EXACT AND TRAINABLE GROWTH OF STATEFUL SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks (SNNs) typically rely on fixed architectures whose capacity is chosen before training. Insufficient width limits representation, whereas excessive width introduces unnecessary parameters and computation. This problem is especially challenging in SNNs because structural changes can disrupt their temporal dynamics. We introduce an exact and trainable growth mechanism for stateful SNNs. The proposed paired basis adds new neurons to a partially trained network while preserving the downstream temporal computation and output logits at every simulation step. Unlike conventional zero readout, which preserves the current output but blocks the immediate task gradient to the incoming parameters of newborn neurons, our construction uses identical neuron pairs whose forward contributions cancel exactly while their backward signals remain nonzero and opposing. Growth is implemented as an atomic transaction that expands the model, migrates Adam state, verifies the required invariants, and resumes surrogate gradient training. We establish temporal function preservation, conditional immediate trainability, preservation of the optimizer state for existing parameters, and layer-wise composition, providing a principled mechanism for expanding stateful SNNs without discarding their learned temporal computation.

1 INTRODUCTION

Spiking neural networks (SNNs) represent information through discrete spikes and recurrent neuronal states, providing a temporally stateful model of computation (Maass, 1997; Gerstner et al., 2014; Roy et al., 2019). Surrogate-gradient methods make such networks trainable with backpropagation through time by replacing the derivative of the spike function during the backward pass (Zenke & Ganguli, 2018; Neftci et al., 2019; Zenke & Vogels, 2021). Nevertheless, the physical architecture of an SNN is usually fixed before optimization, so its width must be chosen without knowing how much capacity the task will require. Here, *physical width* refers to explicitly instantiated model coordinates rather than masked or latent capacity, and does not imply a particular neuromorphic hardware realization.

Capacity adaptation is well studied through constructive learning, architecture search, and dynamic sparsity (Fahlman & Lebiere, 1989; Elsken et al., 2019; Evci et al., 2020). These methods primarily decide which architecture or connectivity pattern to train. A different problem arises when a partially trained stateful network must be physically expanded without discarding its learned temporal computation. Function-preserving transformations developed for conventional networks (Chen et al., 2016; Wei et al., 2016) do not by themselves address this setting. An SNN carries membrane and spike trajectories across time, and newly added neurons must also receive useful learning signals after expansion.

The central challenge is to maintain functional stability while preserving the plasticity of newly introduced neurons. Initializing newborn readout weights to zero preserves the current output but blocks the immediate input-side gradient of the new neurons.

We implement this construction as an atomic transaction that is aware of the optimizer state. The transaction expands parameter tensors, migrates the existing optimizer state, verifies temporal invariants, and commits the expanded model only after successful verification. Controlled experiments confirm exact preservation and immediate trainability, while progressive growth experiments characterize the resulting accuracy and resource trade-offs across multiple SNN settings.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Our contributions are twofold:

- **Exact and immediately trainable growth without rewriting existing coordinates.** Existing function-preserving widening schemes can break symmetry by redistributing or perturbing duplicated units, but doing so generally modifies parameters that already belong to the trained network (Chen et al., 2016; Yu et al., 2026). We instead impose a stricter growth constraint: all existing parameter coordinates and temporal states remain unchanged at the growth instant. Under this constraint, the proposed paired basis preserves the complete downstream temporal computation while providing nonzero and opposing gradients to the incoming parameters of newborn neurons under nondegenerate conditions.
- **Physical expansion consistent with ongoing training.** Classical network transformations such as Net2Net and Network Morphism enlarge trained networks while preserving their represented function (Chen et al., 2016; Wei et al., 2016), while recent work on progressive width expansion also highlights the need to preserve signals while breaking symmetry among duplicated units (Yu et al., 2026). We extend this perspective to stateful spiking computation by integrating physical expansion with the ongoing optimization process. The proposed transaction preserves existing parameters and Adam state, initializes the optimizer state of newly introduced coordinates, and composes across hidden layers. Experiments verify the predicted mechanism and examine its behavior under repeated growth, different capacity allocation strategies, and additional SNN configurations.

2 Related Work

Network expansion with function preservation. Constructive neural networks increase model capacity during optimization rather than fixing the entire architecture in advance. Cascade-Correlation progressively introduces hidden units according to the residual learning signal (Fahlman & Lebiere, 1989), while later methods grow neural architectures by splitting neurons, adding layers, or applying dedicated procedures for structural growth (Wu et al., 2019; 2020; Yuan et al., 2023; Yao et al., 2024). These approaches establish progressive growth as an alternative to repeatedly training independently sized models.

A closely related line of work studies transformations that enlarge a trained network while preserving the function it already represents. Net2Net widens or deepens a trained network while preserving the mapping from inputs to outputs (Chen et al., 2016), and Network Morphism extends this principle to a broader family of architectural transformations (Wei et al., 2016). Recent work on progressive width expansion further identifies signal preservation and symmetry breaking as distinct requirements when duplicated units are introduced (Yu et al., 2026). Function-preserving duplication does not necessarily require duplicated units to remain symmetric. For example, uneven splitting can redistribute a donor's outgoing contribution between the original neuron and its clone, preserving the represented function while assigning different backward scales to the two copies (Chen et al., 2016; Yu et al., 2026). Such constructions, however, modify pre-existing parameter coordinates during expansion. Our setting imposes a stricter constraint: growth should introduce new physical coordinates without rewriting parameters or optimizer state already accumulated by the trained network. The paired-basis construction is designed for this constraint. Our method builds on this perspective in the context of stateful spiking computation. Rather than preserving only a static mapping, we preserve the complete temporal predictor while maintaining an immediate learning path into the newly introduced hidden parameters.

Adaptive network capacity. Neural architecture search adapts model structure through reinforcement learning, evolutionary optimization, continuous relaxation, or supernetworks with shared weights (Zoph & Le, 2017; Liu et al., 2019; Real et al., 2019; Elsken et al., 2019; White et al., 2023). These methods search over candidate architectures, often within predefined search spaces. Approaches that focus specifically on width provide a more targeted alternative. Adaptive Width Neural Networks, for example, learn a soft ordering over hidden neurons and optimize the resulting effective width (Errica et al., 2026). Pruning and dynamic sparsity approach capacity adaptation from the opposite direction by removing or reallocating parameters from an initially larger model (Blalock et al., 2020; Evci et al., 2020; He & Xiao, 2024).

Together, these methods demonstrate the value of adapting capacity rather than treating architecture exclusively as a fixed hyperparameter. Our method addresses a complementary problem, physically expanding an already partially optimized stateful network while preserving the computation represented immediately before the structural change. The expansion can be triggered by a predefined schedule or by an allocation policy driven by validation performance.

Structural adaptation in SNNs. Structural optimization has also become increasingly relevant to SNNs. Architecture search methods designed for spiking computation identify structures that are better suited to SNN dynamics than architectures inherited directly from conventional ANNs (Na et al., 2022; Putra & Shafique, 2025). Other studies adapt SNN structure through pruning, sparse connectivity, synaptic regeneration, or dynamic sparse training (Chen et al., 2022; Han et al., 2022a;b; Huang et al., 2023). Growing SNN methods have additionally introduced new units in online or continual learning settings (Wang et al., 2014; Han et al., 2023).

These directions establish that SNN structure can evolve during learning, but they primarily address architecture search, sparsity adaptation, continual expansion, or training efficiency. Our method focuses on a different question of how new physical neurons can be inserted into a partially trained stateful SNN without changing its downstream temporal computation while retaining an immediate optimization path for the newborn parameters.

3 Exact and Trainable Physical Growth

We consider physical width expansion of a partially trained stateful spiking neural network. Unlike choosing a fixed physical architecture before optimization, a growth event modifies a network whose parameters, membrane states, spike trajectories, and optimizer history already encode learned temporal computation. A valid growth operator should therefore introduce new physical capacity without changing the computation present at the growth instant, while allowing the newborn neurons to participate in optimization immediately afterward.

3.1 Stateful SNN Growth Problem

Consider an input sequence $\mathbf{x}^{1:T} = (\mathbf{x}^1, \dots, \mathbf{x}^T)$ and a hidden layer of D leaky integrate-and-fire (LIF) neurons. Throughout this paper, we study temporally stateful feed-forward LIF SNNs, where temporal state is carried by the membrane dynamics rather than by learned recurrent synaptic connections. Its vectorized dynamics are

$$\mathbf{i}^t = \mathbf{W}_{\mathrm{in}}\mathbf{x}^t + \mathbf{b}_{\mathrm{in}}, \qquad \tilde{\mathbf{u}}^t = \boldsymbol{\beta} \odot \mathbf{u}^{t-1} + \mathbf{i}^t, \qquad \mathbf{s}^t = H(\tilde{\mathbf{u}}^t - \mathbf{V}_{\mathrm{th}}), \qquad \mathbf{u}^t = \tilde{\mathbf{u}}^t - \mathbf{V}_{\mathrm{th}}\,\mathrm{sg}(\mathbf{s}^t), \tag{1}$$

where $\mathbf{u}^t$ and $\mathbf{s}^t$ denote membrane and spike states, respectively, and $\mathrm{sg}(\cdot)$ denotes the stop-gradient operation. During backpropagation, the derivative of H is replaced by a bounded surrogate derivative.

The temporal logits, sequence prediction, and training objective are

$$\mathbf{z}^t = \mathbf{W}_{\mathrm{out}}\mathbf{s}^t + \mathbf{b}_{\mathrm{out}}, \qquad \overline{\mathbf{z}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{z}^t, \qquad \mathcal{L} = \ell(\overline{\mathbf{z}}, y). \tag{2}$$

Because the membrane state evolves recursively through time, a valid growth operation must preserve the temporal computation rather than only the final prediction.

Suppose training reaches an intermediate state with parameters $\boldsymbol{\theta}$, optimizer state $\mathcal{O}$, and hidden width D. A growth event introduces K newborn pairs,

$$\mathcal{G} : (\boldsymbol{\theta}, \mathcal{O}, D) \longmapsto (\boldsymbol{\theta}^+, \mathcal{O}^+, D + 2K). \tag{3}$$

Let E denote the coordinate injection from the original tensors into their expanded counterparts, and let $\boldsymbol{\theta}^{\mathrm{in}}_{\mathrm{new}}$ denote the incoming parameters of the newborn neurons. We require

$$\mathbf{z}^{t,+} = \mathbf{z}^t, \qquad \mathbf{u}^{t,+}_{\mathrm{old}} = \mathbf{u}^t_{\mathrm{old}}, \qquad \mathbf{s}^{t,+}_{\mathrm{old}} = \mathbf{s}^t_{\mathrm{old}}, \qquad \forall t, \tag{4}$$

$$\boldsymbol{\theta}^+_{E(j)} = \boldsymbol{\theta}_j, \qquad \mathcal{O}^+_{E(j)} = \mathcal{O}_j, \qquad \forall j \in \mathcal{I}_{\mathrm{old}}. \tag{5}$$

In addition, the newborn parameters should retain an immediate learning path, with $\left\|\nabla_{\boldsymbol{\theta}^{\mathrm{in}}_{\mathrm{new}}}\mathcal{L}\right\|_2 > 0$ whenever the gradient is nondegenerate. Together, these conditions preserve the temporal computation

and accumulated optimization history while introducing new capacity without rewriting existing coordinates and retaining an immediate learning path for the newborn parameters.

A simple way to preserve the current output is to initialize the newborn readout to zero,

$$\mathbf{z}^{t,+} = \mathbf{W}_{\text{out}}\mathbf{s}^t + \mathbf{W}_{\text{new}}\mathbf{s}^t_{\text{new}} + \mathbf{b}_{\text{out}}, \qquad \mathbf{W}_{\text{new}} = \mathbf{0}. \tag{6}$$

This gives $\mathbf{z}^{t,+} = \mathbf{z}^t$. However, letting $\boldsymbol{\delta}^t = \partial\mathcal{L}/\partial\mathbf{z}^{t,+}$ and $\mathbf{J}^t_j = \partial s^t_{j,\text{new}}/\partial\boldsymbol{\theta}^{\text{in}}_{j,\text{new}}$, the corresponding gradient is

$$\nabla_{\boldsymbol{\theta}^{\text{in}}_{j,\text{new}}}\mathcal{L} = \sum_{t=1}^{T}(\mathbf{J}^t_j)^\top \mathbf{W}^\top_{\text{new},j}\boldsymbol{\delta}^t = \mathbf{0}. \tag{7}$$

Zero readout therefore preserves the current computation but removes the immediate task gradient to the incoming parameters of the newborn neurons. The paired construction below avoids this trade-off.

3.2 Exact Paired-Basis Growth

We introduce newborn neurons in symmetric pairs whose forward contributions cancel at birth while their backward signals remain nonzero, as illustrated in Figure 1.

For a newborn pair (p, q), a donor neuron r is selected from the existing hidden layer. Both neurons copy its incoming parameters and current membrane state,

$$\mathbf{w}^{(0)}_p = \mathbf{w}^{(0)}_q = \mathbf{w}_r, \qquad b^{(0)}_p = b^{(0)}_q = b_r, \qquad u^0_p = u^0_q = u^0_r. \tag{8}$$

Identical initialization and deterministic LIF dynamics imply $u^t_p = u^t_q$ and $s^t_p = s^t_q$ for all t.

We represent the pair through difference and sum coordinates,

$$d^t_{pq} = c_{pq}(s^t_p - s^t_q), \qquad r^t_{pq} = c_{pq}(s^t_p + s^t_q), \qquad \Delta\mathbf{z}^t_{pq} = \mathbf{V}_{pq}d^t_{pq} + \mathbf{U}_{pq}r^t_{pq}, \qquad c_{pq} > 0. \tag{9}$$

At birth, we set $\mathbf{V}^{(0)}_{pq} \neq \mathbf{0}$ and $\mathbf{U}^{(0)}_{pq} = \mathbf{0}$. Since the cloned neurons satisfy $s^t_p = s^t_q$, their difference coordinate is zero, and each newborn pair contributes no change to the temporal logits. For K independently constructed pairs,

$$\mathbf{z}^{t,+} = \mathbf{z}^t + \sum_{k=1}^{K}\Delta\mathbf{z}^{t,(0)}_k = \mathbf{z}^t, \qquad \forall t. \tag{10}$$

Exact forward cancellation does not remove the backward learning path. Let $\boldsymbol{\delta}^t = \partial\mathcal{L}/\partial\mathbf{z}^{t,+}$. Differentiating Equation 9 gives

$$\begin{aligned}\mathbf{g}^t_p &= c_{pq}\left(\mathbf{V}_{pq} + \mathbf{U}_{pq}\right)^\top \boldsymbol{\delta}^t,\\ \mathbf{g}^t_q &= c_{pq}\left(-\mathbf{V}_{pq} + \mathbf{U}_{pq}\right)^\top \boldsymbol{\delta}^t.\end{aligned} \tag{11}$$

Because $\mathbf{U}^{(0)}_{pq} = \mathbf{0}$ at birth,

$$\mathbf{g}^{t,(0)}_p = -\mathbf{g}^{t,(0)}_q = c_{pq}\left(\mathbf{V}^{(0)}_{pq}\right)^\top \boldsymbol{\delta}^t. \tag{12}$$

Let $\boldsymbol{\vartheta}_p$ and $\boldsymbol{\vartheta}_q$ denote corresponding incoming parameter blocks. Their identical temporal trajectories give the same BPTT Jacobian at birth. Consequently,

$$\nabla_{\boldsymbol{\vartheta}_p}\mathcal{L} = \mathbf{C}_\vartheta, \qquad \nabla_{\boldsymbol{\vartheta}_q}\mathcal{L} = -\mathbf{C}_\vartheta, \qquad \mathbf{C}_\vartheta = \sum_{t=1}^{T}(\mathbf{J}^t_\vartheta)^\top c_{pq}\left(\mathbf{V}^{(0)}_{pq}\right)^\top \boldsymbol{\delta}^t. \tag{13}$$

Whenever $C_\vartheta \neq 0$, the newborn incoming parameters receive nonzero and opposing gradients. Under a plain gradient-descent step with learning rate η,

$$\vartheta^{(1)}_p - \vartheta^{(1)}_q = -2\eta C_\vartheta \neq 0, \tag{14}$$

so the initially identical pair separates after the first update. The opposed gradients themselves are established before any optimizer step; the effect of optimizer-specific update scaling is examined separately in Appendix D.1.

The paired basis therefore preserves the temporal predictor at birth while keeping the newly introduced capacity directly connected to the training objective.

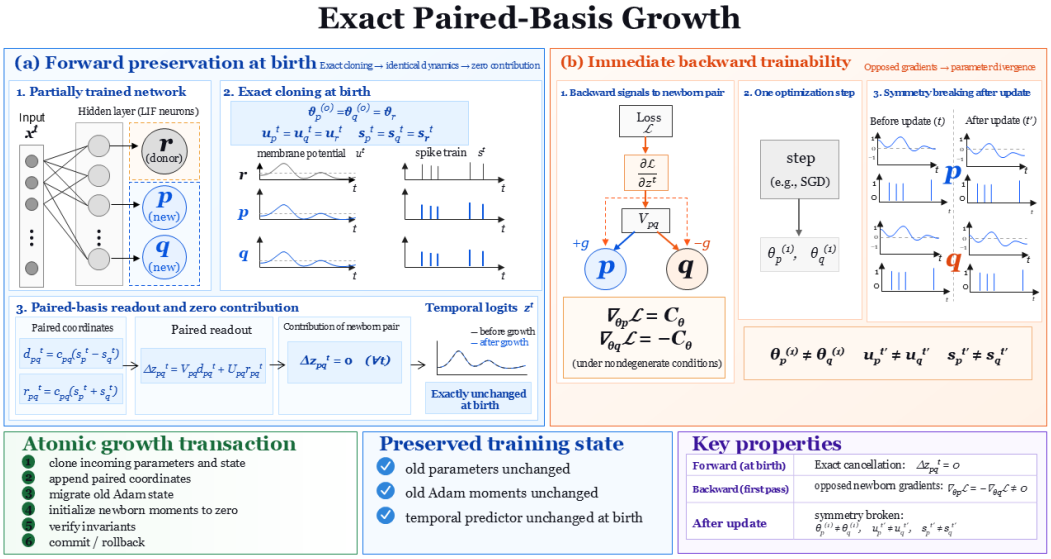

Figure 1: Exact paired-basis growth. At birth, cloned LIF neurons follow identical membrane and spike trajectories, causing their difference coordinate to cancel exactly in the forward pass. The nonzero difference readout transmits equal and opposing learning signals during backpropagation, so the pair begins to diverge after the first optimizer update.

3.3 Theoretical Guarantees and Training Continuity

We summarize the main properties of the construction below. Full proofs are provided in Appendix A.

Theorem 1 (Temporal function preservation)**.** *Consider a growth event that introduces K newborn pairs using the construction in Section 3.2. If all existing parameters and temporal states remain unchanged, then for the same input sequence and temporal initialization,*

$$\mathbf{z}^{t,+} = \mathbf{z}^{t}, \qquad \overline{\mathbf{z}}^{+} = \overline{\mathbf{z}}, \qquad f_{\boldsymbol{\theta}^+}(\mathbf{x}^{1:T}) = f_{\boldsymbol{\theta}}(\mathbf{x}^{1:T}), \qquad \forall t. \tag{15}$$

Every existing neuron also preserves its membrane and spike trajectory,

$$\mathbf{u}_{\text{old}}^{t,+} = \mathbf{u}_{\text{old}}^{t}, \qquad \mathbf{s}_{\text{old}}^{t,+} = \mathbf{s}_{\text{old}}^{t}, \qquad \forall t. \tag{16}$$

Proof sketch. Cloned neurons follow identical temporal trajectories, so every newborn difference coordinate remains zero. Since the sum readout is initialized to zero and the existing computation is unchanged, the temporal logits are preserved. The complete proof is given in Appendix A.

Theorem 2 (Conditional immediate trainability)**.** *Under the conditions of Theorem 1, if $\mathbf{V}_{pq}^{(0)} \neq \mathbf{0}$ and $\mathbf{C}_{\vartheta} \neq \mathbf{0}$, then*

$$\nabla_{\vartheta_p} \mathcal{L} = -\nabla_{\vartheta_q} \mathcal{L} \neq \mathbf{0}. \tag{17}$$

Thus, the newborn pair preserves the network computation at birth while remaining immediately connected to the task loss.

Proof sketch. The cloned neurons have identical surrogate BPTT Jacobians, while the difference readout assigns equal and opposite learning signals. Their parameter gradients are therefore opposed whenever the resulting batch and temporal contraction is nonzero.

Physical expansion also changes the shapes of trainable tensors. For Adam, let $\mathbf{m}$, $\mathbf{v}$, and k denote the first moment, second moment, and optimizer step. The expanded state is

$$\boldsymbol{\theta}^{+} = E(\boldsymbol{\theta}) \oplus \boldsymbol{\theta}_{\text{new}}, \qquad \mathbf{m}^{+} = E(\mathbf{m}) \oplus \mathbf{0}, \qquad \mathbf{v}^{+} = E(\mathbf{v}) \oplus \mathbf{0}, \qquad k^{+} = k. \tag{18}$$

Existing coordinates retain their parameters, Adam moments, and optimization history, while the newly allocated coordinates begin with zero moments.

The construction also extends to intermediate hidden layers. Growth changes the dimension of the spike vector, so the preserved quantity is the temporal synaptic drive received by the next layer. For an expansion at layer ℓ,

$$\Delta \mathbf{a}_{\ell+1}^t = \sum_{k=1}^{K} \left(\mathbf{V}_k d_k^t + \mathbf{U}_k r_k^t \right) = \mathbf{0}, \qquad \forall t. \tag{19}$$

The next layer therefore receives the same temporal input at birth, and the same argument propagates through all downstream layers.

In implementation, each growth event is executed as an atomic transaction. For a deterministic verification set, let $\overline{\mathbf{z}}_n^-$ and $\overline{\mathbf{z}}_n^+$ denote the sequence-level logits of verification sample n immediately before and after growth, respectively. Before the expanded state is accepted, we evaluate

$$\begin{aligned} \epsilon_z &= \max_n \left\| \overline{\mathbf{z}}_n^+ - \overline{\mathbf{z}}_n^- \right\|_\infty, \qquad \epsilon_{\text{old}} = \max_{j \in \mathcal{I}_{\text{old}}} \left| \boldsymbol{\theta}_{E(j)}^+ - \boldsymbol{\theta}_j \right|, \\ \epsilon_{\text{opt}} &= \max_{j \in \mathcal{I}_{\text{old}}} \max \left\{ \left| \mathbf{m}_{E(j)}^+ - \mathbf{m}_j \right|, \left| \mathbf{v}_{E(j)}^+ - \mathbf{v}_j \right| \right\}. \end{aligned} \tag{20}$$

Here, E denotes the embedding from the original parameter coordinates to their corresponding coordinates in the expanded model. The quantity ϵ_z is the implementation-level predictor check, whereas Theorem 1 establishes the stronger preservation of the complete temporal computation in exact arithmetic. The expanded state is accepted only when the required numerical and state-preservation checks are satisfied; otherwise the original model and optimizer state are restored.

4 Experiments

4.1 Experimental Setup

Tasks and protocol. We evaluate the proposed paired-basis growth mechanism on three synthetic classification tasks, DoubleMoon, Spiral, and SpiralHard, together with the UCI Human Activity Recognition (UCI HAR) dataset (Anguita et al., 2013). Among the synthetic tasks, SpiralHard exhibits the clearest capacity-limited regime and is therefore used for both the matched mechanism study and the repeated-growth experiments.

Unless otherwise stated, all experiments use seeds 42, 43, and 44, with identical disjoint training, validation, and test partitions across compared methods within each seed. Synthetic models are directly trained LIF SNNs with direct current input encoding, $T = 10$, $\beta = 0.9$, $V_{\text{th}} = 1$, and subtractive reset. We use Adam with learning rate 10^{-3} and gradient clipping at 5; the common batch size is 128, with dataset-specific deviations reported in Appendix C.

Compared methods. **Fixed-best** denotes an independently trained fixed-width model selected from the width grid using validation performance. **Fixed-matched** is independently trained at the final physical width reached by the corresponding growing model. **Zero readout** clones the same incoming neuron parameters as the proposed method but initializes the newborn outgoing columns to zero. **Paired-basis** uses the exact difference–sum construction described in Section 3.2.

The matched mechanism experiment applies a single $128 \to 256$ intervention to identical trained checkpoints with the same optimizer state, donor neurons, and data order. In the repeated-growth experiments, models begin at width 128 and add 128 neurons whenever growth is triggered. The growing methods use the same controller settings and growth opportunities, while their subsequent training trajectories are allowed to evolve independently. A width-matched uneven-splitting construction is included as an additional matched mechanism control in Appendix D.1.

Selection and metrics. Growth decisions, capacity selection, and checkpoint selection rely only on validation data, and the test set is evaluated after model selection. We report classification accuracy on the synthetic tasks and Macro-F1 on UCI HAR. Physical capacity is characterized by hidden width and trainable parameter count. To measure the amount of capacity instantiated throughout training, we define

$$\text{WidthEpochs} = \sum_{e=1}^{E} D_e, \qquad \text{ParameterEpochs} = \sum_{e=1}^{E} P_e, \tag{21}$$

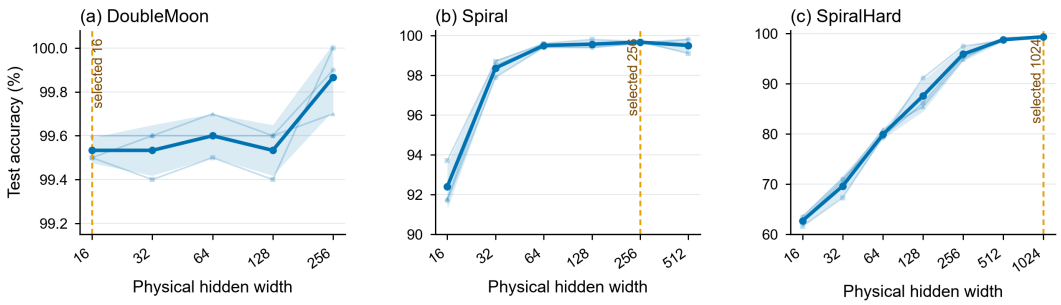

Figure 2: Fixed-width capacity landscapes. Validation-based selection favors widths 16, 256, and 1024 on DoubleMoon, Spiral, and SpiralHard, respectively.

Table 1: Matched $128 \to 256$ growth intervention. G_{new} is the pre-clipping ℓ_1 magnitude of the newborn incoming-weight gradients; triplets follow seeds 42/43/44. N/A denotes an undefined quantity.

Metric	Zero readout	Paired-basis
Δ_{logit}	0	0
G_{new} (42/43/44)	0	2.312/4.309/3.423
E_{opp}	N/A	0
Separation after first update	0	6.302×10^{-3}

where D_e and P_e denote the physical width and parameter count during epoch e. SynOps and spike counts are used as inference computation and activity proxies rather than direct measurements of hardware energy.

4.2 Exactness and Trainability of Growth

Capacity regimes. We first examine whether the benchmark tasks exhibit different capacity requirements. As shown in Figure 2, validation-based selection favors fixed widths 16, 256, and 1024 for DoubleMoon, Spiral, and SpiralHard, respectively. The increasing preferred widths reveal substantially different capacity regimes and make SpiralHard a suitable setting for studying whether useful capacity can be introduced progressively during training.

Matched intervention. We next isolate the immediate effect of the growth construction using the matched $128 \to 256$ intervention. We measure the maximum temporal-logit deviation $\Delta_{\text{logit}} = \max_t \|\mathbf{z}^{t,+} - \mathbf{z}^t\|_\infty$, the pre-clipping newborn incoming-weight gradient magnitude $G_{\text{new}} = \|\nabla_{\mathbf{W}_{\text{new}}} \mathcal{L}\|_1$, and the normalized opposition error

$$E_{\text{opp}} = \frac{\|\mathbf{g}_p + \mathbf{g}_q\|_2}{\|\mathbf{g}_p\|_2 + \|\mathbf{g}_q\|_2 + \epsilon}. \tag{22}$$

Table 1 shows the distinction predicted by the analysis in Section 3.2. Both constructions preserve the temporal logits exactly at the moment of growth. Zero readout, however, provides no immediate task gradient to the incoming parameters of the newborn neurons, which therefore remain identical after the first update. Paired-basis growth preserves the same forward computation while producing nonzero and opposing newborn gradients in every run. The initially identical neurons consequently separate after the first optimizer update.

Appendix D.1 provides two stronger controls. First, a width-matched uneven-splitting construction also produces immediate newborn gradients, but does so by redistributing the effective readout of existing donors and therefore modifies pre-existing parameter coordinates. Second, optimizer controls show that the opposed gradients are already present before any optimizer step and that plain SGD produces nonzero first-step pair separation in all three seeds. Thus, inherited-step Adam affects the magnitude of the first update but is not required for immediate trainability or symmetry breaking.

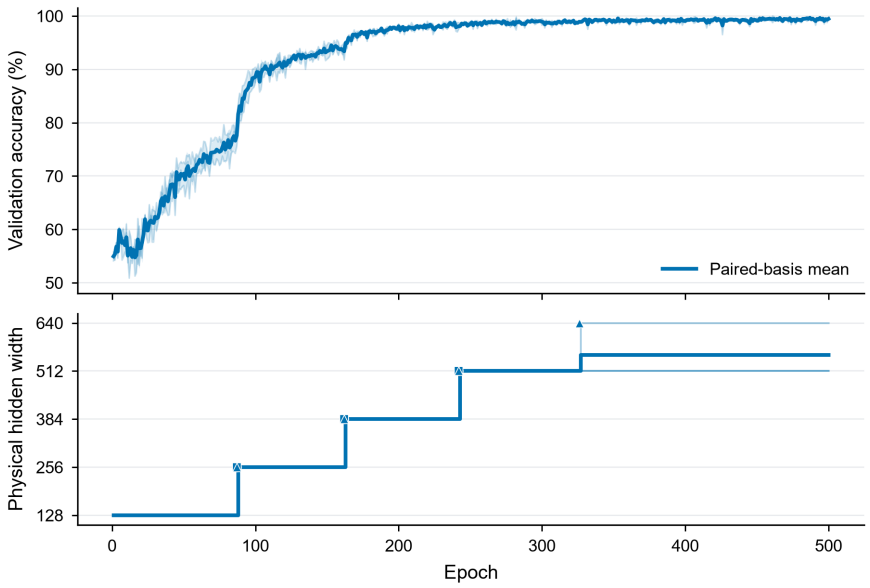

Figure 3: Progressive growth on SpiralHard. Paired-basis models acquire additional physical capacity over training while preserving the temporal computation at each intervention.

Table 2: SpiralHard results averaged over three seeds. W-Epochs and P-Epochs denote cumulative physical width and parameter allocation. Paired-basis contains one additional learned importance scalar relative to Fixed-matched.

Method	Acc.	Width	Params.	W-Epochs	P-Epochs	SynOps	Spikes
Fixed-best	99.400	1024	5122	512.0k	2561.0k	24.953k	2.236k
Fixed-matched	98.717	554.7	2775	277.3k	1387.7k	13.545k	1.226k
Zero readout	98.400	896	4483	230.4k	1153.5k	21.457k	1.768k
Paired-basis	**99.483**	**554.7**	**2776**	**200.6k**	**1004.4k**	**12.170k**	**0.539k**

4.3 Progressive Growth and Final Performance

SpiralHard. We next examine whether the local mechanism remains useful when growth is applied repeatedly during training. All growing models begin at width 128 and add 128 neurons at each triggered event. Figure 3 shows the corresponding validation performance and physical-width trajectories.

Table 2 summarizes the resulting performance and resource usage. Paired-basis growth reaches 99.483% mean test accuracy, compared with 99.400% for the substantially wider Fixed-best reference. At the same time, it reduces width-epochs and parameter-epochs by approximately 60.8%, together with reductions of 51.2% in SynOps and 75.9% in spike count.

The Fixed-matched comparison controls for the final physical capacity. Despite ending at approximately the same mean width, the independently trained fixed models reach 98.717% accuracy, whereas paired-basis growth reaches 99.483%. The zero-readout baseline acquires even more physical capacity but reaches 98.400%. Together with the matched intervention above, this result is consistent with the importance of providing newly introduced neurons with an immediate learning pathway rather than preserving the output through zero outgoing weights alone.

UCI HAR. We finally examine how physical growth behaves when the preferred capacity is less consistent across training and model selection. Figure 4 separates validation-based model selection from a subsequent physical-compaction experiment. The fixed references select widths $256/384/256$ for seeds 42/43/44, whereas the growing models select $128/512/128$. Their mean Macro-F1 increases from 0.92711 ± 0.00190 for the fixed references to 0.92858 ± 0.00048 for paired-basis growth.

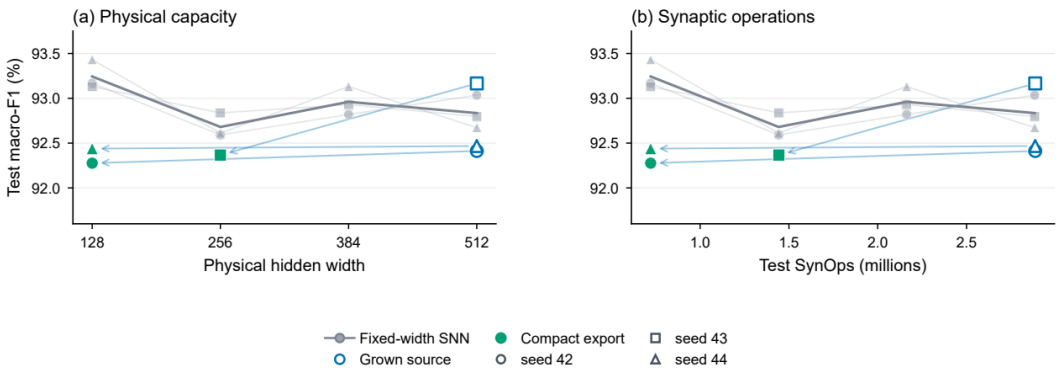

Figure 4: UCI HAR model selection and physical compaction. Validation selects different physical widths along the growing trajectories, while the separate compaction experiment converts the final width-512 source models into smaller physical networks without additional training.

The compaction experiment addresses a separate question, whether the larger physical model reached during training must also be retained for deployment. Starting from the final width-512 growth checkpoints, validation selects compact physical exports at widths $128/256/128$. Their mean Macro-F1 is 0.92360 ± 0.00081, compared with 0.92682 ± 0.00420 for the corresponding width-512 source models, a mean change of -0.00322. No additional training or fine-tuning is performed during this process.

Taken together, these experiments show that paired-basis growth remains exact and trainable across repeated interventions while allowing physical capacity to be introduced progressively. On SpiralHard, this produces a favorable relation between predictive performance and cumulative capacity use. On UCI HAR, the capacity explored during optimization can be separated from the physical capacity ultimately retained for deployment. Additional mechanism details, sensitivity analyses, multilayer experiments, capacity-allocation results, and event-based evaluation are provided in the appendix.

5 Discussion and Conclusion

We studied how a partially trained stateful SNN can acquire additional physical capacity without disrupting the temporal computation it has already learned. The proposed paired-basis construction introduces cancelling neuron pairs that preserve the temporal output exactly at the moment of growth while retaining nonzero and opposing gradients for the newborn parameters. Together with consistent Adam-state migration, this allows structural expansion to be integrated directly into an ongoing optimization trajectory.

The theoretical analysis establishes temporal preservation, conditional immediate trainability, preservation of the optimizer state for existing parameters, and layerwise composition, while the matched intervention experiments confirm the predicted mechanism in practice. On SpiralHard, progressive growth reaches performance comparable to or slightly above the large fixed-capacity reference while using substantially less cumulative physical capacity and lower inference cost proxies. On UCI HAR, validation selects different capacities along the growth trajectories, and subsequent physical compaction produces smaller deployment models with only a modest reduction in predictive performance.

These results support paired-basis growth as a structural primitive for controlled expansion in stateful SNNs. The mechanism can be combined with different policies for deciding when and how much capacity to add.

AI Use Statement

Generative AI tools were used in a limited supporting role during the preparation of this manuscript. Their use included assistance with mathematical expression and derivation checking, LaTeX for-

matting, language polishing, and improving the clarity and organization of selected passages. The research problem formulation, method design, implementation, experiments, data analysis, interpretation of results, and scientific conclusions were carried out and verified by the authors. All AI-assisted content was reviewed and revised by the authors, who take full responsibility for the final manuscript.

REFERENCES

Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra Modha. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7243–7252, 2017.

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L. Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 437–442, 2013.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In *Proceedings of Machine Learning and Systems*, volume 2, pp. 129–146, 2020.

Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. Net2net: Accelerating learning via knowledge transfer. In *International Conference on Learning Representations*, 2016. URL `https://arxiv.org/abs/1511.05641`.

Yanqi Chen, Zhaofei Yu, Wei Fang, Zhengyu Ma, Tiejun Huang, and Yonghong Tian. State transition of dendritic spines improves learning of sparse spiking neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 3701–3715, 2022.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

Federico Errica, Henrik Christiansen, Viktor Zaverkin, Mathias Niepert, and Francesco Alesiani. Adaptive width neural networks. In *International Conference on Learning Representations*, 2026.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2943–2952, 2020.

Scott E. Fahlman and Christian Lebiere. The cascade-correlation learning architecture. In *Advances in Neural Information Processing Systems*, volume 2, 1989.

Wulfram Gerstner, Werner M. Kistler, Richard Naud, and Liam Paninski. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, 2014.

Bing Han, Feifei Zhao, Yi Zeng, and Wenxuan Pan. Adaptive sparse structure development with pruning and regeneration for spiking neural networks. *arXiv preprint arXiv:2211.12219*, 2022a.

Bing Han, Feifei Zhao, Yi Zeng, and Guobin Shen. Developmental plasticity-inspired adaptive pruning for deep spiking and artificial neural networks. *arXiv preprint arXiv:2211.12714*, 2022b.

Bing Han, Feifei Zhao, Yi Zeng, Wenxuan Pan, and Guobin Shen. Enhancing efficient continual learning with dynamic structure development of spiking neural networks. *arXiv preprint arXiv:2308.04749*, 2023.

Yang He and Lingao Xiao. Structured pruning for deep convolutional neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):2900–2919, 2024. doi: 10.1109/TPAMI.2023.3334614.

Shaoyi Huang, Haowen Fang, Kaleel Mahmood, Bowen Lei, Nuo Xu, Bin Lei, Yue Sun, Dongkuan Xu, Wujie Wen, and Caiwen Ding. Neurogenesis dynamics-inspired spiking neural network training acceleration. *arXiv preprint arXiv:2304.12214*, 2023.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eYHoC5FX.

Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. doi: 10.1016/S0893-6080(97)00011-7.

Byunggook Na, Jisoo Mok, Seongsik Park, Dongjin Lee, Hyeokjun Choe, and Sungroh Yoon. AutoSNN: Towards energy-efficient spiking neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16253–16269, 2022.

Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. doi: 10.1109/MSP.2019.2931595.

Rachmad Vidya Wicaksana Putra and Muhammad Shafique. SpikeNAS: A fast memory-aware neural architecture search framework for spiking neural network-based embedded ai systems. *IEEE Transactions on Artificial Intelligence*, 2025. doi: 10.1109/TAI.2025.3586238.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4780–4789, 2019. doi: 10.1609/aaai.v33i01.33014780.

Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575:607–617, 2019. doi: 10.1038/s41586-019-1677-2.

Jinling Wang, Ammar Belatreche, Liam P. Maguire, and T. Martin McGinnity. An online supervised learning method for spiking neural networks with adaptive structure. *Neurocomputing*, 144: 526–536, 2014. doi: 10.1016/j.neucom.2014.04.017.

Tao Wei, Changhu Wang, Yong Rui, and Chang Wen Chen. Network morphism. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 564–572, 2016. URL https://proceedings.mlr.press/v48/wei16.html.

Colin White, Mahmoud Safari, Rhea Sukthanker, Binxin Ru, Thomas Elsken, Arber Zela, Debadeepta Dey, and Frank Hutter. Neural architecture search: Insights from 1000 papers. *arXiv preprint arXiv:2301.08727*, 2023.

Lemeng Wu, Dilin Wang, and Qiang Liu. Splitting steepest descent for growing neural architectures. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Lemeng Wu, Bo Liu, Peter Stone, and Qiang Liu. Firefly neural architecture descent: A general approach for growing neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22373–22383, 2020.

Yiqun Yao, Zheng Zhang, Jing Li, and Yequan Wang. Masked structural growth for 2x faster language model pre-training. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rL7xsg1aRn.

Qifan Yu, Xinyu Ma, Zhijian Zhuo, Minrui Wang, Deyi Liu, Shiyi Zhan, Yiyuan Ma, Liang Xiang, Xingyan Bin, and Di He. Sparkling: Balancing signal preservation and symmetry breaking for width-progressive learning. *arXiv preprint arXiv:2602.02472*, 2026.

Xin Yuan, Pedro Savarese, and Michael Maire. Accelerated training via incrementally growing neural networks using variance transfer and learning rate adaptation. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2306.12700.

Friedemann Zenke and Surya Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural Computation*, 30(6):1514–1541, 2018. doi: 10.1162/neco_a_01086.

Friedemann Zenke and Tim P. Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, 33(4):899–925, 2021. doi: 10.1162/neco_a_01367.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=r1Ue8Hcxg.

A Additional Theoretical Results

A.1 Proof of Temporal Function Preservation

We prove Theorem 1. Consider a growth event that introduces K newborn pairs using the construction in Section 3.2. The input sequence, the parameters of all existing neurons, and their temporal states are unchanged when the expanded network is created.

For one newborn pair (p, q) cloned from the same donor neuron, the incoming parameters and initial membrane states satisfy

$$\mathbf{w}_p^{(0)} = \mathbf{w}_q^{(0)}, \qquad b_p^{(0)} = b_q^{(0)}, \qquad u_p^0 = u_q^0. \tag{A.1}$$

The two neurons also use identical membrane decay, threshold, spike, and reset dynamics, so they receive the same input current at every time step.

Their temporal trajectories remain identical by induction. Suppose $u_p^{t-1} = u_q^{t-1}$. Since $i_p^t = i_q^t$,

$$\widetilde{u}_p^t - \widetilde{u}_q^t = \beta\left(u_p^{t-1} - u_q^{t-1}\right) + \left(i_p^t - i_q^t\right) = 0. \tag{A.2}$$

Applying the same threshold and reset operations therefore gives $u_p^t = u_q^t$ and $s_p^t = s_q^t$. Together with the identical initial states, this proves

$$u_p^t = u_q^t, \qquad s_p^t = s_q^t, \qquad \forall t \in \{1, \dots, T\}.$$

For the newborn pair, the difference and sum coordinates are $d_{pq}^t = c_{pq}(s_p^t - s_q^t)$ and $r_{pq}^t = c_{pq}(s_p^t + s_q^t)$. At birth, $\mathbf{V}_{pq}^{(0)} \neq \mathbf{0}$ and $\mathbf{U}_{pq}^{(0)} = \mathbf{0}$. Since the two spike trajectories are identical, $d_{pq}^{t,(0)} = 0$, and the pair contributes

$$\Delta\mathbf{z}_{pq}^{t,(0)} = \mathbf{V}_{pq}^{(0)} d_{pq}^{t,(0)} + \mathbf{U}_{pq}^{(0)} r_{pq}^{t,(0)} = \mathbf{0}, \qquad \forall t. \tag{A.3}$$

The same argument applies independently to all K newborn pairs, so their total contribution to the temporal logits is zero at every simulation step.

The existing neurons retain their parameters and temporal states. Applying the same deterministic dynamics to the same input sequence therefore preserves their membrane and spike trajectories. Combining their unchanged contribution with the zero contribution of the newborn pairs gives

$$\mathbf{z}^{t,+} = \mathbf{W}_{\text{out}}\mathbf{s}_{\text{old}}^{t,+} + \mathbf{b}_{\text{out}} + \Delta\mathbf{z}_{\text{new}}^t = \mathbf{W}_{\text{out}}\mathbf{s}^t + \mathbf{b}_{\text{out}} = \mathbf{z}^t, \qquad \forall t. \tag{A.4}$$

Thus the complete temporal logit sequence is unchanged. In particular, the temporal average satisfies $\overline{\mathbf{z}}^+ = \overline{\mathbf{z}}$, and hence

$$f_{\boldsymbol{\theta}^+}(\mathbf{x}^{1:T}) = f_{\boldsymbol{\theta}}(\mathbf{x}^{1:T}).$$

This proves Theorem 1.

A.2 Proof of Conditional Immediate Trainability

We prove Theorem 2. Consider one newborn pair (p, q) at the moment of growth. From Appendix A.1, the two neurons have identical membrane and spike trajectories. Their surrogate BPTT Jacobians with respect to corresponding incoming parameter blocks are therefore also identical.

Let $\boldsymbol{\delta}^t = \partial\mathcal{L}/\partial\mathbf{z}^{t,+}$. For the paired basis in Section 3.2, the spike gradients are

$$\begin{aligned} \mathbf{g}_p^t &= c_{pq}\left(\mathbf{V}_{pq} + \mathbf{U}_{pq}\right)^\top \boldsymbol{\delta}^t, \\ \mathbf{g}_q^t &= c_{pq}\left(-\mathbf{V}_{pq} + \mathbf{U}_{pq}\right)^\top \boldsymbol{\delta}^t. \end{aligned} \tag{A.5}$$

Since $\mathbf{U}_{pq}^{(0)} = \mathbf{0}$ at birth, these reduce to

$$\mathbf{g}_p^{t,(0)} = -\mathbf{g}_q^{t,(0)} = c_{pq}\left(\mathbf{V}_{pq}^{(0)}\right)^\top \boldsymbol{\delta}^t. \tag{A.6}$$

Thus exact forward cancellation does not eliminate the learning signal received by the newborn neurons.

Let $\boldsymbol{\vartheta}_p$ and $\boldsymbol{\vartheta}_q$ denote corresponding incoming parameter blocks. Because the two neurons follow the same temporal trajectory, their BPTT Jacobians satisfy $\mathbf{J}_p^t = \mathbf{J}_q^t = \mathbf{J}_\vartheta^t$ at birth. Define

$$\mathbf{C}_\vartheta = \sum_{t=1}^{T} \left(\mathbf{J}_\vartheta^t\right)^\top c_{pq} \left(\mathbf{V}_{pq}^{(0)}\right)^\top \boldsymbol{\delta}^t. \tag{A.7}$$

The corresponding incoming parameter gradients are therefore

$$\nabla_{\boldsymbol{\vartheta}_p} \mathcal{L} = \mathbf{C}_\vartheta, \qquad \nabla_{\boldsymbol{\vartheta}_q} \mathcal{L} = -\mathbf{C}_\vartheta. \tag{A.8}$$

The condition $\mathbf{V}_{pq}^{(0)} \neq \mathbf{0}$ creates a backward path to the newborn parameters, but a nonzero readout alone does not guarantee a nonzero parameter gradient. Immediate trainability requires $\mathbf{C}_\vartheta \neq \mathbf{0}$. This excludes cases in which the task error is zero, the surrogate derivative is inactive, or the contributions over time and samples cancel exactly.

Under this condition, the two newborn parameter blocks receive nonzero gradients with opposite signs. Under a plain gradient-descent step with learning rate η,

$$\boldsymbol{\vartheta}_p^{(1)} - \boldsymbol{\vartheta}_q^{(1)} = -2\eta \mathbf{C}_\vartheta \neq \mathbf{0}. \tag{A.9}$$

The pair therefore leaves its initially symmetric parameter state after the first update and can develop distinct temporal responses.

The difference and sum readout coordinates behave differently at birth. Since $d_{pq}^{t,(0)} = 0$, the gradient of the difference readout is zero, $\nabla_{\mathbf{V}_{pq}} \mathcal{L} = \mathbf{0}$. In contrast, $r_{pq}^{t,(0)} = 2c_{pq} s_p^{t,(0)}$, so

$$\nabla_{\mathbf{U}_{pq}} \mathcal{L}\big|_{\text{birth}} = 2c_{pq} \sum_{t=1}^{T} \boldsymbol{\delta}^t \left(s_p^{t,(0)}\right)^\top, \tag{A.10}$$

which may be nonzero. The difference coordinate therefore provides the opposing signal that separates the newborn hidden parameters, while the sum coordinate can begin adapting the shared readout direction.

This proves conditional immediate trainability.

A.3 Preservation of Optimizer State and Layerwise Composition

We establish two additional properties needed to continue training after physical expansion: preservation of the Adam state associated with existing parameters and composition of the temporal preservation property across hidden layers.

Let E denote the coordinate injection that maps each existing parameter to its corresponding location in the expanded tensors. For Adam, let $\mathbf{m}$, $\mathbf{v}$, and k denote the first moment, second moment, and optimizer step. After growth,

$$\boldsymbol{\theta}^+ = E(\boldsymbol{\theta}) \oplus \boldsymbol{\theta}_{\text{new}}, \qquad \mathbf{m}^+ = E(\mathbf{m}) \oplus \mathbf{0}, \qquad \mathbf{v}^+ = E(\mathbf{v}) \oplus \mathbf{0}, \qquad k^+ = k. \tag{A.11}$$

Proposition 1 (Preservation of optimizer state for existing parameters)**.** *Under Equation A.11, every existing parameter retains its value, Adam moments, and optimizer step. Newly introduced coordinates are initialized with zero first and second moments.*

Proof. For every existing coordinate j, the definition of E gives $\boldsymbol{\theta}_{E(j)}^+ = \boldsymbol{\theta}_j$, $\mathbf{m}_{E(j)}^+ = \mathbf{m}_j$, and $\mathbf{v}_{E(j)}^+ = \mathbf{v}_j$. Since $k^+ = k$, the Adam bias correction continues from the same optimization step. The appended coordinates receive zero moments by construction. □

We next consider growth at an intermediate hidden layer ℓ. The expanded spike vector contains additional coordinates, so the complete spike vector cannot be directly equated with its earlier counterpart. Instead, preservation is expressed through the temporal synaptic drive received by layer $\ell + 1$.

Let $\mathbf{a}_{\ell+1}^t$ denote this synaptic drive. The contribution introduced by the K newborn pairs is

$$\Delta\mathbf{a}_{\ell+1}^t = \sum_{k=1}^{K} \left(\mathbf{V}_{\ell,k} d_{\ell,k}^t + \mathbf{U}_{\ell,k} r_{\ell,k}^t \right). \tag{A.12}$$

At birth, every newborn pair satisfies $d_{\ell,k}^{t,(0)} = 0$ and $\mathbf{U}_{\ell,k}^{(0)} = \mathbf{0}$, so $\Delta\mathbf{a}_{\ell+1}^{t,(0)} = \mathbf{0}$ for every t. The existing neurons also preserve their temporal trajectories. Therefore,

$$\mathbf{a}_{\ell+1}^{t,+} = \mathbf{W}_{\ell+1}\mathbf{s}_\ell^t + \mathbf{b}_{\ell+1} = \mathbf{a}_{\ell+1}^t, \qquad \forall t. \tag{A.13}$$

Corollary 1 (Layerwise temporal composition)**.** *If growth at layer ℓ preserves the temporal synaptic drive received by layer $\ell+1$, and the parameters and initial states of all downstream layers remain unchanged, then their membrane states, spike trajectories, and temporal logits are preserved.*

Proof. Layer $\ell+1$ receives the same input sequence over time and starts from the same initial state. Its deterministic dynamics therefore produce the same membrane and spike trajectories. Applying the same argument successively to each downstream layer preserves the temporal logits of the network. □

The same reasoning extends to repeated growth. Consider a finite sequence of valid growth events $\mathcal{G}_1, \ldots, \mathcal{G}_M$, and let $f^{(m)}$ denote the temporal predictor after the mth event. If each event preserves the temporal interface presented to the unchanged part of the network, then $f^{(m)}(\mathbf{x}^{1:T}) = f^{(m-1)}(\mathbf{x}^{1:T})$ at every growth instant. Repeated application therefore gives

$$f^{(M)}(\mathbf{x}^{1:T}) = f^{(0)}(\mathbf{x}^{1:T}). \tag{A.14}$$

Thus the preservation property composes across both hidden layers and successive growth events.

B Atomic Growth Transaction

Physical growth is implemented as a single atomic transaction over the model and optimizer. Each growth event constructs the newborn paired basis, expands the parameter tensors, transfers the Adam state, verifies the expanded model, and resumes training only after all required conditions are satisfied.

For an expansion of $2K$ neurons, the existing neurons are ranked deterministically according to the ℓ_2 norm of their effective downstream readout. We use a stable descending sort, with the lower physical neuron index used to resolve exact ties. The first K neurons in this ordering are selected as donors.

For donor r_k, the newborn neurons (p_k, q_k) receive identical copies of its incoming parameters,

$$\mathbf{w}_{p_k}^{(0)} = \mathbf{w}_{q_k}^{(0)} = \mathbf{w}_{r_k}, \qquad b_{p_k}^{(0)} = b_{q_k}^{(0)} = b_{r_k}. \tag{B.1}$$

The implementation associates each physical neuron with a positive importance coefficient. Let M denote the reference maximum width and let $\xi_j = (j-1)/(M-1)$ be the normalized physical position of neuron j. A learned raw parameter ρ determines the positive importance rate

$$\lambda_{\text{imp}} = \text{softplus}(\rho) + 10^{-8}, \tag{B.2}$$

and the coefficient assigned to neuron j is

$$a_j = \frac{\exp(-\lambda_{\text{imp}}\xi_j)}{\frac{1}{M}\sum_{\ell=1}^{M}\exp(-\lambda_{\text{imp}}\xi_\ell)}. \tag{B.3}$$

Hence $a_j > 0$ for every valid physical position. For a newborn pair (p_k, q_k), its paired scale is

$$c_k = \sqrt{a_{p_k} a_{q_k}} > 0. \tag{B.4}$$

For a standard neuron j with stored readout column $\mathbf{w}_j$, we define its effective downstream vector as

$$\mathbf{e}_j = a_j \mathbf{w}_j.$$

For an existing paired segment (p, q), the corresponding effective vectors are

$$\mathbf{e}_p = c_{pq}(\mathbf{V}_{pq} + \mathbf{U}_{pq}), \qquad \mathbf{e}_q = c_{pq}(-\mathbf{V}_{pq} + \mathbf{U}_{pq}).$$

Donors are ranked according to $\|\mathbf{e}_j\|_2$.

Let $\mathbf{e}_{r_k}$ denote the effective downstream output vector of donor r_k. The difference and sum columns are initialized as

$$\mathbf{V}_k^{(0)} = \frac{\gamma}{c_k}\mathbf{e}_{r_k}, \qquad \mathbf{U}_k^{(0)} = \mathbf{0}, \qquad \gamma = \frac{1}{2}. \tag{B.5}$$

This initialization is used throughout the reported experiments.

The Adam state is expanded together with the model parameters. Let E denote the coordinate injection that places each existing entry into its corresponding location in the enlarged tensors. If $\boldsymbol{\theta}$ denotes the model parameters and $\mathbf{m}$, $\mathbf{v}$, and k denote the Adam first moment, second moment, and optimizer step, respectively, then

$$\boldsymbol{\theta}^+ = E(\boldsymbol{\theta}) \oplus \boldsymbol{\theta}_{\text{new}}, \qquad \mathbf{m}^+ = E(\mathbf{m}) \oplus \mathbf{0}, \qquad \mathbf{v}^+ = E(\mathbf{v}) \oplus \mathbf{0}, \qquad k^+ = k. \tag{B.6}$$

The existing coordinates therefore retain their parameter values, Adam moments, and bias correction history, while the newly allocated coordinates begin with zero first and second moments.

Before expansion, the complete model and optimizer state are stored. The provisional expanded model is then evaluated on the same deterministic verification input used for the original model.

For the deterministic verification set, let $\overline{\mathbf{z}}_n^-$ and $\overline{\mathbf{z}}_n^+$ denote the sequence-level logits of sample n immediately before and after growth. The predictor discrepancy is

$$\epsilon_z = \max_n \left\| \overline{\mathbf{z}}_n^+ - \overline{\mathbf{z}}_n^- \right\|_\infty . \tag{B.7}$$

Preservation of the existing parameters and Adam state is measured by

$$\epsilon_{\text{old}} = \max_{j \in \mathcal{I}_{\text{old}}} \left| \boldsymbol{\theta}_{E(j)}^+ - \boldsymbol{\theta}_j \right|, \tag{B.8}$$

$$\epsilon_{\text{opt}} = \max_{j \in \mathcal{I}_{\text{old}}} \left\{ \left| \mathbf{m}_{E(j)}^+ - \mathbf{m}_j \right|, \left| \mathbf{v}_{E(j)}^+ - \mathbf{v}_j \right| \right\}. \tag{B.9}$$

For the newly allocated optimizer coordinates, we additionally compute

$$\epsilon_{\text{new}} = \max_{j \in \mathcal{I}_{\text{new}}} \left\{ |\mathbf{m}_j^+|, |\mathbf{v}_j^+| \right\}. \tag{B.10}$$

The expanded state is accepted only if

$$\epsilon_z \le 10^{-6}, \qquad \epsilon_{\text{old}} = 0, \qquad \epsilon_{\text{opt}} = 0, \qquad \epsilon_{\text{new}} = 0, \tag{B.11}$$

the verification predictions remain unchanged, the newborn incoming parameters match their donors exactly, and the difference and sum columns satisfy the prescribed initialization. Exact equality is required for preservation of the model parameters and Adam state, whereas the numerical tolerance is used only for the verification logits. Algorithm 1 summarizes the complete growth transaction.

Let $\mathcal{S}^-$ denote the complete training state before growth, and let $\widetilde{\mathcal{S}}^+$ denote the provisional expanded state. The transaction can be summarized as

$$\mathcal{S}^+ = \begin{cases} \widetilde{\mathcal{S}}^+, & \text{if all required checks are satisfied,} \\ \mathcal{S}^-, & \text{otherwise.} \end{cases} \tag{B.12}$$

Training therefore continues either from a fully verified expanded state or from the unchanged original state, never from a partially constructed or partially migrated model.

C Datasets and Experimental Environment

Datasets. We evaluate paired-basis growth on three synthetic classification tasks, **DoubleMoon**, **Spiral**, and **SpiralHard**, together with two additional benchmark datasets. The synthetic tasks are two-dimensional classification problems with different capacity requirements and are used to study fixed-width behavior, controlled growth interventions, and capacity allocation. Among them, SpiralHard exhibits the clearest limitation under insufficient width and is therefore used as the primary synthetic benchmark in the main experiments.

Algorithm 1 Atomic paired-basis growth transaction

Require: Model parameters $\boldsymbol{\theta}$, optimizer state $\mathcal{O}$, current width D, number of newborn pairs K
Ensure: Verified expanded state or restored original state
1: Store the current model and optimizer state
2: Evaluate the reference verification logits
3: Rank existing neurons and select K donors
4: **for** $k = 1, \ldots, K$ **do**
5: Clone the incoming parameters of donor r_k
6: Construct newborn pair (p_k, q_k)
7: Initialize $\mathbf{V}_k \leftarrow \gamma \mathbf{e}_{r_k}/c_k$ and $\mathbf{U}_k \leftarrow \mathbf{0}$
8: **end for**
9: Expand the parameter tensors
10: Transfer the existing Adam state and append zero moments
11: Evaluate the verification logits of the expanded model
12: Check the model, optimizer, and functional invariants
13: **if** all required checks are satisfied **then**
14: Commit the expanded model and optimizer
15: **else**
16: Restore the stored model and optimizer state
17: **end if**

We further evaluate the method on the UCI Human Activity Recognition dataset (UCI HAR) (Anguita et al., 2013). UCI HAR contains 561 input features and six activity classes. We use the feature representation provided by the dataset and construct disjoint training, validation, and test partitions for model selection and final evaluation.

DVS Gesture (Amir et al., 2017) provides an additional event-based benchmark with 11 gesture classes. We retain the official train/test separation and construct a subject-disjoint validation split from the training subjects. A deterministic, non-learned preprocessing stage divides each event sequence into 10 temporal bins and aggregates events over a 16×16 spatial grid with two polarities, producing 512 input features per SNN time step. Across these datasets, the growth operations are evaluated under the same preservation requirements, while the predictive effect after growth is allowed to vary across tasks and seeds.

Environment and common training protocol. The core experiments were conducted with Python 3.11.15, PyTorch 2.4.0, and CUDA 12.1 on NVIDIA GeForce RTX 3090 GPUs. Two GPUs were used to run independent experiment queues. Within each matched comparison, the seed, data split, model configuration, and device assignment were kept unchanged.

Unless otherwise specified, experiments use seeds 42, 43, and 44, direct-current input encoding, and leaky integrate-and-fire neurons with $T = 10$ simulation steps, membrane decay $\beta = 0.9$, threshold $V_{\mathrm{th}} = 1$, subtractive reset, and a fast-sigmoid surrogate gradient with slope 10. Optimization uses Adam with learning rate 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, with no learning-rate scheduler or weight decay. Global gradient clipping is applied at norm 5. Synthetic and UCI HAR experiments use batch size 128, whereas DVS Gesture uses batch size 64.

Synthetic experiments use 500 training epochs unless otherwise specified. The DoubleMoon fixed-width sweep uses 200 epochs, and both UCI HAR and DVS Gesture use 200 epochs. The fixed-width capacity sweeps cover widths 16–256 for DoubleMoon, 16–512 for Spiral, and 16–1024 for SpiralHard. For UCI HAR and DVS Gesture, independently trained fixed-width references use widths $\{128, 256, 384, 512\}$, while growing models start from width 128 and expand in increments of 128 up to a maximum width of 512.

For each seed, all compared methods use the same disjoint training, validation, and test partitions. Normalization statistics are computed from the training partition only. Growth decisions, width selection, and checkpoint selection use validation data only, while the test partition is evaluated only after model selection. Experiment-specific growth schedules, allocation policies, and configurations that differ from this common protocol are described with the corresponding experiments in the following appendices.

Matched control protocol. The matched mechanism study in Appendix D.1 starts from the seed-specific epoch-87, width-128 SpiralHard checkpoints and applies one $128 \to 256$ growth event. Each intervention selects 64 distinct donors and creates two newborn clones per donor. All compared arms use the same checkpoint, optimizer state, donor ranking, verification inputs, first post-growth minibatch, and subsequent data order.

The ten-epoch continuation covers epochs 88–97. Its early validation-loss AUC is computed as the unnormalized trapezoidal integral over the ten validation losses in this interval,

$$\mathrm{AUC}_{88:97} = \sum_{e=88}^{96} \frac{\mathcal{L}_{\mathrm{val}}^{e} + \mathcal{L}_{\mathrm{val}}^{e+1}}{2}, \tag{C.1}$$

corresponding to nine unit-width intervals. The epoch-87 validation loss is not included.

The uneven-splitting control preserves the same 128-neuron increment by redistributing each donor's effective downstream contribution across the donor and its two newborn clones in the ratio $\frac{2}{3} : \frac{1}{6} : \frac{1}{6}$. Stored readout coordinates are obtained after accounting for the corresponding importance factors. Because this construction modifies the donor readout, preservation of existing parameters and preservation of existing optimizer state are reported separately. Its sequence-level logit deviations are reported directly in Appendix D.1, including the seed-42 value of 1.4305×10^{-6}.

The optimizer comparison starts from identical paired-basis post-growth parameters and identical pre-update gradients. Inherited Adam continues from step 4959 and performs the first post-growth update at step 4960. Fresh-step Adam resets the newborn first and second moments and uses step 1 for the newborn coordinates, while leaving the update of pre-existing coordinates unchanged. Plain SGD uses learning rate 10^{-3} with zero momentum and zero weight decay. Newborn gradients are measured before global clipping. The quantity G_{new} reported in Appendix D.1 is the ℓ_1 magnitude of the newborn incoming-weight gradients.

D Additional Controlled Experiments

D.1 Matched Growth Mechanism Comparison

We conduct a matched single-growth experiment on SpiralHard to isolate the effect of the newborn readout construction. For each seed, the zero-readout and paired-basis models follow identical training through epoch 87 and then undergo the same forced $128 \to 256$ expansion. Within each seed, the two methods start from the same checkpoint and optimizer state, use the same donor neurons and cloned incoming parameters, and follow the same data order and training schedule after growth. The only difference in this primary comparison is the readout assigned to the newborn neurons. The **zero-readout** control appends the same cloned neurons with zero outgoing columns, whereas **paired-basis** uses the difference and sum construction in Section 3.2.

Both constructions preserve the logits at the growth instant, but their backward behavior differs. Zero readout gives zero task gradient to the incoming parameters of the newborn neurons and therefore produces no pair separation after the first optimizer update. Paired-basis instead produces nonzero newborn gradients with zero opposition error at the recorded precision, and the paired neurons separate immediately after the first update. The newborn neurons are active at birth in both methods, so the zero gradient observed for zero readout is not caused by inactive spiking.

To characterize short-horizon adaptation after growth, we use the early validation-loss AUC defined in Equation C.1. Paired-basis gives lower early validation-loss AUC than zero readout in all three matched runs, with paired-minus-control differences of -0.453282, -0.515633, and -0.580897 for seeds 42, 43, and 44, respectively. Table D.1 summarizes the seed-level matched comparison.

Final test accuracy varies across seeds. The paired-minus-control differences are -0.15, $+1.80$, and $+0.70$ percentage points, respectively, with a mean difference of $+0.783$ percentage points. These results are not used to claim a universal improvement in final accuracy. The primary matched comparison instead isolates the mechanism: both constructions preserve the predictor at growth, while only paired-basis provides an immediate opposing learning signal to the newborn incoming parameters.

Table D.1: Matched zero-readout and paired-basis comparison on SpiralHard. G_{new} is the pre-clipping ℓ_1 newborn incoming-weight gradient magnitude. Early-loss AUC covers epochs 88-97; separation is measured after the first update.

Seed	Method	Max logit Δ	G_{new}	Opposition error	First separation	Early loss AUC	Test acc. (%)
42	Zero readout	0	0	N/A	0	3.9513	93.60
42	Paired-basis	0	2.3122	0	0.00630236	3.4980	93.45
43	Zero readout	0	0	N/A	0	3.8492	93.75
43	Paired-basis	0	4.3092	0	0.00630239	3.3336	95.55
44	Zero readout	0	0	N/A	0	3.6714	93.95
44	Paired-basis	0	3.4233	0	0.00630239	3.0905	94.65

Comparison with uneven splitting. We further compare paired-basis growth with a stronger function-preserving symmetry-breaking construction based on uneven splitting. The matched intervention introduces 128 newborn neurons from 64 donors, with two newborn clones per donor. To preserve the same donor set and physical width, the effective downstream contribution of each selected donor is redistributed across the donor and its two clones in the ratio $2/3 : 1/6 : 1/6$. This is a width-matched three-copy extension of uneven splitting rather than the literal one-donor/one-clone $2/3 : 1/3$ construction.

The redistributed effective readouts preserve the intended donor contribution in exact arithmetic and give nonzero newborn incoming gradients in all three seeds. Unlike paired-basis growth, however, this construction modifies the pre-existing donor readout coordinate. The corresponding Adam steps and moments are left numerically unchanged at growth, although the associated donor parameter itself has been altered.

In float32, the sequence-level logit deviations for uneven splitting are 1.4305×10^{-6}, 9.5367×10^{-7}, and 9.5367×10^{-7} for seeds 42, 43, and 44, respectively. Thus two runs remain within the same 10^{-6} numerical tolerance used for the growth checks. Paired-basis, by comparison, retains bitwise-identical probe logits and unchanged old parameter coordinates in all three runs. Table D.2 summarizes the corresponding structural properties.

Table D.2: Structural controls for the matched $128 \to 256$ intervention. Triplets follow seeds 42/43/44. $\Delta\theta_{\text{old}}$ and ΔO_{old} denote changes to existing parameters and optimizer state.

Method	Max logit Δ	G_{new}	$\Delta\theta_{\text{old}}$	ΔO_{old}
Zero readout	$0/0/0$	$0/0/0$	$0/0/0$	$0/0/0$
Uneven split	$1.431\times10^{-6}/9.537\times10^{-7}/9.537\times10^{-7}$	$0.771/1.436/1.141$	$1.006/1.097/1.056$	$0/0/0$
Paired-basis	$0/0/0$	$2.312/4.309/3.423$	$0/0/0$	$0/0/0$

For reference, the ten-epoch continuation gives validation-loss AUCs of 3.823971 ± 0.141636 for zero readout, 3.307367 ± 0.205006 for paired-basis with the inherited Adam clock, and 3.909617 ± 0.096647 for uneven splitting (mean $\pm$ sample standard deviation over three seeds). These short-horizon values describe the complete initialization schemes and are not interpreted as an isolated causal comparison of coordinate preservation or as evidence of universal final-accuracy superiority.

Optimizer control. The magnitude of the first paired-basis update can depend on the Adam bias-correction clock even though the opposed gradients are determined before the optimizer step. To separate these effects, we repeat the first update from the same paired-basis birth state using three conditions: the original inherited-step Adam, a fresh-step Adam control for the newborn coordinates, and momentum-free, decay-free SGD. All conditions use the same minibatch, loss, raw and clipped gradients, and base learning rate of 10^{-3}.

The newborn gradients are identical before the optimizer update and have zero measured opposition error in all three seeds. Plain SGD also produces nonzero first-step pair separation in every run, demonstrating that inherited-step Adam is not required for immediate symmetry breaking. In the fresh-step Adam control, only the newborn bias-correction clock is changed; the first-step parameters and

Adam state of the old coordinates remain identical to those under inherited Adam. The corresponding first-update statistics are summarized in Table D.3.

Table D.3: Optimizer controls for paired-basis birth. Values are mean ± sample standard deviation over three seeds; incoming update norm is the ℓ_2 norm over newborn incoming weights and biases.

Optimizer	G_{new}	First separation	Incoming update norm	$\text{AUC}_{88:97}$
Inherited Adam	3.348264 ± 1.0006	$0.006302377 \pm 1.72\times10^{-8}$	$0.06170064 \pm 7.20\times10^{-5}$	3.307367 ± 0.205006
Fresh-step Adam	3.348264 ± 1.0006	0.002000093 ± 0	$0.01959469 \pm 1.77\times10^{-6}$	3.601274 ± 0.151755
SGD	3.348264 ± 1.0006	$0.000165651 \pm 4.48\times10^{-5}$	$0.000375853 \pm 1.20\times10^{-4}$	–

For inherited Adam, the optimizer clock is 4959 immediately before growth and 4960 on the first update. With $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, the epsilon-free inherited-to-fresh first-update ratio at this clock is 3.1511973. The measured ratio of newborn incoming-update norms is 3.148844 ± 0.003392 across seeds, consistent with the corresponding epsilon-aware prediction up to floating-point update effects. The inherited Adam clock therefore changes the scale of the first update, but it does not generate the opposed newborn gradients and is not necessary for immediate pair separation.

Together, these controls clarify the distinction between the proposed growth operation and alternative symmetry-breaking mechanisms. Uneven splitting can provide immediate newborn gradients by redistributing an existing donor contribution, but this redistribution rewrites a pre-existing parameter coordinate. Paired-basis growth instead introduces trainable newborn coordinates while preserving the existing parameter and Adam coordinates. The optimizer comparison further confirms that the opposed-gradient structure is present before optimization and remains sufficient to break pair symmetry under a plain SGD update.

D.2 Sensitivity to Initial Physical Width

We examine the sensitivity of repeated paired-basis growth to the initial physical width on SpiralHard. Initial widths of 64, 128, and 256 are evaluated with seeds 42, 43, and 44, with width 128 serving as the reference configuration used in the main experiments. All settings use the same data partitions, a 500-epoch training horizon, a growth increment of 128 neurons, checkpoint selection based on validation performance, and the same paired-basis growth mechanism.

All nine runs reach a validation accuracy of at least 99.5%. Across all executed growth events, the predictor, existing parameters, and Adam state satisfy the exact preservation checks. The newborn parameters also receive immediate gradients with the expected opposing structure and separate after their first update. These observations show that changing the initial width does not alter the validity of an individual paired-basis growth event.

The resulting capacity trajectories, however, depend on the initial width. Table D.4 reports the seed-level results. For all nine runs, the checkpoint selected by validation performance occurs at the final realized width, so the selected and final widths are reported in a single column.

Table D.4: Sensitivity to initial physical width on SpiralHard over seeds 42, 43, and 44.

Initial width	Seed	Selected/final width	Test acc. (%)	Growth events
64	42	576	99.05	4
64	43	448	99.15	3
64	44	576	99.05	4
128	42	512	99.75	3
128	43	512	99.25	3
128	44	640	99.45	4
256	42	768	99.70	4
256	43	768	99.05	4
256	44	640	99.45	3

Across seeds, initial widths 64, 128, and 256 achieve test accuracies of $99.083\% \pm 0.058\%$, $99.483\% \pm 0.252\%$, and $99.400\% \pm 0.328\%$, respectively. Their corresponding mean final widths are 533.3,

554.7, and 725.3 neurons. The reported variability is the sample standard deviation over seeds 42, 43, and 44.

These results separate the validity of the growth mechanism from the trajectory produced by a particular initialization. Exact preservation and immediate newborn trainability hold across all evaluated starting widths, while the sequence of growth events and the final capacity vary with the initial width. Initial width therefore affects how capacity is introduced during training, even though it does not alter the exactness of an individual paired-basis growth event.

E Extensions to Multiple Layers and Capacity Allocation

E.1 Growth Across Multiple Hidden Layers

We evaluate whether paired-basis growth extends beyond a single hidden layer using a two-hidden-layer LIF-SNN on SpiralHard. The experiment compares **Fixed-small**, **Fixed-max**, **cloned-zero-outgoing growth**, and **paired-basis growth** over seeds 42, 43, and 44.

The two growth methods start from physical widths $[128, 128]$ and follow the same predetermined schedule across layers. Four expansions are applied at epochs 42, 72, 102, and 132, with the selected layers ordered as $1, 2, 1, 2$. Each event adds 128 physical neurons to the selected layer, producing the trajectory $[128, 128] \to [256, 128] \to [256, 256] \to [384, 256] \to [384, 384]$.

Within each seed, the zero-outgoing and paired-basis methods use the same data partitions, minibatch order, optimizer settings, growth epochs, layer order, donor selection rule, and final physical architecture. The only difference between them is the downstream initialization of the newborn neurons. The cloned-zero-outgoing control sets their outgoing coordinates to zero, whereas paired-basis uses the exact difference and sum construction described in Section 3.2.

Across the two growth methods, all 24 layerwise expansion events satisfy the preservation conditions. At every event, the temporal logits and the membrane and spike trajectories of all existing neurons remain unchanged. Their model parameters and Adam state are also preserved, with a maximum recorded logit deviation of $\max_{\text{events}} \epsilon_z = 0$. The newly allocated Adam moments are initialized to zero, the new parameters are correctly incorporated into the optimizer, and the newborn neurons are active after expansion.

The backward behavior again distinguishes the two initialization rules. All 12 cloned-zero-outgoing events give zero gradients to the incoming weights of the newborn neurons. In contrast, all 12 paired-basis events produce nonzero gradient norms ranging from 6.366×10^{-4} to 1.727×10^{-1}, while the corresponding gradient opposition errors remain zero at the recorded precision. The multilayer setting therefore reproduces the same forward preservation and immediate learning behavior established for the single-layer case.

Table E.1 reports the mean results over three seeds. Paired-basis growth achieves a mean test accuracy of 99.800%, compared with 99.917% for Fixed-max and 99.733% for the matched cloned-zero-outgoing growth method.

Table E.1: Two-hidden-layer SpiralHard results averaged over seeds 42, 43, and 44. SynOps and spike counts are evaluated at the validation-selected checkpoint.

Method	Test acc. (%)	Final widths	Width-epochs	Parameter-epochs	Test SynOps	Test spikes
Fixed-small	98.950	$[128, 128]$	41,472	2.779M	38,081.9	551.2
Fixed-max	99.917	$[512, 512]$	165,888	42.965M	610,474.1	2,210.8
Cloned-zero-outgoing	99.733	$[384, 384]$	80,384	11.383M	153,531.4	1,630.9
Paired-basis	99.800	$[384, 384]$	80,384	11.383M	154,765.2	1,497.0

Relative to Fixed-max, paired-basis growth reduces width-epochs by 51.54%, parameter-epochs by 73.51%, SynOps at the selected checkpoint by 74.65%, and the test spike count by 32.29%. The corresponding difference in mean test accuracy is -0.117 percentage points. These comparisons show that capacity can be introduced progressively across multiple hidden layers without requiring the largest physical architecture to be present throughout training.

More importantly, the multilayer experiment confirms that the central mechanism is not restricted to a single hidden layer. Each paired-basis expansion preserves the temporal synaptic drive presented to the unchanged downstream network while retaining an immediate learning path to the newly introduced hidden parameters. This is consistent with the layerwise composition result in Appendix A.3.

E.2 Capacity Allocation with Different Growth Policies

The paired-basis mechanism specifies how physical expansion is carried out without disrupting the current temporal computation. The decision of when additional capacity should be introduced is separate from this mechanism and can be supplied by different allocation policies. We compare no growth, a fixed schedule, a validation-loss plateau rule, and a cost-aware policy on DoubleMoon and SpiralHard. Growth and model selection decisions use validation data only, and whenever an expansion occurs, the same paired-basis transaction is used. Across the 65 growth transactions generated in these experiments, the maximum recorded logit deviation at expansion is zero. For the plateau-based policies, growth decisions begin after epoch 30, with patience 10, a minimum validation-loss improvement of 0.01, and a 75-epoch cooldown. The fixed schedules use epochs 75 and 150 on DoubleMoon and epochs 75, 150, 225, 300, 375, and 450 on SpiralHard. For the cost-aware policy, each exploratory cohort is allowed to mature for 75 epochs. Further growth is permitted only when the best validation-accuracy gain exceeds λ times the added parameter fraction, with $\lambda = 0.01$ in the primary configuration.

The two tasks produce different patterns of capacity allocation, as summarized in Table E.2 and Figure E.1. DoubleMoon is already well served by a compact model. The cost-aware policy performs one expansion and finishes at width 32 while retaining 100% test accuracy. Relative to the plateau policy, this configuration reduces parameter-epochs by 16.78% and SynOps at the selected checkpoint by 32.98%.

SpiralHard requires substantially more capacity. Under the cost-aware policy, the final widths are 640, 768, and 768 for seeds 42, 43, and 44, respectively, giving a mean final width of 725.3. The resulting mean test accuracy is $99.617\% \pm 0.144\%$. Relative to the fixed schedule, the cost-aware policy reduces parameter-epochs by 12.04% and SynOps at the selected checkpoint by 18.01%.

Table E.2: Capacity allocation under four growth policies. Values are means over three seeds; selected width corresponds to the validation-selected checkpoint.

Task	Policy	Final width	Selected width	Test acc. (%)	Param.-epochs / SynOps
DoubleMoon	No growth	16.0	16.0	100.000	16,600 / 422.5
DoubleMoon	Fixed	48.0	48.0	100.000	30,600 / 1,237.7
DoubleMoon	Plateau	48.0	48.0	100.000	34,013 / 1,254.8
DoubleMoon	Cost-aware	32.0	32.0	100.000	28,307 / 840.9
SpiralHard	No growth	128.0	128.0	91.517	321,500 / 3,118.0
SpiralHard	Fixed	896.0	896.0	99.483	1,233,500 / 19,003.2
SpiralHard	Plateau	896.0	810.7	99.467	1,150,940 / 17,293.6
SpiralHard	Cost-aware	725.3	725.3	99.617	1,085,020 / 15,580.0

To examine how the cost assigned to additional capacity affects this allocation, we further vary the coefficient λ in the cost-aware policy. Increasing λ from 0.005 to 0.020 reduces the mean final width on SpiralHard from 853.3 to 597.3 and lowers SynOps at the selected checkpoint from 16,435.5 to 13,009.2. Over the same range, mean test accuracy changes from 99.583% to 99.500%, a difference of -0.083 percentage points. Table E.3 and Figure E.2 summarize how the amount of acquired capacity changes as its assigned cost increases.

The allocation experiments are not intended to define a unique growth policy. Instead, they demonstrate that the paired-basis expansion mechanism can be separated from the rule that decides when additional capacity is introduced. Different policies can therefore explore different capacity trajectories while using the same exact growth operation.

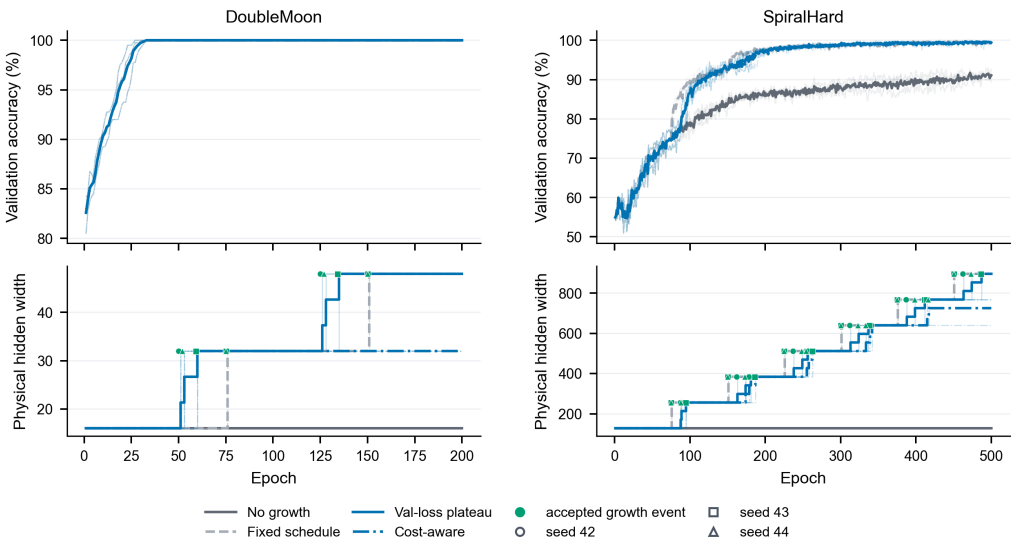

Figure E.1: Capacity allocation under different growth policies on DoubleMoon and SpiralHard. The paired-basis growth mechanism is unchanged across the policies; only the rule used to decide when additional capacity is introduced differs.

Table E.3: Sensitivity to the capacity cost coefficient on SpiralHard. Values are mean ± sample standard deviation over seeds 42, 43, and 44.

λ	Final width	Selected SynOps	Test acc. (%)
0.005	853.3 ± 60.3	$16{,}435.5 \pm 112.7$	99.583 ± 0.103
0.010	725.3 ± 73.9	$15{,}580.0 \pm 1{,}344.9$	99.617 ± 0.144
0.020	597.3 ± 60.3	$13{,}009.2 \pm 1{,}282.9$	99.500 ± 0.147

F Additional Evaluation Results

F.1 UCI HAR Transfer and Physical Compaction

We further evaluate paired-basis growth on the UCI Human Activity Recognition dataset (UCI HAR). Growing models are compared with independently trained fixed-width SNNs of widths 128, 256, 384, and 512 over seeds 42, 43, and 44. The growing checkpoints selected by validation performance achieve a mean test Macro-F1 of 0.92858 ± 0.00048, corresponding to a mean difference of $+0.00147$ relative to the fixed-width references selected in the same way. The selected widths are 128, 512, and 128 for the three seeds, respectively, although all growth trajectories eventually reach width 512. This shows that the capacity explored during training need not coincide with the capacity preferred by validation performance.

We next examine whether the final width-512 models can be converted into smaller physical networks after training. No retraining, fine-tuning, or modification of the source checkpoints is performed. The reported physical exports use the learned importance ranking, with the retained width selected using validation data only. For comparison, the compaction analysis additionally evaluates three alternative deterministic ranking rules. The resulting physical exports retain widths 128, 256, and 128 for seeds 42, 43, and 44, respectively. Their mean test Macro-F1 is 0.92360 ± 0.00081, corresponding to a signed mean change of -0.00322 ± 0.00417 relative to the final width-512 source checkpoints. All three exports reduce parameter count, test SynOps, and spike count. Table F.1 summarizes the resulting source and compact-export statistics.

The compaction analysis evaluates 48 deterministic candidates, comprising three seeds, four neuron-ranking rules, and four retained widths. For every candidate, the physically materialized network agrees with its corresponding logically selected subnetwork within the prescribed 10^{-5} numerical

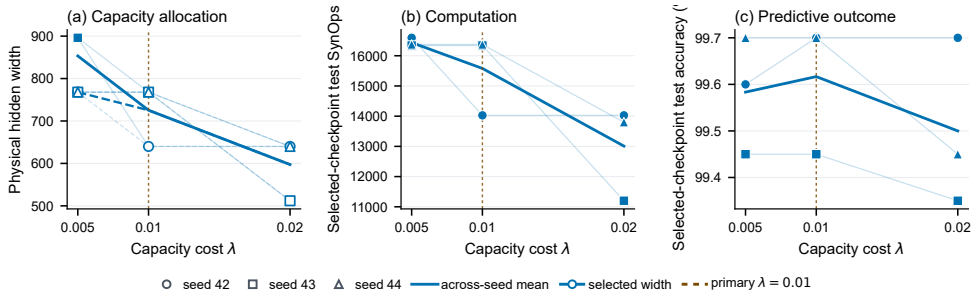

Figure E.2: Sensitivity to the capacity cost coefficient λ. Increasing the assigned cost reduces the physical width acquired during training and the corresponding SynOps, while predictive performance changes modestly over the evaluated range.

Table F.1: UCI HAR physical compaction from the final width-512 growth checkpoints. Width triplets follow seeds 42/43/44; values are mean $\pm$ sample standard deviation.

Model	Widths	Test Macro-F1	Parameters	Test SynOps	Test spikes
Final growth source	512/512/512	0.92682 ± 0.00420	290,822	$2.88668\text{M} \pm 0.00011\text{M}$	2392.6 ± 18.1
Compact physical export	128/256/128	0.92360 ± 0.00081	$96{,}945 \pm 41{,}976$	$0.96202\text{M} \pm 0.41685\text{M}$	762.5 ± 377.4

tolerance, and their class predictions are identical. The original width-512 source checkpoints remain unchanged throughout the analysis.

These results distinguish the capacity explored during optimization from the capacity retained in the final physical model. A growing model may temporarily acquire additional neurons during training, while validation performance can subsequently support a smaller physical realization without additional optimization.

F.2 Event-Based Evaluation on DVS Gesture

We additionally evaluate paired-basis growth on DVS Gesture, an event-based temporal classification benchmark. Growing SNNs are compared with independently trained fixed-width models of widths 128, 256, 384, and 512 over seeds 42, 43, and 44.

The structural properties of paired-basis growth also hold in this event-based setting. All nine growth events preserve the temporal predictor at the moment of expansion, with zero recorded maximum logit deviation. The existing model parameters and Adam state are also preserved at every event. Relative to training a width-512 model throughout optimization, progressive growth reduces cumulative width-epochs by 24.125%–28.125% across the three seeds.

The growing models eventually reach width 512 in all three runs and obtain test Macro-F1 scores of 0.6919, 0.7049, and 0.6763 for seeds 42, 43, and 44, respectively. The corresponding fixed-width references selected by validation performance achieve 0.7172, 0.6821, and 0.7087. The paired differences are therefore -0.0253, $+0.0228$, and -0.0324, giving a mean difference of -0.0116 ± 0.0301. Predictive performance therefore varies across seeds, while exact expansion and the reduction in cumulative width remain consistent. Table F.2 reports the seed-level results, while Figure F.1 compares predictive performance with physical width and cumulative width exposure.

Table F.2: Seed-level DVS Gesture evaluation. Fixed references are selected using validation data; all growing models reach width 512.

Seed	Growth F1	Fixed F1	Difference	Fixed width	Width-epoch reduction
42	0.6919	0.7172	-0.0253	256	28.125%
43	0.7049	0.6821	$+0.0228$	384	24.875%
44	0.6763	0.7087	-0.0324	384	24.125%
Mean $\pm$ SD	0.6910 ± 0.0143	0.7027 ± 0.0183	-0.0116 ± 0.0301	–	$25.71 \pm 2.13\%$

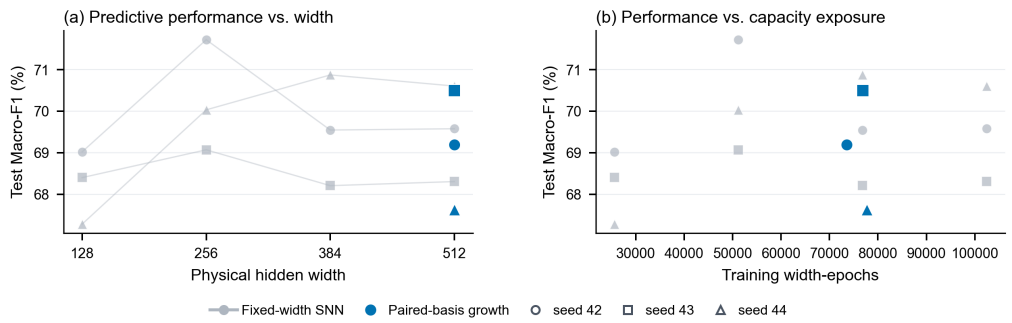

Figure F.1: Event-based evaluation on DVS Gesture. Panel (a) compares test Macro-F1 with physical width, and Panel (b) compares test Macro-F1 with cumulative width-epochs during training. Gray markers denote independently trained fixed-width references, while blue markers denote paired-basis growth. All growth events satisfy the exact preservation conditions, while progressive growth reduces cumulative width exposure relative to training at width 512 throughout.

Overall, the DVS Gesture experiments show that the exact structural properties of paired-basis growth extend to event-based temporal inputs. Although final predictive performance varies across seeds, every growth event preserves the existing temporal computation, and the growing models use less cumulative physical width during training than a model maintained at width 512 throughout.