Sim`: For the device encoder, we use a single convolution layer by a depth-wise convolution layer followed by layer normalization in ConvNeXt[28] style and GELU, a skip connected is added connecting from input of the depth-wise convolution to the end of GELU. Then, the above structure is copied once and cascaded together to form our device encoder. The output channel for the four convolutional layers is $1 \rightarrow 72 \rightarrow 72 \rightarrow 48 \rightarrow 48$. For MMI and Metaline, the kernel size is [3 3 5 5], and for MMR, the kernel size is [5 5 5 5]. The device encoder is padded with replicate mode. For the field encoder, we use almost the same configuration as the device encoder except for the number of channels; the channels are now becoming # of input fields + # of sources $\rightarrow 72 \rightarrow 72 \rightarrow 72 \rightarrow 72$. The kernel size becomes [1 3 1 3], and the fields are padded using zero padding. For the hidden state adaptor, we use a single point-wise convolution, with input channels of 72+72=144 and output channels of 72. For the backbone, we use 8 layers. For MMI and Metaline, each layer has a local aggregation depth-wise convolution layer whose channel number is 72 and kernel size equals 5, a DPAConv layer with kernel size 5 and dilation 4 to provide enough receptive field followed by layer normalization and GELU, all these modules are included within skip connection. For MRR, the basic structure remains the same except for the kernel size for DPAConv, which becomes 7 to provide a wider receptive field. For the decoder, we use two point-wise convolutional layers in which the first one lifts the D-dimension feature to a 512-dimension vector and the second one projects it back to the required output frames, in our case, 80. The total parameters are 4.4M for MRR and 2.4M for MMI and Metaline.

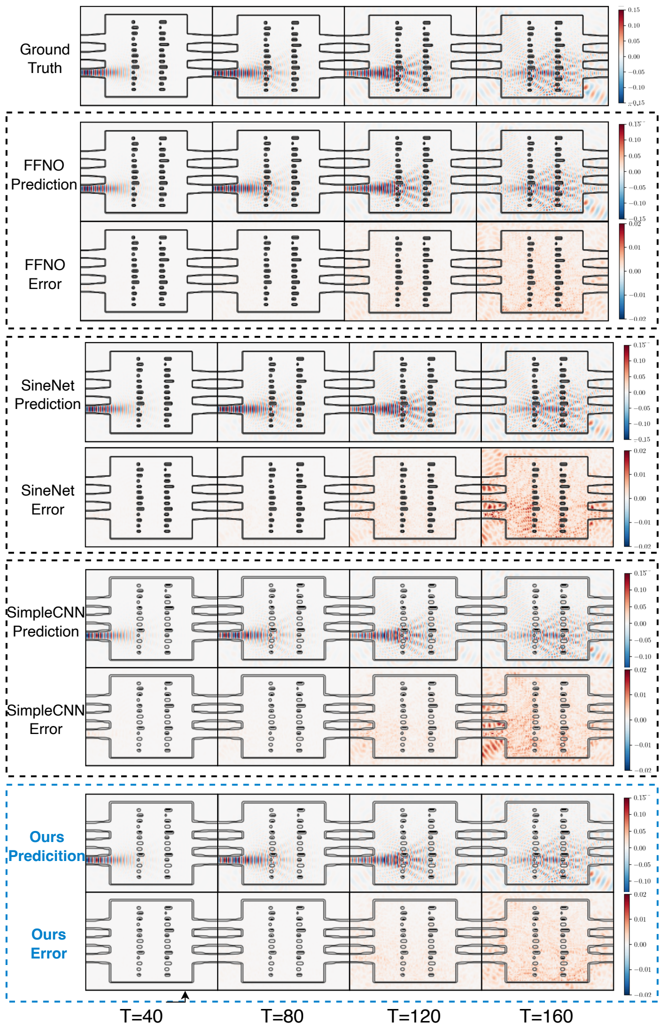


FIG. 11: `PIC`^{2}`O-Sim` Visualization on Metaline, sampled every 40 frames

8. Prediction Result Visualization

In this section, we select the best-performed baselines and our `PIC`^{2}`O-Sim` to show their performance in 160 frames prediction on different devices. We sampled the 160-frames whole video every 40 frames, and to make the error more obvious, and the error is plotted with a smaller scale from -0.02 to +0.02

Figure 12 shows the visualization of the 160 frames prediction of the selected baselines and `PIC`^{2}`O-Sim` on MMI. Since the relatively simple structure, the error among all the baselines is small. However, there is still an obvious performance gap between `PIC`^{2}`O-Sim` and other well-performed baselines.

Figure 13 shows the visualization of the 160 frames prediction of the selected baselines and `PIC`^{2}`O-Sim` on MRR in which a pulse of light is propagating through the slim ring waveguide and coupled to the straight waveguide. More complicated devices raise more challenging tasks for these surrogates. FFNO suffers from the Fourier integral operation and shows huge errors. SineNet also performed badly due to the irrelevant features. CNN-based operators obtained better fi-

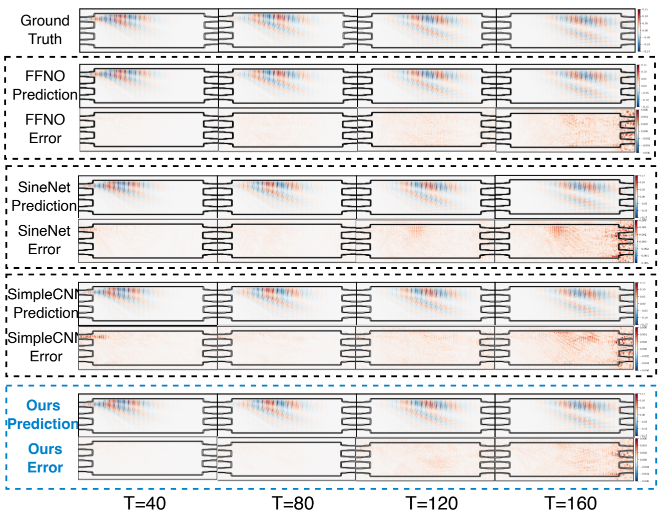


FIG. 12: PIC^{2}O-Sim Visualization on MMI, sampled every 40 frames

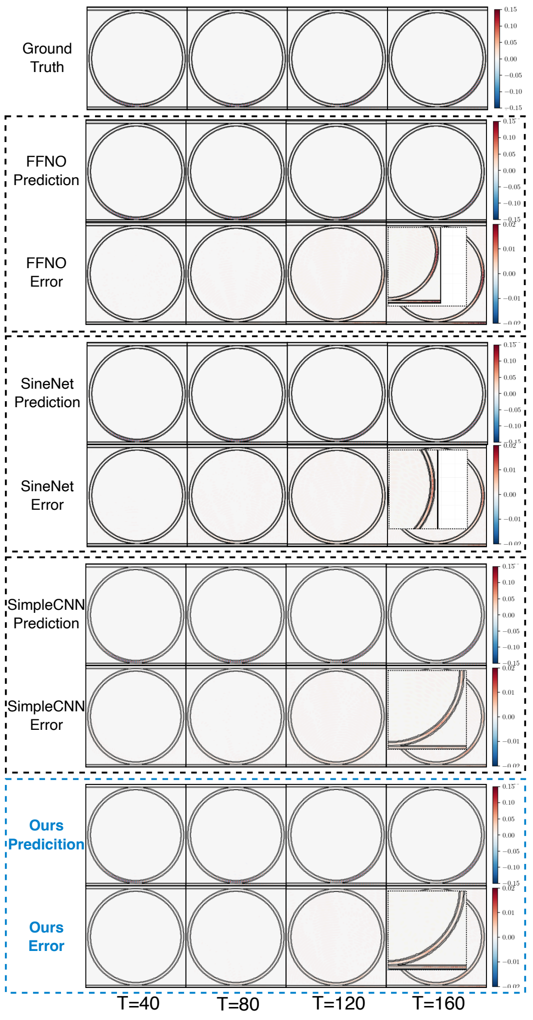


FIG. 13: PIC^{2}O-Sim Visualization on MRR with zoom-in views, sampled every 40 frames.

delity, and the PIC^{2}O-Sim, due to its physics causal dynamic kernel, has a slightly better performance than SimpleCNN.

Figure 11 shows the visualization of the 160 frames prediction of the selected baselines and PIC^{2}O-Sim on Metaline. The Metaline has the most complicated structure, which causes the larger performance gap between PIC^{2}O-Sim and SimpleCNN as expected.

9. Potential Applications

PIC^{2}O-Sim is particularly well-suited for scenarios where long simulation times are required, but waiting for the full decay of all signals is impractical. One representative example is the FDTD simulation of high-Q resonators, where long runtimes are necessary for the field to decay to a sufficiently small value, making the computation expensive. However, in high-Q resonators, the field typically follows an exponential decay, given by $E(t) \approx E_0 e^{-\alpha t}$. This suggests that it is possible to simulate only the early stages of field evolution and accurately extrapolate the full decay behavior. PIC^{2}O-Sim can be especially beneficial in such cases, significantly reducing computational costs while maintaining accuracy.

10. Limitations and Future Directions

Although PIC^{2}O-Sim demonstrates the best prediction fidelity among the strong ML-based PDE surrogate solver baselines, we still observe error accumulation during auto-regressive predictions that currently cannot support end-to-end FDTD simulation over arbitrarily long time frames. The source of the accumulated error in roll-out long-range prediction can be categorized into neural network function approximation error, model generalization error, and optimization-induced error. Future directions to push toward AI-enabled optical simulation include better model architecture (ansatz) with strong inductive bias for better PDE representation, new formulation besides simple autoregression to suppress temporal error accumulation, and better optimization methods to ensure robust model learning to distribution shifts during prediction.

REFERENCES

[1]Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," (2021), arXiv:2010.08895 [cs.LG].

[2]H. Su, V. Jampani, D. Sun, O. Gallo, E. Learned-Miller, and J. Kautz, "Pixel-adaptive convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

[3]M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," Journal of Computational Physics **378**, 686–707 (2019).

[4]A. Tran, A. Mathews, L. Xie, and C. S. Ong, "Factorized fourier neural operators," in *The Eleventh International Conference on Learning Representations* (2023).

[5]N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural operator: Learning maps between function

spaces with applications to PDEs," Journal of Machine Learning Research **24**, 1–97 (2023).

[6]M. Ashiqur Rahman, Z. E. Ross, and K. Azizzadenesheli, "U-no: U-shaped neural operators," arXiv e-prints , arXiv–2204 (2022).

[7]O. Ovadia, V. Oommen, A. Kahana, A. Peyvan, E. Turkel, and G. E. Karniadakis, "Real-time inference and extrapolation via a diffusion-inspired temporal transformer operator (ditto)," (2023), arXiv:2307.09072 [cs.LG].

[8]M. L. Taccari, O. Ovadia, H. Wang, A. Kahana, X. Chen, and P. K. Jimack, "Understanding the efficacy of u-net & vision transformer for groundwater numerical modelling," (2023), arXiv:2307.04010 [physics.flu-dyn].

[9]G. Yang and S. Sommer, "A denoising diffusion model for fluid field prediction," (2023), arXiv:2301.11661 [cs.LG].

[10]T. Kurth, S. Subramanian, P. Harrington, J. Pathak, M. Mardani, D. Hall, A. Miele, K. Kashinath, and A. Anandkumar, "Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators," in *Proceedings of the Platform for Advanced Scientific Computing Conference* (2023).

[11]M. Chen, R. Lupoiu, C. Mao, D.-H. Huang, J. Jiang, P. Lalanne, and J. Fan, "Physics-augmented deep learning for high-speed electromagnetic simulation and optimization," Nature (2021).

[12]J. Lim and D. Psaltis, "Maxwellnet: Physics-driven deep neural network training based on maxwell's equations," Appl. Phys. Lett. (2022).

[13]Y. Tang, J. Fan, X. Li, J. Ma, M. Qi, C. Yu, and W. Gao, "Physics-guided and physics-explainable recurrent neural network for time dynamics in optical resonances," Nat. Compu. Sci. (2022).

[14]J. Gu, Z. Gao, C. Feng, H. Zhu, R. T. Chen, D. S. Boning, and D. Z. Pan, "Neurolight: A physics-agnostic neural operator enabling parametric photonic device simulation," in *Proc. NeurIPS* (2022).

[15]Y. Augenstein, T. Repän, and C. Rockstuhl, "Neural operator-based surrogate solver for free-form electromagnetic inverse design," ACS Photonics (2023).

[16]O. Noakoasteen, S. Wang, Z. Peng, and C. Christodoulou, "Physics-informed deep neural networks for transient electromagnetic analysis," IEEE Open Journal of Antennas and Propagation **1**, 404–412 (2020).

[17]P. Zhang, Y. Hu, Y. Jin, S. Deng, X. Wu, and J. Chen, "A maxwell's equations based deep learning method for time domain electromagnetic simulations," in *2020 IEEE Texas Symposium on Wireless and Microwave Circuits and Systems (WMCS)* (2020) pp. 1–4.

[18]T. W. Hughes, I. A. D. Williamson, M. Minkov, and S. Fan, "Wave physics as an analog recurrent neural network," Science Advances (2019).

[19]X. Zhang, J. Helwig, Y. Lin, Y. Xie, C. Fu, S. Wojtowytsch, and S. Ji, "Sinenet: Learning temporal dynamics in time-dependent partial differential equations," in *Proc. ICLR* (2024).

[20]O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, edited by N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi (Springer International Publishing, Cham, 2015) pp. 234–241.

[21]M. Xu, R. Ding, H. Zhao, and X. Qi, "Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *CVPR* (2021).

[22]H. Hachiya, K. Nagayoshi, A. Iwaki, T. Maeda, N. Ueda, and H. Fujiwara, "Position-dependent partial convolutions for supervised spatial interpolation," in *Proceedings of The 14th Asian Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 189, edited by E. Khan and M. Gonen (PMLR, 2023) pp. 420–435.

[23]T. W. Hughes, I. A. D. Williamson, M. Minkov, and S. Fan, "Wave physics as an analog recurrent neural network," Science Advances **5**, eaay6946 (2019).

[24]J. Brandstetter, D. Worrall, and M. Welling, "Message Passing Neural PDE Solvers," in *Proc. ICLR* (2023).

[25]A. Oskooi, D. Roundy, M. Ibanescu, J. J. P. Bermel, and S. Johnson, "Meep: A flexible free-software package for electromagnetic simulations by the fdtd method," Computer Physics Communications (2010).

[26]W. Xiong, X. Huang, Z. Zhang, R. Deng, P. Sun, and Y. Tian, "Koopman neural operator as a mesh-free solver of non-linear partial differential equations," (2023), arXiv:2301.10022 [cs.LG].

[27]P. Ma, H. Yang, Z. Gao, D. S. Boning, and J. Gu, "Pi2co-sim open-source codes and data," (2024), https://github.com/ScopeX-ASU/PIC2O-Sim.

[28]Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," (2022), arXiv:2201.03545 [cs.CV].