Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

From Physics to Attention: Building Vision Transformers from Oscillatory Neural Network Hardware

Tamas Endrei
Pázmány Péter Catholic University
Andras Horvath
Pázmány Péter Catholic University
Gyorgy Csaba
csaba.gyorgy@itk.ppke.hu
Pázmány Péter Catholic University

Article

Keywords: Oscillatory neural networks, Transformers, Energy-based models, Hopfield networks, Neuromorphic hardware, Analog circuits

Posted Date: September 1st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10748299/v1

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

Additional Declarations: No competing interests reported.

From Physics to Attention: Building Vision Transformers from Oscillatory Neural Network Hardware

Tamás Endrei[1], András Horváth[1], György Csaba[1,*]

[1]Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, Práter u. 50/A, 1083 Budapest, Hungary

E-mail: endrei.tamas.roland@gmail.com; horvath.andras@itk.ppke.hu; csaba.gyorgy@itk.ppke.hu

*Corresponding author: György Csaba, csaba.gyorgy@itk.ppke.hu, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, Práter u. 50/A, 1083 Budapest, Hungary

July 2026

Abstract

Oscillatory Neural Networks (ONNs) promise energy-efficient neuromorphic computing, yet practical demonstrations remain confined to shallow fully-connected layers on simple tasks, with no clear pathway for scaling them to the level of modern, deep AI pipelines. We present a computational design of HoKuVit, a Convolutional Vision Transformer composed of ONNs that scales ONNs far beyond their current state of the art. HoKuVit uses Kuramoto-oscillator-based convolutional layers that implement spatial filtering through oscillator synchronization, and Hopfield-network-based linear layers that implement energy-minimizing associative memory. Both primitives are trained end-to-end via backpropagation. By interconnecting Hopfield and Kuramoto-based primitives within a hierarchical Convolutional Vision Transformer backbone, we scale HoKuVit to 0.866 million effective parameters, of which 88.04% are implemented as ONN components. The network achieves 79.78% accuracy on CIFAR-10, which is respectable for a network of this size and the strongest result reported for an architecture built almost entirely on oscillator dynamics.

Keywords: Oscillatory neural networks; Transformers; Energy-based models; Hopfield networks; Neuromorphic hardware; Analog circuits.

1 Introduction

Unconventional hardware promises to run AI workloads faster, and at lower energy, than digital logic can. Modern neural networks, however, have outgrown anything neuromorphic hardware has demonstrated. Contemporary generative AI systems [1, 2] handle language, vision and reasoning well enough to emulate aspects of human behavior, whereas most neuromorphic hardware research remains largely confined to toy problems [3, 4, 5, 6], such as the classification of simple handwritten digits or low-resolution patterns. Oscillator networks - a prominent example of unconventional, dynamic computing - solve specific problems at very low energy cost [7, 8], or act as preprocessors (first layers) to complex pipelines [9, 10], yet, to our knowledge, nobody has scaled these substrates to architectures that handle real-world vision [3, 11, 12]. Here we show that ONN circuit blocks can serve as computational primitives from which modern AI architectures can be assembled. Specifically, Hopfield linear layers and Kuramoto convolutional layers are sufficient to carry 88% of the parameters of a vision transformer that operates on real-world images, with only the attention projections and classification head remaining digital. Ideally, this could give a factor of 4 power saving compared to a digital implementation, if the ONN blocks operate at negligibly small power level compared to the remaining digital circuitry.

Emerging hardware substrates differ in how readily they slot into existing AI architectures. Memristive crossbars accelerate the very operation conventional deep learning is already built on – the weighted sum – so they can serve as drop-in replacements for the linear layers of an otherwise unchanged architecture [13, 14, 15]. Coupled oscillator networks, the substrate we study here, offer no such correspondence: they do not evaluate a feedforward function but are dynamical systems forming a collective steady state. There is therefore no layer for them to replace, and exploiting them requires co-designing the AI architecture around their dynamics rather than substituting them into an existing AI pipeline as drop-in replacements.

ONNs encode information in the phase of coupled oscillators [16]. Phase-locking dynamics support associative memory and pattern recognition directly [11], and maps directly onto physical devices such as spin-torque oscillators [7, 17], parametric oscillators [18], ring oscillators [19] and relaxation oscillators [20], all of them compact and low-power at the nanoscale. Despite their theoretical capacity to address even NP-hard combinatorial problems [11, 12], current ONN demonstrations remain confined to rather specific tasks only. While multilayer ONNs have been designed by machine learning algorithms [9], the core operational principles of these networks still rely on traditional Hopfield-like associative dynamics, which scale poorly and lack the representational capacity of modern deep learning architectures.

Some of the computational principles that originally motivated neuromorphic hardware have re-emerged in mainstream machine learning [21]. Modern continuous Hopfield networks reinterpret associative memory retrieval as an attention operation, establishing a formal correspondence between energy-

minimizing Hopfield dynamics and the transformer attention mechanism [22, 23, 24] and achieving exponential storage capacity through a softmax-based update rule. This correspondence motivates the present work. Transformer-class architectures are, at their core, related to the attractor dynamics that oscillator hardware implements natively. The modern Hopfield formulation itself, however, is not directly related to a physical system and to the minimization of a physical energy. Its capacity gains rest on the softmax nonlinearity and on continuous, high-precision state updates, operations that do not map to the phase-locking dynamics of coupled oscillators. Real circuits implement the classical Hopfield model with symmetric resistive couplings [11, 25, 26]. We therefore adopt the classical formulation. This keeps physical realizability at the cost of per-layer capacity.

In this work, we present HoKuVit (Figure 1), a Convolutional Vision Transformer that scales oscillatory neural networks to a modern deep learning architecture by interconnecting Kuramoto-based convolutional layers and Hopfield-based linear layers as its core computational primitives. Building on the idea of assembling larger networks from smaller ONN components [9], we show that traditional Hopfield networks and oscillator-based convolvers, primitives that on their own can solve only elementary tasks, can be interconnected to form modern transformer-like architectures. Kuramoto convolvers naturally produce the phase-encoded representations that Hopfield blocks later process through their energy-minimization dynamics. By stacking the Hopfield and Kuramoto convolver primitives within a hierarchical CvT-style [29] backbone, HoKuVit scales to 0.866 M effective parameters.

2 Results

2.1 Architecture overview

HoKuVit follows a hierarchical, three-stage Convolutional Vision Transformer (CvT) architecture [29], in which each stage progressively reduces spatial resolution while increasing feature dimensionality (Fig. 1). The physical building blocks are Kuramoto convolutional layers and Hopfield linear layers. The digital parameters are found inside the Attention projection linear layers, the layernorm parameters, the LayerScale parameters and the MLP Head of the model. The convolutional layers appear in the convolutional token embeddings and in the convolutional transformer blocks, where they are split into depthwise and pointwise convolutions; the Hopfield linear layers replace the MLP layers of the transformer blocks. Together these account for 88.04% of the network's parameters, with the remaining 11.96% consisting of standard attention projections and the classification head. More details of both primitives are given in the Methods.

Both the Hopfield layers and the Kuramoto layers are implementable as ONNs. Weakly coupled (Kuramoto) oscillator dynamics is known to implement Hopfield / Ising networks, so a Hopfield network is the simplest model of an

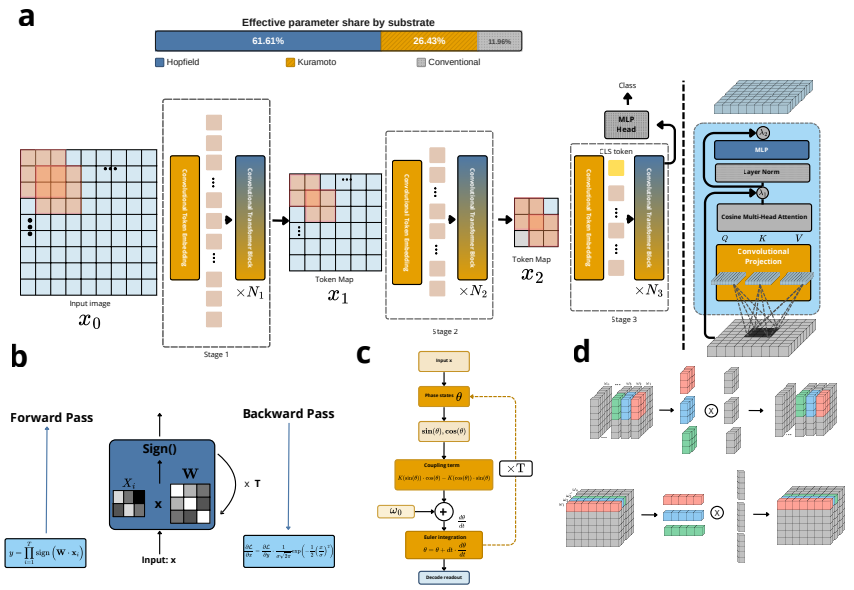


Figure 1: Overview of the proposed HoKuVit architecture. Panel **a** displays the overall architecture of the network, showcasing Hopfield layer parts in blue, Kuramoto layer parts in yellow and conventional neural network parts in gray, together with the effective parameter share of each substrate. In the transformer block we utilize Cosine Multi-head attention [27] and LayerScale [28]. Panel **b** highlights the traditional Hopfield layer and details the use of a different backward function to approximate the forward function, allowing for the training of deeper architectures. Subfigure **c** showcases the flowchart of the Kuramoto convolutional layer and how the ODE integration evolves. Panel **d** shows the differences in pointwise and depthwise Kuramoto convolutions. Depthwise convolutions use a single filter for each channel, while each channel learns a different natural frequency. Pointwise convolutions learn a single filter that spans across all channels for a given spatial position. Natural frequencies are learned per spatial position in the pointwise setting.

ONN [11, 16]. The Kuramoto convolver layers are explicitly coupled Kuramoto oscillators in the spirit of [30]. Every parameter of the Hopfield and Kuramoto networks therefore maps onto an oscillatory building block. We envision that the oscillatory building blocks of HoKuVit will eventually be realizable by hardware elements (such as ring oscillators) even if the learning model presented here uses simple oscillator models (Kuramoto model) and in the case of the Hopfield blocks, discretizes the dynamics, which, as we will show, does not change the converged states.

The learnable parameters of the ONN components correspond to circuit parameters: the Hopfield layers learn coupling weights, which eventually cor-

respond to resistive couplings in an ONN. In the Kuramoto layers, oscillator frequencies and couplings are the learned hardware parameters.

The Kuramoto layers are shown schematically in Fig. 1c. Oscillators with different (learnable) ω_i frequencies are interconnected with a trainable K_{ij} weight matrix. Owing to the different ω_i frequencies, this network does not typically converge to a stationary attractor state. In the computational and training model the network is modeled by numerically integrating its ODEs in order to capture the transient dynamics (see Methods).

The motivation for utilizing a continuous Kuramoto model stems from the literature on binary neural networks [31, 32]. When the idea of binary networks is expanded into Vision Transformers [33] due to the long-tailed distribution of softmax attention values [34] the binarization error limits the classification accuracy of the network.

In contrast to the majority of ONN literature, which relies on Hebbian or correlation-based learning rules [11], our Hopfield and Kuramoto layers are optimized via backpropagation. Both layer types therefore train jointly with the rest of the network, and the coupling parameters come from the data instead of being fixed in advance. For the Hopfield layers, this is achieved by decoupling the forward pass, which retains the classical sign-based update rule, from the backward pass, which substitutes a smooth Gaussian surrogate gradient in place of the non-differentiable sign function (Fig. 1b). For the Kuramoto layers, the oscillator ODE is numerically integrated over a fixed number of Euler steps, unrolling the dynamics into a computational graph through which gradients can be backpropagated directly, in the same manner as backpropagation through time in a recurrent network (Fig. 1c). The classification objective therefore shapes the ONN primitives directly. The network finds attractor structures and synchronization regimes that a handcrafted or Hebbian-trained system cannot reach.

2.2 Classification performance

We evaluate HoKuVit on CIFAR-10 [35], a much harder benchmark than the MNIST [36] pattern recognition tasks commonly used to assess oscillatory neural networks. CIFAR-10 consists of 60k natural, colored images of 10 classes made up of animals and vehicles, of which 50k images (5k training samples per class) form the training set and 10k images are held out for evaluation.

HoKuVit reaches 79.78% top-1 accuracy on CIFAR-10 (Table 1), the highest reported for an oscillatory neural network and, to our knowledge, the first time an ONN-based architecture has been evaluated on a vision task beyond MNIST-level pattern recognition. At 0.866 M parameters it is comparable in size to the MobileViT-XXS and CvT baselines. The accuracy loss relative to the same-configuration digital CvT (92.32%) and the spiking SpikingResFormer (94.25%) is the cost of constraining almost the entire network to physically-realizable oscillator dynamics.

Model	Acc. (%)	Params (M)	Digital GFLOPs	Neuro. GFLOPs
Conventional digital models				
MobileNetV3-Small [37]	84.80	1.528	0.0058	—
MobileViT-XXS [38]	84.90	0.954	0.0060	—
CvT (same config)	92.32	0.882	0.0646	—
Spiking neural networks				
SpikingResFormer [39]	94.25	10.79	1.2264	0.3766
Oscillatory neural networks (mixed hardware)				
HoKuVit (ours)	**79.78**	**0.866**	**0.0162**	**0.6091**

Table 1: Comparison of HoKuVit with conventional digital and neuromorphic baselines on CIFAR-10 in terms of accuracy, parameter count, and computational cost. Digital GFLOPs counts the operations executed on conventional digital hardware; Neuro. GFLOPs counts the digital-equivalent operations that would be carried out natively by a neuromorphic substrate in a physical realization – Hopfield and Kuramoto dynamics for HoKuVit, and synaptic operations (SOPs) for the spiking SpikingResFormer (see the energy discussion in Section 3.1).

2.3 Hopfield layers operate as learned associative memory

The phase dynamics of weakly coupled oscillators is equivalent to the dynamics of continuous Hopfield networks [16]. Simulation and machine learning based on a dynamics given by ordinary differential equations (ODEs) would be prohibitively costly and our computational model is run on a discrete model. We find that this discrete model implements attractor states identical to those of a continuous Hopfield network.

The Hopfield linear layers in HoKuVit behave as associative memories. Three measurements support this. First, we track the Lyapunov energy of each Hopfield layer throughout its iterative update process (Fig. 2a). Across all three stages and both projection types of the Hopfield MLP, the energy decreases monotonically and stabilizes within three to five update steps. The learned dynamics thus reach well-defined attractors during inference. We use six update steps, which balances accuracy against physical realizability and computational cost.

Second, we examine the converged Hopfield states for inputs drawn from different CIFAR-10 classes (Fig. 2b), taken from the last Hopfield layer's down projection. The final binary patterns are visibly class-discriminative: each class maps to a distinct attractor signature, with consistent activation structure along the sequence-position axis. The Hopfield layers therefore learn class-relevant representations instead of collapsing onto one fixed point shared by every input.

Third, we examine the converged states for multiple images drawn from a single class (Fig. 2c). The final attractors for five distinct *dog* images are visually similar to one another and visibly distinct from the attractors of other classes shown in panel (b). Similar inputs map to nearby attractors. This is what

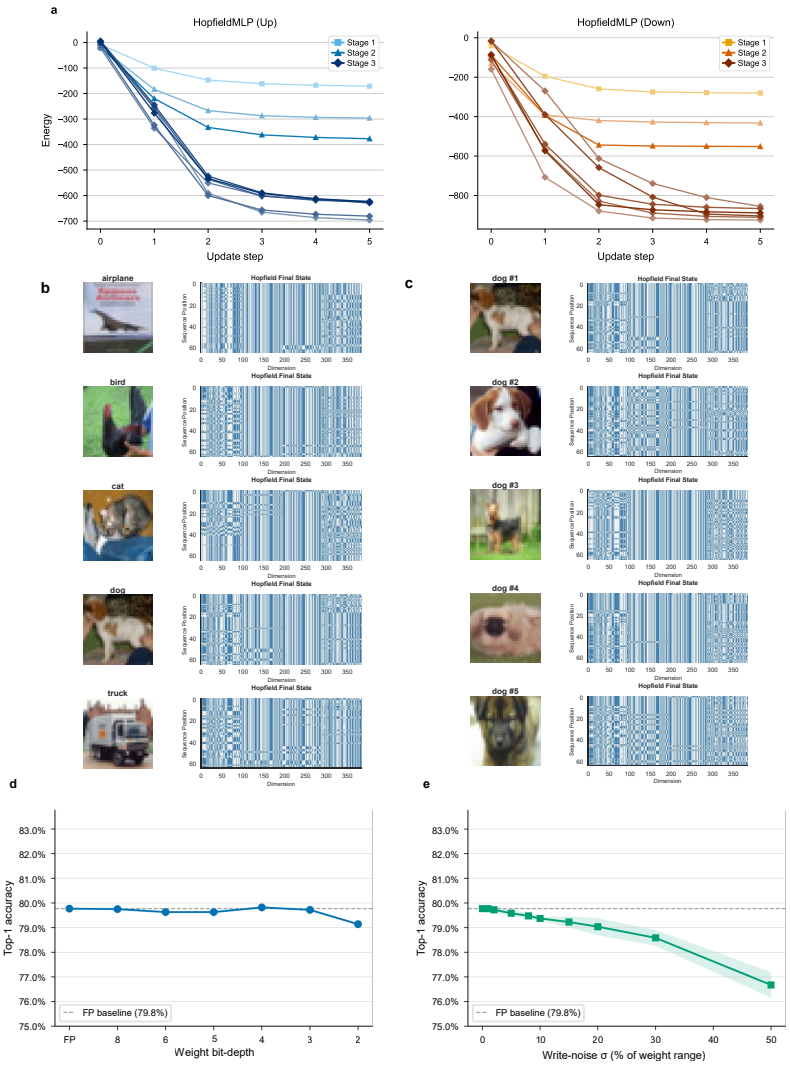


Figure 2: **Hopfield layer dynamics in HoKuVit.** **(a)** Energy trajectories of the up and down projections of a Hopfield layer across the three stages. **(b)** Hopfield final states for five CIFAR-10 classes reveal class-discriminative attractor patterns. **(c)** Final states for five different *dog* images converge to visually similar, but still different attractors, indicating within-class consistency. **(d)** Performance degradation with various levels of weight quantization. **e** Top-1 accuracy with corrupted weights.

an associative memory does: the network projects the high-dimensional input manifold onto a small set of class-organized fixed points.

We also verified for several weight matrices that solving the ODEs of a continuous Hopfield model gives the exact same converged state as the discretized model used for training.

Physical interconnects have limited precision. Figure 2d showcases the top-1 accuracy of the model, where the Hopfield weight matrices are post-training quantized from full floating-point precision down to a 2-bit integer representation. The model's accuracy degrades by less than 0.63 percentage points from the full-precision 32-bit representation down to 2 bits. The Hopfield layers of HoKuVit should therefore map onto existing crossbar hardware without retraining or quantization-aware fine-tuning.

Physical couplings may also introduce stochastic noise during the programming phase. Fig. 2e characterizes HoKuVit's robustness to this noise source by injecting zero-mean Gaussian noise $\mathcal{N}(0, \sigma^2)$ into the Hopfield weight matrices at inference time, where σ is expressed as a percentage of the full weight range. Accuracy degrades by 0.4 percentage points for white noise up to $\sigma = 10\%$, and by approximately 0.73 percentage points at $\sigma = 20\%$. Above $\sigma = 30\%$ the confidence interval widens sharply, as inference becomes stochastic under severe device variability. Further proof of physical realizability is presented via the learned Hopfield weight matrices in the Supplementary material.

2.4 Kuramoto layers implement three distinct synchronization regimes

The motivation for using oscillators for convolutional filtering comes from earlier work [40, 30, 41] - in fact, oscillator-based convolution was one of the first demonstrated applications of ONNs.

The principle that the degree of synchronization of a coupled oscillator array encodes the degree of match between an input fragment and a convolutional kernel has been demonstrated experimentally in CMOS by Nikonov et al. [40, 30], where inputs were encoded as oscillator frequency shifts and a single degree-of-match was read out per relaxation. Here we generalize this scheme: the input enters as initial phases, the kernel enters as learnable coupling weights, and the readout is spatially resolved, so that one relaxation of a single two-dimensional array produces the full feature map. The coupled oscillators never reach a stable phase configuration. Their transient dynamics carry the result of the computation.

We analyze the Kuramoto convolutional layers to verify that they operate within a dynamically expressive regime, learn class-discriminative phase representations, and tolerate the phase imprecision expected from physical oscillator hardware.

The Kuramoto model's behavior can be explained by comparing the natural-frequency spread ω_0 to the maximum coupling strength $K_{\max}$. When this ratio is less than one, the coupling entirely overwhelms the natural oscillations and

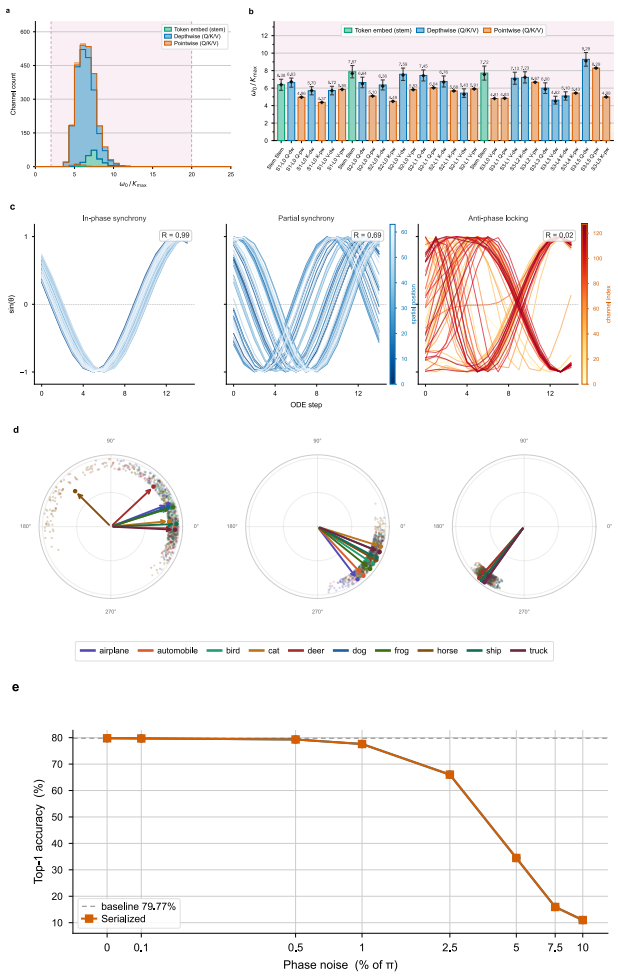


Figure 3: Evaluation of the Convolutional Kuramoto Layers. **a** Showcases the ω/K ranges of the token embedding, depthwise and pointwise convolutional layers. **b** Displays this for selected number of standalone layers. **c** Shows the synchronization types found inside the layers. **d** Displays the final angles of different layers for each class. **e** Shows the noise resilience of the network, where noise is added to the outputs of the kuramoto layer angles.

the oscillators trivially collapse into a completely phase-locked, uniform synchronized state regardless of the input [42, 43]. Conversely, when this ratio is

much greater ($\omega_0/K \to \infty$), the intrinsic frequencies dominate, rendering the coupling forces too weak to entrain the oscillators and collapsing the system into a state of total incoherence [44].

Fig. 3a shows the distribution of $\omega_0/K_{\max}$ across all channels of all Kuramoto layers, separated by layer type. The distribution peaks between 5 and 10, with no channels falling outside the expressive interval $1 < \omega_0/K_{\max} < 20$, the empirical range over which we observe non-trivial synchronization dynamics in the trained network. This regime targets the boundary between incoherence and trivial fixed points [45]. Layer-wise averages (Fig. 3b) confirm that this regime holds across all three stages. Training has converged to dynamics that exploit oscillator coupling instead of degenerating into trivial fixed points.

To characterize the dynamical behavior of individual channels, we inspect representative oscillator-phase trajectories during the 15-step ODE integration (Fig. 3c). We observe distinct synchronization regimes: full in-phase synchrony, where all spatial positions lock to the same phase ($R = 0.99$); partial synchrony, where subgroups of positions form coherent clusters ($R = 0.69$); and anti-phase locking, where opposing positions oscillate in opposition ($R = 0.02$). Because these regimes coexist across channels, the Kuramoto layers evidently learn their own dynamics rather than settling on one type. The regimes emerge from training; they are not handcrafted.

To investigate the internal representations after a Kuramoto layer we showcase depthwise Kuramoto channels shown in Fig. 3d, where we plot the final oscillator phases and their averages across 100 images of each CIFAR-10 class. Each class forms a tight phase cluster at a distinct angular position, and these clusters separate with varying degrees across layers. We can observe that in best cases these clusters separate cleaner, with usually one class being in a different quadrant, in the median cases (middle) the clusters are distinguishable yet still reside in the same quadrant, and in the worst cases the classes collapse to the same point. The lottery ticket hypothesis [46] offers one reading of this spread: an overparameterized network contains sparse subnetworks that carry most of the learned function. Here, a subset of Kuramoto channels develops class-discriminative phase structure while the rest stay dormant.

Finally, because any physical realization on real oscillators will exhibit phase imprecision, we test HoKuVit's robustness to phase noise injected at the output of each Kuramoto layer (Fig. 3e). Accuracy remains within 2 percentage points of the noise-free baseline (79.77%) for phase noise up to $0.01 \times \pi$, and falls off slowly beyond that. The discrepancy between this baseline and the headline accuracy is due to averaging across multiple different seeds.

2.5 Physical feasibility and interpretability

HoKuVit is physically realizable, and this is one of its main contributions. We investigate the Kuramoto and Hopfield dynamics in their continuous formulations and evaluate how closely our discrete implementations match them, both in their energy trajectories and in the similarity of their final feature maps.

The Kuramoto convolutional layers integrate oscillator phases according to a

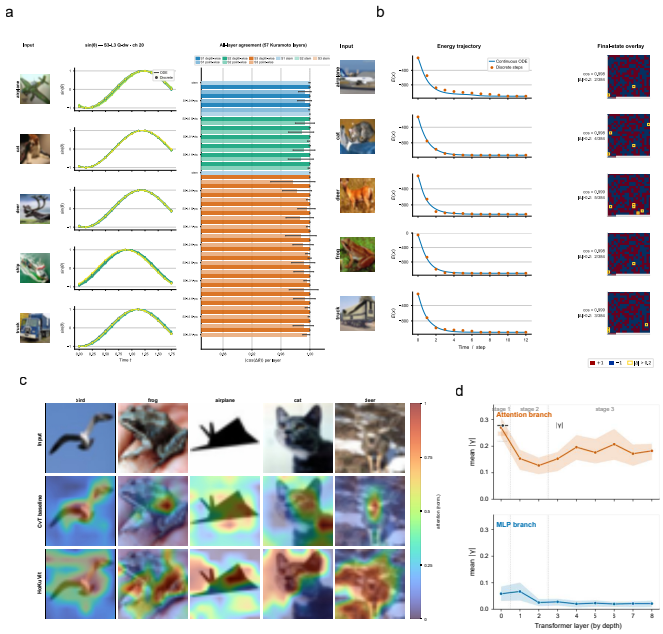


Figure 4: **Interpretability of HoKuVit's oscillatory and Hopfield components. a** Analysis of the Kuramoto convolutional layer ODE. The comparison of a continuous and discrete ODE integration and the cosine similarity of the final phase values. **b** Discrete vs continuous Hopfield attractor dynamics behavior. We compare the energy trajectories of a continuous and discrete Hopfield model and evaluate the difference in the final states of the model. **c** Class token Attention comparison for a traditional CvT and HoKuVit. **d** Layerscale contribution plot. Showcasing the MLP and Attention layers' contributions at each stage.

continuous ODE model with a fixed step size Δt during the forward pass. Fig. 4a shows that our discrete Kuramoto integration closely resembles the continuous ODE trajectories calculated via RK4 integration. Across all 57 Kuramoto layers the cosine similarity stays above 0.96. The 15-step Euler integration with step size $\Delta t = 0.125$ therefore introduces negligible approximation error, and the learned coupling weights are stable in this regime. Physically realized oscillators would produce the same phase states as the trained digital model.

Previously we have shown in Fig. 2a that the Hopfield layers operate as traditional Hopfield layers via their energy-minimizing nature. In Fig. 4b, we further display that the proposed Hopfield layers closely match the energy trajectories of a continuous Hopfield ODE and that their energy converges to the same fixed point: the final features closely resemble each other and differ only in a few positions. The minor differences in the final states can be attributed to the discretization error of our discrete implementation. Together with the Kuramoto result, this shows that physical hardware can reproduce HoKuVit's

learned representations faithfully.

Fig. 4c shows class-token attention maps for HoKuVit and the conventional CvT baseline across five different CIFAR-10 classes. HoKuVit produces larger attention scores on a higher percentage of the images. The convergence behaviour of the Kuramoto and Hopfield layers is the likely cause: in this model, a pixel and its neighbours tend to share a value. The large proportion of pixels with an attention norm of 1 shows this. HoKuVit also attends to semantically meaningful regions, despite implementing its attention mechanism with Kuramoto oscillator components.

To assess the relative contribution of each Hopfield-based MLP and Kuramoto-based attention layer, we plot the γ LayerScale magnitudes for the corresponding layers at each stage in Fig. 4d. We can observe from the figure that the network attributes larger scaling to the Kuramoto based convolutional layers compared to the Hopfield MLP layers. Continuous Kuramoto outputs carry more information than binary Hopfield values, which likely explains the difference. The network also weights the earlier Hopfield layers more heavily, so their attractors do more of the feature extraction.

3 Discussion

HoKuVit demonstrates that physically realizable ONN primitives can be composed into a modern, hierarchical vision architecture and trained end-to-end by backpropagation through their dynamics.

3.1 Energy considerations

The ultimate purpose of this work is the design of neuromorphic hardware components with energy efficiency far surpassing that of software neural networks. While we do not simulate the ONN blocks on the device level, back-of-envelope estimates of the attainable efficiency are possible from the operation counts of Table 1 together with published, measured per-oscillator energy figures.

Table 1 splits HoKuVit's computational cost into two parts. The digital part (16.2 M FLOPs per inference) covers the conventional components: the attention projections, softmax, normalization layers, LayerScale and the classification head. The neuromorphic part (609.14 M FLOPs per inference) counts the digital-equivalent operations required to *emulate* the unrolled Hopfield and Kuramoto dynamics. In a physical realization these operations are not executed as arithmetic at all: each Hopfield retrieval and each Kuramoto convolution is performed by letting the corresponding oscillator network relax toward its attractor, and the energy cost of such a native operation is the dissipation of the participating oscillators over the relaxation time,

$$E_{\mathrm{ONN}} \;=\; N_{\mathrm{ev}} \cdot E_{\mathrm{osc}}, \qquad E_{\mathrm{osc}} \;=\; P_{\mathrm{osc}} \cdot t_{\mathrm{settle}}, \tag{1}$$

where N_{ev} is the number of oscillator-relaxation events per inference, P_{osc} the per-oscillator power of a given technology, and t_{settle} the physical settling time

of one relaxation.

Number of relaxation events. Counting one event per participating oscillator per layer evaluation (Table 2, with token maps of 16×16, 8×8 and 4×4 across the three stages), the Hopfield MLPs contribute $8h$ oscillators per token per layer, where h is the stage embedding dimension (48/64/96) and the factor 8 counts four block-diagonal sub-blocks in each of the two projections (two projections of four sub-blocks each), i.e. $256{\cdot}8{\cdot}48 + 2{\cdot}64{\cdot}8{\cdot}64 + 6{\cdot}16{\cdot}8{\cdot}96 \approx 2.4 \times 10^5$ events, while the Kuramoto token embeddings and the depthwise and pointwise attention projections contribute $C{\cdot}H{\cdot}W$ oscillators per layer, totaling $\approx 2.0{\times}10^5$ events. Altogether,

$$N_{\text{ev}} \;\approx\; 4.3 \times 10^5 \quad \text{oscillator relaxations per inference.} \tag{2}$$

Name / Location in network	**Number of oscillators**	**Interconnections (all-to-all / sparse)**	**Number of block instances in network**
Hopfield sub-block, MLP up projection	h per sub-block (h = 48/64/96 by stage)	All-to-all within sub-block (symmetric, zero diagonal)	9 layers × 4 sub-blocks
Hopfield sub-block, MLP down projection	h per sub-block	All-to-all within sub-block (symmetric, zero diagonal)	9 layers × 4 sub-blocks
Kuramoto convolutional token embedding	$C \cdot H \cdot W$ per layer	Sparse: $k \times k$ spatial neighborhood	3 (one per stage)
Kuramoto depthwise convolution (attention projections)	$C \cdot H \cdot W$ per layer	Sparse: $k \times k$ neighborhood within each channel	9 layers × 3 (Q,K,V projections)
Kuramoto pointwise convolution (attention projections)	$C \cdot H \cdot W$ per layer	All-to-all across channels at each spatial position	9 layers × 3 (Q,K,V projections)

Table 2: List of ONN blocks in HoKuVit.

Energy per relaxation event. Two physical observations fix the realistic range of E_{osc}. First, the settling time of HoKuVit's layers is short. Unlike an Ising machine, which anneals a frustrated, all-to-all coupled array toward the ground state of a rugged energy landscape over hundreds of oscillation cycles [19], HoKuVit's layers are feedforward primitives that execute a fixed, brief transient — six Hopfield update steps or fifteen Kuramoto integration steps spanning roughly one natural period — with no global frustration to resolve. A locally coupled array injection-locks to its local mean field within ∼10–15 cycles, consistent with the measured 8 ns convolution inference of a GHz-range coupled CMOS oscillator array [47]. We therefore take $t_{\text{settle}} \approx 15$ oscillation cycles. Second, in dense oscillator arrays the power budget is dominated not by the oscillators but by coupler contention currents, which grow quadratically

Table 3: Energy-per-inference scenarios for HoKuVit's ONN blocks under different oscillator technologies, using Eq. (1) with $N_{\text{ev}} \approx 4.3 \times 10^5$ and $t_{\text{settle}} \approx 15$ oscillation cycles. The digital residue of HoKuVit (16.2 M FLOPs = 8.1 M MACs at the 3.7 pJ/MAC device-physics lower bound [48]) is fixed at 30 μJ; the same accounting applied to the 64.6 M FLOPs (32.3 M MACs) of the same-configuration fully digital CvT yields ≈120 μJ. The advantage column gives the system-level energy ratio of the digital CvT to the mixed-hardware HoKuVit.

Oscillator technology	E_{osc}/ **event**	**ONN energy**	**System total**	**Advantage**
Nanoscale devices, projected (STO, scaled CMOS) [11, 7]	∼30 fJ	0.01 μJ	30 μJ	4.0×
GHz CMOS ring-oscillator convolution array, *measured* [47]	2.1 pJ	0.9 μJ	31 μJ	3.9×
65 nm ring-oscillator chip, oscillator-only [19]	60 pJ	26 μJ	56 μJ	2.1×
65 nm ring-oscillator chip, incl. sparse coupling [19]	0.2 nJ	87 μJ	117 μJ	1.0×

with connectivity: in the 48-node all-to-all ring-oscillator chip of Ref. [19], measured power falls from 105 mW at 90% graph density to 16 mW at 10% density, extrapolating to an oscillator-only floor of roughly 0.1 mW per oscillator. HoKuVit operates in the favorable limit of this scaling: its Kuramoto layers couple each oscillator to only a 3×3 neighborhood, and only the Hopfield sub-blocks (all-to-all over 48–96 oscillators) resemble the dense case.

Table 3 evaluates Eq. (1) under these assumptions for oscillator technologies spanning four orders of magnitude in E_{osc}. The most informative anchor is the coupled CMOS oscillator array of Nikonov *et al.* [30, 47]: it is fabricated hardware performing precisely the operation our Kuramoto layers implement — convolution via synchronization — at a measured 55 pJ per 26-oscillator inference, i.e. $E_{\text{osc}} \approx 2.1$ pJ per oscillator relaxation.

System-level advantage and sensitivity. Under the 3.7 pJ/MAC lower bound [48], the digital residue of HoKuVit — attention projections, softmax, normalization and the classification head — costs $E_{\text{dig}} \approx 30$ μJ per inference, while the same accounting applied to the 64.6 M FLOPs (32.3 M MACs) of Table 1 yields ≈120 μJ for the same-configuration, fully digital CvT. Clearly, the factor of 4 gain is valid if the oscillatory computations are free in terms of power — naturally, by 'free' we mean negligible compared to the digital units. ONN computations are 'free' if oscillator interconnections are free. Resistive RAM interconnections [49] certainly enable such power figures, even if the ring oscillators rely on the relatively higher-power, 65 nm technology.

We believe that the above estimate is conservative in favor of the digital baselines: the per-MAC lower bound excludes memory traffic, which typically dominates the energy budget of digital accelerators, whereas in an ONN the synaptic weights are stationary circuit parameters — resistive couplings and natural frequencies — that never have to be fetched from memory [50]. Second, once the ONN portion is nearly free, the digital residue dominates the total en-

ergy, and the overall efficiency is bounded by the 11.96% of parameters (together with the softmax and normalization operations) that remain digital. Substantial further gains therefore require moving the attention projections and normalization stages into the analog domain, or reducing their share of the workload — an architectural co-design question that we regard as an important direction for future work.

4 Methods

4.1 Hopfield layers: ONN model and training formulation

Hopfield networks [25] can be thought of as recurrent neural network models that implement associative memories by encoding stored patterns as stable attractors in an energy landscape. Each binary neuron $v_i \in \{\pm 1\}$ updates to minimize the global energy described in Eq. 3:

$$E(\mathbf{v}) = -\frac{1}{2}\sum_{i,j} w_{ij} v_i v_j, \quad w_{ij} = \frac{1}{N}\sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu, \tag{3}$$

where $\{\xi^\mu\}$ are stored patterns. Given a noisy input, the network converges to the nearest stored memory through energy minimization [25].

For the Hopfield layer to be physically realizable, three properties have to be satisfied: the weight matrix must be symmetric, its diagonal must be zero, and the update rule must be asynchronous [25]. Our Hopfield layer implementations are shown in Fig. 1b. Our implementation differs from the traditional Hopfield layer in two major ways. First, we decouple the forward and backward functions during training. For the forward function we use the traditional sign based implementation Eq. 4. While for the backward function we approximate the derivative of this function with Eq 5.

$$x_{t+1} = \text{sign}(W_h \cdot x_t) \tag{4}$$

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x}{\sigma}\right)^2\right) \tag{5}$$

This allows for better gradient propagation and avoids the exploding-gradient problem that arises when straight-through estimation is used [51]. For the asynchronous update constraint we use a dropout [52] variant called zoneout [53]. Zoneout (Eq. 6) preserves the activations of randomly selected neurons, offering a more computationally efficient route to fully-asynchronous updates.

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \tag{6}$$

The MLP layers of HoKuVit are composed of n Hopfield blocks of this structure stacked in a block-diagonal arrangement, where n is the architecture's MLP expansion ratio. Each block receives an identical copy of the input and converges

independently to its own attractor. This block-diagonal structure increases the pattern capacity per parameter of the layer: the n independent blocks of dimension h collectively store on the order of $n \cdot \alpha h$ patterns [54], the same order as a single Hopfield network of dimension hn, but with a factor of n fewer parameters, since the block weight matrices scale as $n \cdot h^2$ rather than $(hn)^2$. This trade-off favors parameter efficiency at the cost of cross-block pattern interactions.

4.2 Kuramoto convolutional layers

Traditional convolutional neural networks rely on static weight matrices to transform input features, whereas biological neural systems exhibit rich temporal dynamics through oscillatory activity. Inspired by the Kuramoto model of coupled oscillators [55], we propose a neuromorphic convolution layer in which feature computation emerges from the synchronization dynamics of coupled phase oscillators. Our implementation addresses two design challenges: preventing trivial convergence to uniform phase distributions, and ensuring computational efficiency for deep learning. We achieve both through a formulation in which input features act as initial conditions of the layer.

The classical Kuramoto model describes N coupled oscillators with phases $\theta_i(t) \in [-\pi, \pi]$ evolving according to:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\theta_j - \theta_i) \tag{7}$$

where ω_i is the natural frequency of oscillator i and K is the coupling strength. This model exhibits a phase transition: when K exceeds a critical threshold K_c, oscillators spontaneously synchronize despite heterogeneous natural frequencies [44].

We extend the Kuramoto model to convolutional 2D filters suitable for image processing. For a feature map with C channels and spatial dimensions $H \times W$, we implement two layer variants: a *depthwise* variant, in which each channel c couples only to its own spatial neighborhood and learns a single channel-wise natural frequency ω_c, and a *pointwise* variant, in which each spatial location couples across channels and learns a location-specific natural frequency. When the number of input and output channels differs, a learned traditional convolutional 2D filter generates the additional channels needed for the initial phase state $\theta(0)$, while the remaining channels are inherited directly from a copy of the input; the natural frequencies ω_c remain independent, freely-learned parameters unrelated to this projection.

For both variants, oscillator phases $\theta_{c,h,w}(t)$ evolve according to the following Equation 8:

$$\frac{d\theta_{c,h,w}}{dt} = \omega_c + \sum_{c',h',w'} K_{c,c'}^{(h-h',w-w')} \sin(\theta_{c',h',w'} - \theta_{c,h,w}) \tag{8}$$

where $K_{c,c'}^{(\Delta h,\Delta w)}$ is a learned convolutional kernel acting only within the

channel or spatial location according to the variant type. This way we are restricting coupling to the spatial neighborhood defined by the kernel support, allowing for more diversity and limiting computational demand by synchronizing in fewer steps, due to limiting oscillator array interactions. In the forward pass this ODE is discretized by explicit Euler integration, $\theta^{(s+1)} = \theta^{(s)} + \Delta t \left(\omega_c + \text{Coupling}^{(s)}\right)$, with the step size Δt given in Table 4.

Parameter	**Value**
Architecture	
Stages	3
Embedding dimensions	48, 64, 96
Number of layers	1, 2, 6
Attention heads	1, 2, 3
MLP expansion ratio	4
Hopfield & Kuramoto	
Hopfield update steps	6
Hopfield zoneout probability	0.3
Kuramoto (ONN) update steps	15
ONN minimum frequency	1.75
ONN mean initial frequency	3.0
ONN time step (Δt)	0.125
Training	
Total epochs	100
Batch size	128
Gradient accumulation steps	2
Base learning rate	1×10^{-3}
Weight decay (non-Kuramoto)	0.005
Weight decay (Kuramoto)	0.02
Hopfield L1 penalty λ	0.005
Warmup epochs	5
Regularization	
Dropout probability	0.0
Attention dropout	0.0
Stochastic depth (drop path)	0.1
Mixup α	0.8
CutMix α	1.0
Mixup probability	1.0
Mixup/CutMix switch probability	0.5
Label smoothing ϵ	0.1

Table 4: Hyperparameter configuration for HoKuVit training on CIFAR-10.

During the forward pass, this dynamics is integrated for a fixed number of Euler steps. Because materializing the full pairwise $\sin(\theta_i - \theta_j)$ tensor is memory-prohibitive at the spatial resolutions used in HoKuVit, we apply the trigonometric identity $\sin(\alpha-\beta) = \sin\alpha\cos\beta - \cos\alpha\sin\beta$ to rewrite the coupling sum as a pair of ordinary convolutions:

$$\text{Coupling}_{c,h,w} = \cos(\theta_{c,h,w}) \cdot (K_c * \sin(\boldsymbol{\theta}))_{c,h,w} - \sin(\theta_{c,h,w}) \cdot (K_c * \cos(\boldsymbol{\theta}))_{c,h,w} \quad (9)$$

where $*$ denotes 2D convolution. This formulation requires only two convolution operations per integration step regardless of spatial resolution, making the layer computationally tractable for deep networks.

To decode the layer's output, we take into consideration the noisy nature of the hardware substrate by reading out each oscillator's mean phase relative to the population, rather than its instantaneous phase alone. Specifically, we compute the mean-field expansion of $\overline{\sin(\theta_{c,h,w} - \theta_{\cdot,h,w})}$ averaged over channels at each spatial location:

$$y_{c,h,w} = \sin(\theta_{c,h,w}) \cdot \overline{\cos(\theta)}_{h,w} - \cos(\theta_{c,h,w}) \cdot \overline{\sin(\theta)}_{h,w} \quad (10)$$

where the overline denotes the mean across the channel dimension at each spatial position. This readout is invariant to a global phase rotation of all oscillators, making it robust to the specific number of integration steps used at inference and to phase noise introduced by physical hardware.

4.3 Assembling the vision transformer and role of the non-ONN blocks

Vision transformers process images by dividing them into patches and embedding them as tokens. The core mechanism is multi-head self-attention (MSA), which computes relationships between all tokens through query ($\mathbf{Q}$), key ($\mathbf{K}$), and value ($\mathbf{V}$) projections [56, 33]:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \quad (11)$$

where d_k is the key dimension. A major limitation of the Scaled Dot-Product Attention is the phenomenon where hidden states with larger magnitude can dominate the attention score irrespective of their semantic content, causing a small number of tokens or patches to absorb the majority of the attention weight. This not only starves the remaining tokens of gradient signal during backpropagation but, in the context of a physically realizable network, forces the oscillator-derived query and key projections to encode information in their magnitude rather than their phase relationships. Scaled dot product attention is therefore poorly matched to a substrate whose physical observable is phase, not amplitude, and would require additional analog circuitry to track and normalize signal magnitude reliably. We instead opt to use scaled cosine attention [27] (Eq. 12). Scaled cosine attention discards magnitude information entirely and computes attention purely from the angular relationship between queries and keys. This choice is motivated by two reasons: it stabilizes training by preventing any single token from monopolizing attention weight, and it aligns the attention mechanism with the phase-based representations natively produced

by the Kuramoto convolutional layers, removing the need for a hardware-side magnitude normalization stage.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{Q}{\|Q\|_2} \cdot \left(\frac{K}{\|K\|_2}\right)^T \cdot \min\left(e^{\lambda}, 100\right)\right) V \quad (12)$$

For multi-head attention with h heads, this is performed in parallel across learned linear projections, allowing the model to attend to different representation subspaces [56] (in HoKuVit, cosine attention, Eq. 12, is used inside each head in place of scaled dot-product attention):

$$\text{MSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O \quad (13)$$

where $\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$ and $\mathbf{W}^O$ is the output projection. Each transformer block combines multi-head self-attention (MSA) with a feed-forward network (FFN), as shown in Eq. 14 and Eq. 15. Rather than combining each sublayer with its residual branch through simple addition, we adopt LayerScale [28], which introduces a learnable, per-channel scaling factor γ_{l1}, γ_{l2} on the output of each sublayer before it is added to the residual stream:

$$\mathbf{y} = \mathbf{x} + \gamma_{l1} \odot \text{MSA}(\text{LayerNorm}(\mathbf{x})) \quad (14)$$

$$\mathbf{z} = \mathbf{y} + \gamma_{l2} \odot \text{FFN}(\text{LayerNorm}(\mathbf{y})) \quad (15)$$

where γ_{l1}, γ_{l2} are the scaling factors for layer l and $\odot$ denotes element-wise multiplication. As shown by Touvron et al. [28], this per-channel scaling disperses each sublayer's contribution to the residual stream more evenly across the network depth, improving training stability in deep transformers. This property is additionally advantageous in our setting: because each transformer block mixes conventional attention projections with Kuramoto- and Hopfield-based components of differing dynamical character, LayerScale allows the network to learn an explicit, per-layer weighting of how strongly each substrate contributes to the output, which we analyze directly in Section 2.5 via the learned γ values shown in Fig. 4d. This even dispersal is also practically relevant for hardware deployment, as it avoids a single dominant pathway whose failure or noise sensitivity would disproportionately degrade the network's output.

Pure vision transformers lack the inductive biases of convolutional networks, since they make no rigid assumptions about visual structure [33, 57]. This allows them to scale well with data and capture global context effectively, but it also means they require larger datasets and longer training schedules to reach competitive accuracy. Convolutional vision transformers offer a middle ground between ViTs and CNNs, retaining spatial structure in the patch embeddings and between stages while preserving the global modeling capacity of self-attention [29]. CvT achieves this through two modifications: overlapping strided convolutions replace the linear token projections, yielding hierarchical, multi-scale

downsampling that preserves spatial relationships; and depthwise separable convolutions replace the query/key/value projections in self-attention, embedding local receptive fields directly into the attention mechanism.

From a neuromorphic hardware standpoint the use of a CvT architecture is advantageous for two main reasons: First, the multi-head self-attention mechanism of any ViT architecture allows for the distribution of parameters across multiple heads, allowing for the use of smaller sub-components. Second, the additional inductive bias of the convolutional structure reduces training time, which is essential for offsetting the additional computational cost introduced by the recurrent nature of the Kuramoto and Hopfield layers.

4.4 Training configuration

Our HoKuVit architecture is based on the CvT backbone [29], augmented with Hopfield associative-memory layers and Kuramoto oscillatory layers, with a hierarchical three-stage design in which each stage progressively reduces spatial resolution while increasing feature dimensionality. The full hyperparameter configuration is given in Table 4.

For data augmentation we utilize random cropping with 4-pixel padding and horizontal flipping. Inputs were normalized using CIFAR-10 statistics. Regularization is applied by Mixup [58] ($\alpha = 0.8$) and CutMix [59] ($\alpha = 1.0$), switching between them with probability 0.5. Label smoothing ($\epsilon = 0.1$) was used to prevent overconfident predictions, and stochastic depth [60] with drop-path probability $p = 0.1$ provided additional regularization.

We used the NAdam optimizer [61] for training, which combines Nesterov momentum with adaptive learning rates. The base learning rate was scaled with the effective batch size, $\mathrm{lr} = \mathrm{lr}_{\mathrm{base}} \times (\mathrm{batch}_{\mathrm{effective}}/256)$, where the effective batch size accounts for gradient-accumulation steps. Training began with a linear warmup over 5 epochs from 10^{-6} to the target learning rate, followed by cosine annealing [62].

We relied on the PyTorch [63] native automatic mixed precision (AMP) to reduce memory consumption and accelerate training, with gradient accumulation over 2 steps to simulate larger effective batch sizes without exceeding GPU memory.

4.5 Evaluation protocol

Model performance was evaluated on the CIFAR-10 test set every epoch during training. We report top-1 accuracy as the primary metric; the best checkpoint was selected by accuracy on the test set, and final results correspond to this best-performing snapshot rather than the last epoch. For fair comparison, we used official implementations where available and otherwise trained under identical conditions using our own pipeline: the same data-augmentation pipeline, computational budget, and hardware. Gradient norms were optionally monitored to diagnose optimization issues, though this was disabled for production runs due to overhead. The hierarchical architecture, combined with the iterative

dynamics of the Hopfield and Kuramoto layers, results in substantially longer training times than standard transformers; the convolutional inductive biases of CvT mitigate this by enabling faster convergence with limited data. As a result, HoKuVit trains fully within 12 hours on a single NVIDIA A100 40GB GPU.

We note that selecting the best checkpoint by test-set accuracy may bias the reported figures slightly upward: each checkpoint evaluation on the 10k-image test set carries a binomial sampling uncertainty of $\sqrt{p(1-p)/n} \approx \sqrt{0.8 \cdot 0.2/10^4} \approx 0.4$ percentage points, and maximizing over the checkpoints of the accuracy plateau can therefore inflate the result by up to a few tenths of a percentage point. Since the identical protocol was applied to all models trained in this work, the relative comparisons in Table 1 are unaffected, and the margin over prior ONN results far exceeds this potential bias.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[3] Catherine D Schuman, Shruti R Kulkarni, Maryam Parsa, J Parker Mitchell, Prasanna Date, and Bill Kay. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, 2(1):10–19, 2022.

[4] Adnan Mehonic and Anthony J Kenyon. Brain-inspired computing needs a master plan. *Nature*, 604(7905):255–260, 2022.

[5] Catherine D Schuman, Thomas E Potok, Robert M Patton, J Douglas Birdwell, Mark E Dean, Garrett S Rose, and James S Plank. A survey of neuromorphic computing and neural networks in hardware. *arXiv preprint arXiv:1705.06963*, 2017.

[6] Mike Davies, Andreas Wild, Garrick Orchard, Yulia Sandamirskaya, Gabriel A Fonseca Guerra, Prasad Joshi, Philipp Plank, and Sumedh R Risbud. Advancing neuromorphic computing with Loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–928, 2021.

[7] Miguel Romera, Philippe Talatchian, Sumito Tsunegi, Flavio Abreu Araujo, Vincent Cros, Paolo Bortolotti, Juan Trastoy, Kay

Yakushiji, Akio Fukushima, Hitoshi Kubota, et al. Vowel recognition with four coupled spin-torque nano-oscillators. *Nature*, 563(7730):230–234, 2018.

[8] Markus Graber and Klaus Hofmann. An enhanced 1440 coupled cmos oscillator network to solve combinatorial optimization problems. In *2023 IEEE 36th International System-on-Chip Conference (SOCC)*, pages 1–6. IEEE, 2023.

[9] Tamas Rudner, Wolfgang Porod, and Gyorgy Csaba. Design of oscillatory neural networks by machine learning. *Frontiers in Neuroscience*, 18:1307525, 2024.

[10] Kwantae Kim, Chang Gao, Rui Graça, Ilya Kiselev, Hoi-Jun Yoo, Tobi Delbruck, and Shih-Chii Liu. A 23-μw keyword spotting ic with ring-oscillator-based time-domain feature extraction. *IEEE Journal of Solid-State Circuits*, 57(11):3298–3311, 2022.

[11] Gyorgy Csaba and Wolfgang Porod. Coupled oscillators for computing: A review and perspective. *Applied physics reviews*, 7(1), 2020.

[12] Aida Todri-Sanial, Corentin Delacour, Madeleine Abernot, and Filip Sabo. Computing with oscillators from theoretical underpinnings to applications and demonstrators. *Npj unconventional computing*, 1(1):14, 2024.

[13] Mirko Prezioso, Farnood Merrikh-Bayat, Brian D Hoskins, Gina C Adam, Konstantin K Likharev, and Dmitri B Strukov. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature*, 521(7550):61–64, 2015.

[14] Weier Wan, Rajkumar Kubendran, Clemens Schaefer, Sukru Burc Eryilmaz, Wenqiang Zhang, Dabin Wu, Stephen Deiss, Priyanka Raina, He Qian, Bin Gao, et al. A compute-in-memory chip based on resistive random-access memory. *Nature*, 608:504–512, 2022.

[15] Stefano Ambrogio, Pritish Narayanan, Atsuya Okazaki, Andrea Fasoli, Charles Mackin, Kohji Hosokawa, Akiyo Nomura, Takeo Yasuda, An Chen, Andrew Friz, et al. An analog-ai chip for energy-efficient speech recognition and transcription. *Nature*, 620(7975):768–775, 2023.

[16] Frank C Hoppensteadt and Eugene M Izhikevich. Oscillatory neurocomputers with dynamic connectivity. *Physical Review Letters*, 82(14):2983, 1999.

[17] Andrei Slavin and Vasil Tiberkevich. Nonlinear auto-oscillator theory of microwave generation by spin-polarized current. *IEEE Transactions on Magnetics*, 45(4):1875–1918, 2009.

[18] I Mahboob and H Yamaguchi. Bit storage and bit flip operations in an electromechanical oscillator. *Nature nanotechnology*, 3(5):275–279, 2008.

[19] Hao Lo, William Moy, Hanzhao Yu, Sachin S. Sapatnekar, and Chris H. Kim. An Ising solver chip based on coupled ring oscillators with a 48-node all-to-all connected array architecture. *Nature Electronics*, 6:771–778, 2023.

[20] Olivier Maher, Manuel Jiménez, Corentin Delacour, Nele Harnack, Juan Núñez, María J Avedillo, Bernabé Linares-Barranco, Aida Todri-Sanial, Giacomo Indiveri, and Siegfried Karg. A cmos-compatible oscillation-based vo2 ising machine solver. *Nature Communications*, 15(1):3334, 2024.

[21] Takeru Miyato, Sindy Löwe, Andreas Geiger, and Max Welling. Artificial Kuramoto oscillatory neurons. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.13821.

[22] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, et al. Hopfield networks is all you need. *International Conference on Learning Representations (ICLR)*, 2021.

[23] Dmitry Krotov and John Hopfield. Large associative memory problem in neurobiology and machine learning. *arXiv preprint arXiv:2008.06996*, 2020.

[24] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

[25] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[26] Frank C Hoppensteadt and Eugene M Izhikevich. Pattern recognition via synchronization in phase-locked loop neural networks. *IEEE Transactions on Neural Networks*, 11(3):734–738, 2000.

[27] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.

[28] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 32–42, 2021.

[29] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22–31, 2021.

[30] Dmitri E Nikonov, Peter Kurahashi, James S Ayers, Hai Li, Telesphor Kamgaing, Georgios C Dogiamis, Hyung-Jin Lee, Yongping Fan, and IA Young.

Convolution inference via synchronization of a coupled cmos oscillator array. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 6(2):170–176, 2020.

[31] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, 28, 2015.

[32] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in neural information processing systems*, 29, 2016.

[33] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[34] Yefei He, Zhenyu Lou, Luoming Zhang, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Bivit: Extremely compressed binary vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5651–5663, 2023.

[35] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[36] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits. *ATT Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2(6):18, 1998.

[37] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[38] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*, 2021.

[39] Xinyu Shi, Zecheng Hao, and Zhaofei Yu. Spikingresformer: Bridging resnet and vision transformer in spiking neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5610–5619, 2024.

[40] Dmitri E Nikonov, Gyorgy Csaba, Wolfgang Porod, Tadashi Shibata, Danny Voils, Dan Hammerstrom, Ian A Young, and George I Bourianoff. Coupled-oscillator associative memory array operation for pattern recognition. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 1:85–93, 2015.

[41] Minsuk Koo, MR Pufall, Yong Shim, Anthony B Kos, Gyorgy Csaba, Wolfgang Porod, WH Rippard, and Kaushik Roy. Distance computation based on coupled spin-torque oscillators: Application to image processing. *Physical review applied*, 14(3):034001, 2020.

[42] Juan A Acebrón, Luis L Bonilla, Conrad J Pérez Vicente, Félix Ritort, and Renato Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Reviews of modern physics*, 77(1):137–185, 2005.

[43] Yoshiki Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In *International Symposium on Mathematical Problems in Theoretical Physics*, volume 39 of *Lecture Notes in Physics*, pages 420–422. Springer, 1975.

[44] Steven H Strogatz. From kuramoto to crawford: exploring the onset of synchronization in populations of coupled oscillators. *Physica D: Nonlinear Phenomena*, 143(1-4):1–20, 2000.

[45] Florian Dörfler and Francesco Bullo. Synchronization in complex networks of phase oscillators: A survey. *Automatica*, 50(6):1539–1564, 2014.

[46] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[47] D. E. Nikonov, P. Kurahashi, J. S. Ayers, H.-J. Lee, Y. Fan, and I. A. Young. A coupled CMOS oscillator array for 8ns and 55pJ inference in convolutional neural networks. *arXiv preprint arXiv:1910.11803*, 2019.

[48] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pages 10–14. IEEE, 2014.

[49] Wooseok Choi, Thomas Van Bodegraven, Jelle Verest, Olivier Maher, Donato F Falcone, Antonio La Porta, Daniel Jubin, Bert J Offrein, Siegfried Karg, Valeria Bragaglia, et al. Hardware implementation of ring oscillator networks coupled by beol integrated reram for associative memory tasks. In *2025 IEEE International Memory Workshop (IMW)*, pages 1–4. IEEE, 2025.

[50] Abu Sebastian, Manuel Le Gallo, Riduan Khaddam-Aljameh, and Evangelos Eleftheriou. Memory devices and applications for in-memory computing. *Nature nanotechnology*, 15(7):529–544, 2020.

[51] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[52] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[53] David Krueger, Tegan Maharaj, János Kramár, Mohammad Pezeshki, Nicolas Ballas, Nan Rosemary Ke, Anirudh Goyal, Yoshua Bengio, Aaron Courville, and Chris Pal. Zoneout: Regularizing rnns by randomly preserving hidden activations. *arXiv preprint arXiv:1606.01305*, 2016.

[54] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical review letters*, 55(14):1530, 1985.

[55] Yoshiki Kuramoto. *Chemical Oscillations, Waves, and Turbulence*, volume 19 of *Springer Series in Synergetics*. Springer, Berlin, Heidelberg, 1984.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[57] Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International conference on machine learning*, pages 2286–2296. PMLR, 2021.

[58] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[59] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[60] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[61] Timothy Dozat. Incorporating nesterov momentum into adam. In *ICLR Workshop*, 2016.

[62] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[63] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Funding

This work has received funding from the European Union's Horizon EU research and innovation programme under grant agreement No. 101092096 (PHASTRAC).

Acknowledgements

We are grateful to our PHASTRAC project partners especially to Prof. Aida Todri-Sanial (TU Eindhoven) for discussions.

Author contributions statement

G.C. and A.H. conceived the idea. T.E. designed and wrote the code, performed the simulations, and drafted the manuscript. All authors analyzed the data and contributed to the discussion and revision of the manuscript. G.C. acquired the funding.

Competing interests

The authors declare no competing interests.

Data Availability Statement

All models, the machine learning framework developed for this study can be found on GitHub. https://github.com/TomasDelaney/HoKuVit