Globally Optimal Training of Spiking Neural Networks via Parameter Reconstruction

Anonymous authors
Paper under double-blind review

Abstract

Spiking Neural Networks (SNNs) have been proposed as biologically plausible and energy-efficient alternatives to conventional Artificial Neural Networks (ANNs). However, the training of SNN usually relies on surrogate gradients due to the non-differentiability of the spike function, introducing approximation errors that accumulate across layers. To address this challenge, we extend the work on convexification of parallel feedforward threshold networks to parallel recurrent threshold networks, which subsume parallel SNNs as a structured special case. Building on this theoretical framework, we propose a parameter reconstruction algorithm for SNN training that demonstrates consistent and significant advantages across various tasks, both as a standalone method and in combination with surrogate-gradient training. The ablations further demonstrate the data scalability and robustness to model configurations of our training algorithm, pointing toward its potential in large-scale SNN training.

1 Introduction

Recent breakthroughs in Artificial Intelligence were enabled by powerful foundation models built with large-scale Artificial Neural Networks (ANNs), which, however, are generally inefficient in training and inference, and unable to model the internal dynamics across time which human brain is known to process. Spiking neural networks (SNNs) (Ghosh-Dastidar & Adeli, 2009), a category of neural networks with spiking neurons, hold great promise for addressing these limitations. Different from standard Artificial Neural Networks (ANNs), where each neuron outputs a numerical scalar transformed from inputs, a spiking neuron generates a discrete time series to represent information, where the dynamics over time are governed by some differential equations. A most commonly used spiking neuron is Leaky-Integrate-and-Fire (LIF) model, where the representations are binary sequences.

There are several desiderata about spiking neurons and SNNs. A spiking neuron simulates the behaviors that a biological neuron in human brains processes electric signals. The spiking neuron integrates incoming signals, causing its membrane potential to rise and fall over time. When the accumulated membrane potential exceeds a threshold, the neuron "fires", emitting a discrete spike (action potential) as in biological neurons. Second, the sequential representations generated by SNNs are believed to be more expressive than static values in ANNs, especially on time series tasks and tasks that require memories. Finally, it is regarded as an energy-efficient architecture, as the binary representations reduce the computations in SNNs to simple logical operations that can be computed at a low cost. Therefore, SNNs are believed to be highly promising for shaping the next-generation AI architectures (Schuman et al., 2022; Mehonic & Kenyon, 2022; Eshraghian et al., 2023; Pfeiffer & Pfeil, 2018).

Despite of great promise, there are a few fundamental challenges that have blocked the development of SNNs, especially to the scale on par with modern Large Language Models (LLMs). On top of them is the lack of effective training algorithms. The indifferentiability of the spike functions makes the direct use of gradient backpropagation infeasible. As a result, surrogate gradients (Neftci et al., 2019) are commonly used in modern SNN training, where a continuous surrogate function (usually a sigmoid function) is adopted to approximate the spike function during backpropagation to compute the surrogate of "real gradients". Due to

this forward-backward mismatch, parameters in SNNs are not necessarily optimized to optimal directions, and such mismatching error accumulates across layers, making the training of deeper SNNs even more difficult (Deng et al., 2023).

To address this problem, we propose a novel training paradigm for SNNs. The paradigm first enumerates all realizable spike activation patterns of the hidden layers into a finite spike dictionary, then solves a convex optimization problem over this dictionary to recover the globally optimal output coefficients. The theoretical foundation behind rests on two of our contributions: we extend the convexification theory of feedforward parallel threshold networks (Ergen et al., 2023) to parallel recurrent threshold networks, and further prove that SNNs based on LIF neurons (LIF-SNNs) are a structured special case of this recurrent framework, establishing the global optimality of parameter reconstruction as an SNN training algorithm. To evaluate whether the proposed paradigm leads to better SNNs in practice, we conduct experiments across a range of tasks and model configurations. Results show consistent and significant advantages over surrogate-gradient baselines, and reveal that the two approaches are complementary: combining parameter reconstruction with surrogate-gradient training yields further performance gains.

Our main contributions are as follows:

- **Theory.** We extend the convexification of overparameterized parallel threshold networks (Ergen & Pilanci) to feedforward threshold networks, and prove zero-duality gap for overparameterized parallel recurrent threshold networks under weight decay. We further prove that parallel LIF-SNNs are a structured special case of this recurrent framework, establishing zero-duality gap for SNN training.
- **Algorithm.** We propose a new globally optimal training algorithm for parallel LIF-SNNs via parameter reconstruction, enabled by the zero-duality gap result above.
- **Experiments.** Evaluation results demonstrate consistent performance gains over surrogate-gradient baselines across tasks and model configurations, both as a standalone method and combined with surrogate-gradient training. Ablations reveal positive data scaling and robustness to model configurations of our algorithm, and expose a performance ceiling in models trained with surrogate gradient baselines that persists regardless of data volume.

2 Related Works

2.1 Training Algorithms for Spiking Neural Networks

The major bottleneck of SNN training is the non-differentiability of the spike functions. One way to handle this problem is through surrogate gradients, where gradients are computed via continuous approximations such as arc-tangent or sigmoid functions to facilitate backpropagation (Neftci et al., 2019; Bohté et al., 2000; Wu et al., 2018). However, to the best of our knowledge, there is no analytical or theoretical study that establishes comparable optimality conditions for model weights after surrogate-gradient SNN training. Moreover, the approximation error between forward and backward computations in surrogate-gradient methods accumulates across model layers and lowers model quality (Deng et al., 2023).

Another category of approaches to this problem is through ANN-SNN conversion, which an ANN is trained first and then converted to an SNN by replacing artificial neurons with spiking neurons (Cao et al., 2015; Li & Zeng, 2022). These methods do not train the temporal dynamics directly and typically require a large number of timesteps to approximate the source ANN (Bu et al., 2023), and suffer from significant performance degradation under low-latency restrictions (Rathi & Roy, 2020). ANN-SNN conversion approaches are most applicable when a pretrained ANN is available, transferring its learned representations directly to the SNN. As our work targets the training of SNN dynamics from scratch, we exclude conversion-based methods from our baselines.

2.2 Convex Reconstruction of Neural Networks

A line of work establishes globally optimal training of overparameterized parallel neural networks via convex reformulation, including feedforward networks (Ergen & Pilanci, 2021a; 2025; Wang & Pilanci, 2021),

CNNs (Ergen & Pilanci, 2021b), and Transformers (Ergen et al., 2022). However, neural networks with recurrent structures such as RNNs and SNNs remain underexplored in this framework. The foundations most relevant to our work are Ergen & Pilanci, which establishes global optimality for path-regularized parallel feedforward ReLU networks building on the path regularizer of Tomioka & Srebro, and Neyshabur et al. (2016), which applies path normalization to recurrent ReLU networks. Ergen et al. (2023) further extends this to feedforward parallel threshold networks under weight-decay regularization. Our work extends these results to parallel recurrent threshold networks under weight-decay regularization, and establishes the first convex reformulation of LIF-SNN training.

3 Globally Optimal Training of Spiking Neural Networks

3.1 Notation and Preliminaries

For $m \in \mathbb{N}$, let $[m] = \{1, \dots, m\}$. We write $\mathbb{I}\{\cdot\}$ for the indicator function, $\|\cdot\|_p$ for the ℓ_p norm, and p^* for the dual exponent. The training set has n samples. For feedforward networks, inputs are given by $X \in \mathbb{R}^{n\times d}$; for recurrent or spiking neural networks, inputs are denoted as $X^{1:T} = (X^1, \dots, X^T)$ with each $X^t \in \mathbb{R}^{n\times d}$. Labels are collected in Y, and all losses $\mathcal{L}(\cdot, Y)$ are assumed convex in their first argument. The scaled threshold activation is denoted as $\sigma_s(x) = s\mathbb{I}\{x \geq 0\}$, with s a trainable amplitude. The scale of s may be absorbed into the outgoing weights of each hidden layer, and we write $\sigma(x) = \mathbb{I}\{x \geq 0\}$ for the unit-amplitude gate. The key property used throughout is *positive scaling invariance*: $\sigma(cx) = \sigma(x)$ for every $c > 0$. Thus, positively rescaling the pre-activation of any threshold neuron leaves its output unchanged.

We represent a neural network as a weighted Directed Acyclic Graph (DAG), $G(V, E, w)$, with input nodes V_{in} and output nodes V_{out}. An edge $e = (u, v) \in E$ has a weight $w(e)$.

For a fully connected feedforward threshold network of depth L and width H, $f^{L,\Theta}(X) = \sigma(\cdots\sigma(\sigma(XW_1)W_2)\cdots W_{L-1})\,W_L$, the group-norm regularizer of Neyshabur et al. (2016) with norm p is defined over the DAG as

$$\mu_{p,q}(\Theta) = \left(\sum_{v \in V \setminus V_{\text{in}}} \Big(|s_v|^p + \sum_{(u,v)\in E} |w(u,v)|^p \Big)^{q/p} \right)^{1/q},$$

the ℓ_q aggregation across nodes of each node's incoming parameters (its weight column and amplitude s_v) in ℓ_p. We use the one with $q = 2$,

$$\mu_{p,2}(\Theta)^2 = \sum_{v \in V \setminus V_{\text{in}}} \Big(|s_v|^p + \sum_{(u,v)\in E} |w(u,v)|^p \Big)^{2/p},$$

which is an additive sum over nodes of squared incoming ℓ_p blocks; for $p = 2$ this is ordinary squared weight decay. The reduction below uses only this additivity across nodes together with the positive scaling invariance of the threshold, recovering the output-layer norm by the additive zero-out of inner magnitudes. A K-parallel threshold network extends such a feedforward network to one consisting of subnetworks $\{G_k(V_k, E_k, w_k)\}_{k=1}^K$ that share input nodes but have disjoint hidden parameters. Its output and weight-decay regularizer are naturally written as $f^{L,K,\Theta}(X) = \sum_{k=1}^K f^{L,k,\Theta_k}(X)$ and $\mu_{p,2}(\Theta)^2 = \sum_{k=1}^K \mu_{p,2}(\Theta_k)^2$ respectively.

Theorem 3.1 (Reduction to outer norms for K-parallel networks)**.** *Let $G(V, E, w)$ be a fully connected K-parallel threshold network with weight-decay regularizer $\mu_{p,2}(\Theta)^2 = \sum_{k=1}^K \mu_{p,2}(\Theta_k)^2$, where each hidden neuron carries a trainable amplitude. Then the weight-decay regularized training problem is equivalent, under the gauge $|s_k^{(L-1)}| = 1$ on the last amplitude of each subnetwork, to the problem in which the regularizer is the outer-layer norm:*

1. *$f^{L,K,\Theta}(X) = f^{L,K,\bar{\Theta}}(X)$ for all inputs X;*

2. *the inner weight magnitudes* $\{\|W_{l,k}\|_p\}_{l\le L-1}$ *and inner amplitudes* $\{s_k^{(l)}\}_{l\le L-2}$ *are driven to zero additively, leaving*

$$\min_{\Theta} \mathcal{L} + \tfrac{\beta_{\text{reg}}}{2}\mu_{p,2}(\Theta)^2 = \min_{\Theta:\, |s_k^{(L-1)}|=1} \mathcal{L} + \beta_{\text{reg}} \sum_{k=1}^{K} \|W_{L,k}\|_p.$$

Thus, the original weight-decay regularized training objective

$$p^L = \min_{\Theta}\ \mathcal{L}(f^{L,K,\Theta}(X), Y) + \tfrac{\beta_{\text{reg}}}{2}\mu_{p,2}(\Theta)^2$$

reduces to

$$p^L = \min_{\Theta}\ \mathcal{L}(f^{L,K,\Theta}(X), Y) + \beta_{\text{reg}} \sum_{k=1}^{K} \|W_{L,k}\|_p.$$

This objective is non-convex and admits no direct theoretical guarantees on the quality of recovered solutions. With the objective above denoted as the *primal* problem, we construct the *dual* and *bidual* problems and prove zero-duality gap among the three, which implies that the globally optimal solution to the non-convex primal can be recovered by solving the strictly convex bi-dual problem.

Theorem 3.2 (Bi-dual of parallel feedforward threshold networks)**.** *Let* $\phi_k(X;\theta_k)$ *denote the final hidden-layer feature map of subnetwork* k*:* $\phi_k(X;\theta_k) = \sigma(\cdots\sigma(\sigma(XW_{1,k})W_{2,k})\cdots W_{L-1,k})$*, and* $\phi(X;\theta) = \{\phi_k(X;\theta_k)\}_{k=1}^K$ *by applying Theorem 3.1. Let* $d^{L,K}$ *denote the reduced weight-decay regularized training problem for a* K*-parallel feedforward threshold network with convex loss* $\mathcal{L}$*, with a regularization parameter* $\beta_{\text{reg}} > 0$ *obtained. Then the dual problem is*

$$d^{L,K} = \max_{\lambda}\ -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\theta_k\in\bar{\Theta}_k} \|\lambda^\top \phi_k(X;\theta_k)\|_{p^*} \le \beta_{\text{reg}}, \qquad k\in[K].$$

The corresponding infinite bi-dual problem is

$$p_\infty^{L,K} = \min_{\mu}\left\{\mathcal{L}\left(\int_{\theta\in\bar{\Theta}} \phi(X;\theta)\, d\mu(\theta), Y\right) + \beta_{\text{reg}}\|\mu\|_{\text{TV}}\right\},$$

where μ *is a finite signed measure over reduced feature maps. Since* $\beta_{\text{reg}} > 0$*, strong duality holds:*

$$d^{L,K} = p_\infty^{L,K}.$$

Moreover, for scalar outputs, if $K \ge K^*$ *for some* $K^* \le n+1$*, then*

$$p^{L,K} = d^{L,K} = p_\infty^{L,K}.$$

For d_{out} *output coordinates, the same Carathéodory argument gives* $K^* \le nd_{\text{out}} + 1$ *after vectorizing the predictions.*

The proof is given in Appendix B. The zero-duality-gap equality follows from Slater's condition at $\lambda = 0$, and the finite-width equivalence follows from Carathéodory's theorem. The global optimality proven in Theorem 3.2 will be used as the basis for the recurrent and spiking neural network reconstructions below.

3.2 Threshold Recurrent Neural Networks

We consider K-parallel recurrent threshold networks with fixed timestep T and depth L. For subnetwork $k\in[K]$, layer $l\in[L-1]$, and timestep $t\in[T]$, the hidden state evolves as

$$H_{k,l}^t = \sigma\left(H_{k,l-1}^t P_{k,l,\text{in}} + H_{k,l}^{t-1} P_{k,l,\text{rec}}\right), \qquad H_{k,0}^t = X^t.$$

Unless stated otherwise, $H_{k,l}^0 = 0$. The final-time readout is

$$f^{L,T,K,\Theta}(X^{1:T}) = \sum_{k=1}^{K} H_{k,L-1}^T P_{k,\text{out}},$$

where $P_{k,l,\text{in}} \in \mathbb{R}^{m_{l-1,k} \times m_{l,k}}$, $P_{k,l,\text{rec}} \in \mathbb{R}^{m_{l,k} \times m_{l,k}}$, and $P_{k,\text{out}} \in \mathbb{R}^{m_{L-1,k} \times d_{\text{out}}}$.

A fixed-timestep recurrent network can be represented as an unrolled DAG with tied edge weights across timesteps. The only difference from the feedforward case is that each hidden neuron i in layer l and subnetwork k has two tied incoming blocks, $P_{k,l,\text{in}}[:, i]$ and $P_{k,l,\text{rec}}[:, i]$, which the reduction scales jointly. Scaling both by a common $\alpha > 0$ scales the whole pre-activation by α, which the threshold absorbs, so the network function is unchanged. Hence the joint magnitude

$$a_{k,l,i} = \left(\|P_{k,l,\text{in}}[:, i]\|_p^p + \|P_{k,l,\text{rec}}[:, i]\|_p^p \right)^{1/p}$$

may be pushed to zero, playing the role of the incoming weight column in Theorem 3.1. With Theorem 3.1 and Theorem 3.2 applied to the unrolled DAG, we similarly prove strong duality for parallel recurrent threshold networks.

Theorem 3.3 (Bi-dual of parallel recurrent threshold networks)**.** *Let $p^{L,T,K}$ denote the reduced weight-decay regularized recurrent training problem above, with convex loss $\mathcal{L}$ and regularization parameter $\beta_{\text{reg}} > 0$. Theorem 3.2 applies directly by substituting $\phi_k(X; \theta_k)$ with $\phi_{T,k}(X^{1:T}; \theta_k)$. Since $\beta_{\text{reg}} > 0$, strong duality holds:*

$$d^{L,T,K} = p_\infty^{L,T,K}.$$

Moreover, for scalar outputs, if $K \geq K^*$ for some $K^* \leq n + 1$, then

$$p^{L,T,K} = d^{L,T,K} = p_\infty^{L,T,K}.$$

For d_{out} output coordinates, the same Carathéodory argument gives $K^* \leq nd_{\text{out}} + 1$ after vectorizing the predictions.

The proof is given in Appendix C.2. The strong duality in Theorem 3.3 is based on an infinite number of subnetworks. To obtain a finite convex formulation, we adopt the practice in Ergen et al. (2023) to obtain the *spike dictionary*, denoted as $D^{L-1,T} \in \{0, 1\}^{n \times P^{L-1,T}}$ whose $P^{L-1,T}$ columns are the distinct spike activation patterns at layer $L - 1$, timestep T. We call this process *witness generation* and the details are provided in Appendix C.3. Note that $P^{L-1,T}$ is finite because every column of $D^{L-1,T}$ lies in $\{0, 1\}^n$.

Theorem 3.4 (Finite convex formulation for recurrent threshold networks)**.** *For weight decay with $p = 2$, let $D^{L-1,T} \in \{0, 1\}^{n \times P^{L-1,T}}$ be the complete witnessed arrangement matrix at the final hidden layer and final timestep. Then the reduced recurrent training problem admits the finite convex bi-dual objective*

$$\widetilde{p}^{L,T,K} = \min_{\widetilde{w} \in \mathbb{R}^{P^{L-1,T}}} \mathcal{L}(D^{L-1,T}\widetilde{w}, Y) + \frac{\beta_{\text{reg}}}{\sqrt{m_{L-1}}} \|\widetilde{w}\|_1.$$

Moreover, for scalar outputs,

$$\widetilde{p}^{L,T,K} = p_\infty^{L,T,K}$$

whenever $K \geq K^*$ for some $K^* \leq n + 1$. For d_{out} output coordinates, the corresponding sufficient bound is $K^* \leq nd_{\text{out}} + 1$.

The proof is given in Appendix C.4.

3.3 Spiking Neural Networks as Structured Recurrent Threshold Networks

In this section, we provide a perspective to view an SNN as a structured recurrent threshold network, which allows us obtain the globally optimal solution to an SNN under the similar weight-decay regularized objective as defined in §3.1, via Theorem 3.3 and Theorem 3.4.

We use the threshold-centered membrane convention $S_l^t = \sigma(U_l^t) = \mathbb{I}\{U_l^t \geq 0\}$. For layer l and timestep t, the LIF recurrence is

$$U_l^t = S_{l-1}^t P_{l,\text{in}} + U_l^{t-1} B_l - S_l^{t-1} R_l, \qquad S_l^t = \sigma(U_l^t),$$

where $B_l = \mathrm{Diag}(\beta_l)$ where $\beta_l < 1$ is the leak coefficient, $R_l = \mathrm{Diag}(U^l_{\mathrm{thr}})$ is the soft-reset matrix and U^l_t, U^{t-1}_l represent the membrane potentials of the neuron. Equivalently,

$$U^t_l = S^t_{l-1} P_{l,\mathrm{in}} + \begin{bmatrix} U^{t-1}_l & S^{t-1}_l \end{bmatrix} \begin{bmatrix} B_l \\ -R_l \end{bmatrix}.$$

Thus, an LIF-SNN is a recurrent threshold network whose recurrent matrix is constrained to the leak-reset block form above, rather than being an arbitrary dense recurrent matrix.

Lemma 3.5 (LIF membrane rescaling)**.** *Let $A_l = \mathrm{Diag}(a_{l,1}, \ldots, a_{l,m_l})$ with $a_{l,i} > 0$. Define*

$$\bar{U}^t_l = U^t_l A_l, \qquad \bar{P}_{l,\mathrm{in}} = P_{l,\mathrm{in}} A_l, \qquad \bar{R}_l = R_l A_l, \qquad \bar{B}_l = A_l^{-1} B_l A_l.$$

Then the rescaled variables satisfy

$$\bar{U}^t_l = S^t_{l-1} \bar{P}_{l,\mathrm{in}} + \bar{U}^{t-1}_l \bar{B}_l - S^{t-1}_l \bar{R}_l.$$

Moreover, if B_l is diagonal, then $\bar{B}_l = B_l$, and the spike outputs are unchanged:

$$\sigma(\bar{U}^t_l) = \sigma(U^t_l A_l) = \sigma(U^t_l).$$

The proof is given in Appendix D.2. By Lemma 3.5 the leak $\beta_l \in B_l$ is a bounded structural recurrent coefficient, not a trainable weight, and is unaffected by rescaling. The trainable parameters $P_{l,\mathrm{in}}[:, i]$ and $U^l_{\mathrm{thr},i}$ feed only U^t_l, so by positive scaling invariance their magnitudes and the inner amplitudes may be pushed to zero. Applying Theorem 3.1 on the unrolled LIF DAG,

$$\min_{\Theta} \mathcal{L} + \tfrac{\beta_{\mathrm{reg}}}{2} \mu_{p,2}(\Theta)^2 = \min_{\Theta:\, |s^{(L-1)}_k| = 1} \mathcal{L} + \beta_{\mathrm{reg}} \sum_{k=1}^{K} \|P_{k,\mathrm{out}}\|_p.$$

The reduction is proved in Appendix D.3. Theorem 3.3 then applies by substituting $\phi_{T,k}(X^{1:T}; \theta_k)$ with $S^T_{k,L-1}$; the construction and finiteness are detailed in Appendix D.4.

Theorem 3.6 (Finite convex formulation for LIF-SNNs)**.** *For weight decay with $p = 2$, let $D^{L-1,T}_{\mathrm{SNN}} \in \{0,1\}^{n \times P^{L-1,T}_{\mathrm{SNN}}}$ be the complete SNN witnessed dictionary at the final hidden layer and final timestep. Then the reduced SNN training problem with weight-decay regularization admits the similar finite convex program as defined in Theorem 3.4*

$$\widetilde{p}^{L,T}_{\mathrm{SNN}} = \min_{\widetilde{w} \in \mathbb{R}^{P^{L-1,T}_{\mathrm{SNN}}}} \mathcal{L}(D^{L-1,T}_{\mathrm{SNN}} \widetilde{w}, Y) + \frac{\beta_{\mathrm{reg}}}{\sqrt{m_{L-1}}} \|\widetilde{w}\|_1.$$

For scalar outputs,

$$\widetilde{p}^{L,T}_{\mathrm{SNN}} = p^{L,T,K}_{\mathrm{SNN}}$$

whenever $K \geq K^*$ for some $K^* \leq n + 1$. For d_{out} output coordinates, the sufficient bound is $K^* \leq n d_{\mathrm{out}} + 1$ after vectorizing the predictions.

3.4 Parameter Reconstruction for Spiking Neural Network Training

We now demonstrate how the optimal weights of an SNN can be obtained from the solution of the convex program in Theorem 3.6. From Carathéodory's argument as stated in §3.1, the solution to this convex problem, $\widetilde{w}^*$, has $n + 1$ non-zero coefficients at most that form a linear combination of spike activations d_i which belongs to the LIF witnessed dictionary $D^{L-1,T}_{\mathrm{SNN}}$. Since $K^* \leq n + 1$ and $K \geq K^*$ we can assign each of these activations to each subnetwork $k \in [K]$. The associated witness ω_i is then initialized as the weights of the subnetwork k.

The exact algorithm applies to the complete witnessed dictionary and gives the globally optimal solution to all model parameters. However, in practice, an exhaustive enumeration of all spike activations is not tractable as the computational complexity grows combinatorially, as derived in the appendix D.6. So, we use fixed or

pretrained witnesses. We generate fixed witnesses by sampling all parameters of the SNN except for the last layer from a Gaussian distribution similar to the practice in Ergen et al. (2023), and generate pretrained witnesses by directly transferring the model parameters of a pretrained SNN trained on the same task. Once these witnesses are fixed, the induced spike dictionary is tractable in size, and the convex construction stage is globally optimal over this spike dictionary, not necessarily for the complete intractable dictionary.

4 Experiments

In this section, we empirically evaluate our parameter reconstruction algorithm for SNN training. We check that the global optimality is achieved by examining the gap between primal and dual objectives, and investigate when such global optimality on training data is translated into better generalization performance than surrogate-gradient baselines. We further examine the two-part training paradigm proposed in §3.4 and check if better witness generation empirically improves model performance.

Tasks. Our main task is arithmetic addition under sequence-length in-distribution (ID) and out-of-distribution (OOD) evaluation, a standard probe of algorithmic length generalization in sequence models (Lee et al., 2023; Kazemnejad et al., 2023; McLeish et al., 2024; Anil et al., 2022). For two n-digit base-b operands, the input space grows exponentially as b^{2n}, so long-sequence OOD performance cannot be explained only by memorizing short training examples given limited model capacity. We also evaluate on FIRST-LAST-XOR and MNIST-SEQ (LeCun et al., 2010) as representative tests on synthetic and non-synthetic tasks respectively.

Method. We evaluate two variants of our parameter reconstruction method: CVX and SG-CVX. CVX generates witnesses via random sampling from a Gaussian Distribution, while SG-CVX uses the spike dictionary from an SNN trained with surrogate gradients on the target task. Then both CVX and SG-CVX optimally reconstruct output coefficients over their respective spike dictionaries. We compare against two surrogate-gradient baselines: SG, the standard surrogate-gradient training method, and SG-SG, which continues surrogate-gradient training from an SG checkpoint as a fair comparison to SG-CVX. Algorithmic details are given in Appendix E.

Main Results. Table 1 verifies the finite convex reconstruction problem is solved to global optimality. Across the number of subnetworks, the primal–dual gap is consistently below 10^{-6} on FIRST-LAST-XOR.

Table 1: Empirical optimality of the convex reconstruction solve on FIRST-LAST-XOR ($T = 11$, $L = 3$). We sweep number of subnetworks K at fixed total width. The primal and dual values are rounded to four decimal places and the precise gaps are below 10^{-6}, numerically verifying global optimality of the sampled convex problem.

K	Test Acc. (%)	Primal	Dual	Abs. gap
2	74.71	0.9817	0.9817	7.01×10^{-8}
16	82.23	0.8370	0.8370	2.94×10^{-7}
32	76.07	0.9459	0.9459	9.55×10^{-8}
64	86.43	0.8734	0.8734	2.65×10^{-7}

Tables 2 and 3 report the main empirical results. In ID settings, CVX-derived methods outperform the best SG-derived variants across addition bases and model configurations on ADDITION. On FIRST-LAST-XOR, the largest gap appears at moderate timestep and large depth: at $L = 10, T = 6$, CVX reaches 0.973 accuracy while SG is near chance at 0.501. On MNIST-SEQ, CVX remains competitive at shallow depth and retains a nontrivial advantage at $L = 10$, where SG collapses to chance.

The OOD results on ADDITION confirm the necessity and effectiveness of the two-part training paradigm. As shown in Table 3, CVX is strong near the training distribution and degrades dramatically on longer sequence test, while SG-CVX results in much more stable OOD performances even on test lengths that are 10 times the training lengths.

Table 2: Experiment results on synthetic and non-synthetic tasks. Each cell reports test accuracy of the best variant within its method class. Best CVX-variant selects the best performance metrics across {CVX, SG-CVX} and Best SG-variant selects across {SG, SG-SG}. Addition tasks report in-distribution token-wise accuracy at $n = 5$.

Task	**Timestep**	**Best CVX-variant**	**Best SG-variant**
Depth $L = 3$			
$\text{ADDITION}_{\text{Base-2}}$	$T = 6$	**0.954**	0.738
$\text{ADDITION}_{\text{Base-3}}$	$T = 6$	**0.363**	0.212
$\text{ADDITION}_{\text{Base-5}}$	$T = 6$	**0.235**	0.189
FIRST-LAST-XOR	$T = 6$	**1.000**	0.746
FIRST-LAST-XOR	$T = 14$	**0.689**	0.641
MNIST-SEQ	$T = 2$	**0.925**	0.913
Depth $L = 10$			
$\text{ADDITION}_{\text{Base-2}}$	$T = 6$	**0.500**	0.103
$\text{ADDITION}_{\text{Base-3}}$	$T = 6$	**0.269**	0.239
$\text{ADDITION}_{\text{Base-5}}$	$T = 6$	**0.144**	**0.144**
FIRST-LAST-XOR	$T = 6$	**0.973**	0.501
FIRST-LAST-XOR	$T = 14$	**0.504**	**0.504**
MNIST-SEQ	$T = 2$	**0.271**	0.099

Table 3: Token-wise accuracy on arithmetic addition across training variants in OOD lengths.

Base	Group	Method	ID (n=5)	OOD-2×	OOD-5×	OOD-10×
$b = 3$	SG-variant	SG	0.217	0.197	0.182	0.177
		SG-SG	0.238	0.220	0.207	0.202
	CVX-variant	CVX	**0.366**	0.302	0.254	0.234
		SG-CVX	0.340	**0.314**	**0.307**	**0.310**
$b = 5$	SG-variant	SG	0.159	0.148	0.142	0.141
		SG-SG	0.178	0.163	0.152	0.153
	CVX-variant	CVX	0.218	0.164	0.135	0.125
		SG-CVX	**0.230**	**0.188**	**0.173**	**0.172**

5 Analysis and Ablation Study

The experiments in §4 show that the models trained with CVX-variants reach global optimality on training data. In this section, we further ablate and analyze key factors that influence when such a theoretical property is translated into better generalization performance.

5.1 Ablations on Model Depth, Timestep and Size of Training Set

Tables 4 and 5 examine how model depth L and timestep horizon T each affect performance of models trained with CVX and SG. We use FIRST-LAST-XOR as a controlled benchmark, as its fixed binary input distribution ensures that varying L or T in isolation cleanly probes two factors respectively. MNIST-SEQ serves as the sensory counterpart, where each image is streamed as sequential pixel patches so that increasing T spreads the same information across more recurrent steps.

Depth scaling. As shown at the left in Table 4 and 5, baseline models with SG degrade sharply with depth, where accuracy collapses to chance when increasing L to 10 on both tasks. On the contrary, CVX degrades more gracefully and remains significantly better performance on deep models.

Table 4: FIRST-LAST-XOR ablations. Left: Depth ablation at fixed timestep $T = 6$. Right: Timestep ablation at fixed depth $L = 3$. Test accuracy is reported.

(a) Depth ablation with $T = 6$

Depth L	CVX	SG
3	**1.000**	0.746
5	**1.000**	0.744
10	**0.973**	0.501
15	**0.841**	0.504

(b) Timestep ablation with $L = 3$

Timestep T	CVX	SG
6	**1.000**	0.746
8	**1.000**	0.724
11	**0.864**	0.622
14	**0.689**	0.640

Table 5: MNIST-SEQ ablations. Left: Depth ablation at fixed timestep $T = 2$. Right: Timestep ablation at fixed depth $L = 3$. Test accuracy is reported.

(a) Depth ablation, fixed $T = 2$

Depth L	CVX	SG
3	**0.925**	0.913
5	**0.844**	0.740
10	**0.271**	0.099

(b) Timestep ablation, fixed $L = 3$

Timestep T	CVX	SG
14	**0.803**	0.7835
28	**0.831**	0.763
56	**0.821**	0.687

Timestep scaling. Reported at the right in Table 4 and 5, CVX maintains a consistent advantage over SG across all values of T, confirming that its optimization benefits are robust to the choice of timestep horizon.

Size of training set. We further study the effect of training set size on model performance in Table 6. We find that SNNs trained with CVX exhibit better generalization performance as the number of training samples increases, whereas SNNs trained with SG show no such benefit. This finding suggests that CVX may facilitate scaling SNN training in a manner analogous to how modern LLMs benefit from larger datasets, which is left for future work.

Table 6: Test accuracy on base-2 arithmetic addition as numbers of training samples vary on models with L=3, T=6. Both sequential accuracy (correct only if every output token in an addition sequence is correct) and token-wise accuracy are reported. Darker numbers indicate better performance within each column, showing the trend when scaling against number of training samples.

# train	CVX (sequential)	CVX (token-wise)	SG (sequential)	SG (token-wise)
512	0.571	0.907	0.160	0.737
2304	0.717	0.951	0.183	0.744
4096	0.730	0.954	0.163	0.738
5888	0.725	0.954	0.184	0.738
7680	0.728	0.954	0.152	0.738

5.2 Surrogate Gradients as High-quality Witness Generators

An important lesson we learned from empirical results in §4 is that CVX achieves its full potential when used on top of a high-quality witness. As we observe in Table 3, SG-CVX with witnesses transferred from another trained SNN generalizes better than CVX with randomly sampled witnesses. One possible mechanism is via spectral simplicity: recent work connects LIF-SNN stability to low-frequency Fourier–Walsh structure (Araya et al., 2025); if pretrained witnesses generate simpler or more stable boolean features, convex reconstruction may inherit that low-frequency structure through the fixed dictionary. We leave the proof of such a spectral generalization bound in the future work.

6 Conclusion

We propose a globally optimal training algorithm for SNNs that does not suffer from the error accumulation as in surrogate-gradient training. We achieve it by extending the bi-dual framework in weight-decay regularized training objectives from parallel feedforward ReLU networks to parallel recurrent threshold networks and parallel SNNs, and make it computationally tractable by using fixed or transferred witnesses. Experiments confirm the global optimality empirically and show that the proposed new algorithm outperforms baselines across various tasks and model configurations. Ablations further show the robustness of our algorithm across the choices of model configurations and the data scalability, indicating the potential to be adopted in large-scale training of SNNs in the future.

Limitations. The global optimality we establish is exact only over the *complete* witnessed dictionary, whose enumeration grows combinatorially in the number of training samples (Appendix D.6) and is therefore intractable in practice. The CVX and SG-CVX variants instead solve the convex program over a *sampled* dictionary induced by a finite set of fixed or pretrained witnesses, and therefore the quality of the reconstructed model is bounded by the quality of the witness set, and characterizing the gap between the reconstructed model and true optimum remains open. Moreover, our current implementation relies on a general-purpose convex solver for the output-layer reconstruction, which does not exploit modern accelerator hardware. Developing GPU-friendly algorithms tailored to the dictionary program is a promising direction for scaling parameter reconstruction to larger SNNs. Finally, our evaluation covers arithmetic addition, sequential MNIST, and XOR, but not speech or large-scale vision, and the convex reduction targets threshold-RNNs with the block recurrent structure rather than gated (LSTM-style) or attention-augmented SNNs.

References

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556, 2022.

Ernesto Araya, Massimiliano Datres, and Gitta Kutyniok. Random Spiking Neural Networks are Stable and Spectrally Simple, November 2025. URL `http://arxiv.org/abs/2511.00904`.

Sander M. Bohté, Joost N. Kok, and Han La Poutré. Spikeprop: backpropagation for networks of spiking neurons. In *The European Symposium on Artificial Neural Networks*, 2000. URL `https://api.semanticscholar.org/CorpusID:14069916`.

Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks, 2023. URL `https://arxiv.org/abs/2303.04347`.

Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *Int. J. Comput. Vision*, 113(1):54–66, May 2015. ISSN 0920-5691. doi: 10.1007/s11263-014-0788-3. URL `https://doi.org/10.1007/s11263-014-0788-3`.

Hanseul Cho, Jaeyoung Cha, Pranjal Awasthi, Srinadh Bhojanapalli, Anupam Gupta, and Chulhee Yun. Position coupling: Improving length generalization of arithmetic transformers using task structure. *arXiv preprint arXiv:2405.20671*, 2024a.

Hanseul Cho, Jaeyoung Cha, Srinadh Bhojanapalli, and Chulhee Yun. Arithmetic transformers can length-generalize in both operand length and count. *arXiv preprint arXiv:2410.15787*, 2024b.

Shikuang Deng, Hao Lin, Yuhang Li, and Shi Gu. Surrogate module learning: Reduce the gradient error accumulation in training spiking neural networks. In *ICML*, pp. 7645–7657, 2023. URL `https://proceedings.mlr.press/v202/deng23d.html`.

Sjoerd Dirksen, Martin Genzel, Laurent Jacques, and Alexander Stollenwerk. The separation capacity of random neural networks. *Journal of Machine Learning Research*, 23(309):1–47, 2022.

Tolga Ergen and Mert Pilanci. Path Regularization: A Convexity and Sparsity Inducing Regularization for Parallel ReLU Networks.

Tolga Ergen and Mert Pilanci. Convex Geometry and Duality of Over-parameterized Neural Networks, August 2021a. URL http://arxiv.org/abs/2002.11219. arXiv:2002.11219 [cs].

Tolga Ergen and Mert Pilanci. Implicit Convex Regularizers of CNN Architectures: Convex Optimization of Two- and Three-Layer Networks in Polynomial Time, March 2021b. URL http://arxiv.org/abs/2006.14798. arXiv:2006.14798 [cs].

Tolga Ergen and Mert Pilanci. The Convex Landscape of Neural Networks: Characterizing Global Optima and Stationary Points via Lasso Models. *IEEE Transactions on Information Theory*, 71(5):3854–3870, May 2025. ISSN 1557-9654. doi: 10.1109/TIT.2025.3545564. URL https://ieeexplore.ieee.org/document/10902486/.

Tolga Ergen, Behnam Neyshabur, and Harsh Mehta. Convexifying Transformers: Improving optimization and understanding of transformer networks, November 2022. URL http://arxiv.org/abs/2211.11052. arXiv:2211.11052 [cs].

Tolga Ergen, Halil Ibrahim Gulluk, Jonathan Lacotte, and Mert Pilanci. Globally Optimal Training of Neural Networks with Threshold Activation Functions, March 2023. URL http://arxiv.org/abs/2303.03382.

Jason K. Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D. Lu. Training spiking neural networks using lessons from deep learning, 2023. URL https://arxiv.org/abs/2109.12894.

Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

Samy Jelassi, Stéphane d'Ascoli, Carles Domingo-Enrich, Yuhuai Wu, Yuanzhi Li, and François Charton. Length generalization in arithmetic transformers. *arXiv preprint arXiv:2306.15400*, 2023.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Nayoung Lee, Kartik Sreenivasan, Jason D Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers. *arXiv preprint arXiv:2307.03381*, 2023.

Yang Li and Yi Zeng. Efficient and accurate conversion of spiking neural network with burst spikes, 2022. URL https://arxiv.org/abs/2204.13271.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, et al. Transformers can do arithmetic with the right embeddings. *Advances in Neural Information Processing Systems*, 37:108012–108041, 2024.

A. Mehonic and A. J. Kenyon. Brain-inspired computing needs a master plan. *Nature*, 604(7905):255–260, April 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04362-w. URL http://dx.doi.org/10.1038/s41586-021-04362-w.

Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *CoRR*, abs/1901.09948, 2019. URL http://arxiv.org/abs/1901.09948.

Behnam Neyshabur, Yuhuai Wu, Russ R Salakhutdinov, and Nati Srebro. Path-Normalized Optimization of Recurrent Neural Networks with ReLU Activations. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/74563ba21a90da13dacf2a73e3ddefa7-Abstract.html.

Michael Pfeiffer and Thomas Pfeil. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, Volume 12 - 2018, 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00774. URL https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2018.00774.

Nitin Rathi and Kaushik Roy. Diet-snn: Direct input encoding with leakage and threshold optimization in deep spiking neural networks, 2020. URL https://arxiv.org/abs/2008.03658.

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), 01 2022. ISSN ISSN 2662-8457. doi: 10.1038/s43588-021-00184-y. URL https://www.osti.gov/biblio/1881146.

Ryota Tomioka and Nathan Srebro. Norm-Based Capacity Control in Neural Networks.

Roman Vershynin. Memory capacity of neural networks with threshold and rectified linear unit activations. *SIAM Journal on Mathematics of Data Science*, 2(4):1004–1033, 2020.

Yifei Wang and Mert Pilanci. The Convex Geometry of Backpropagation: Neural Network Gradient Flows Converge to Extreme Points of the Dual Convex Program, October 2021. URL http://arxiv.org/abs/2110.06488.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12, May 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00331. URL http://dx.doi.org/10.3389/fnins.2018.00331.

Appendix

A Feedforward Threshold Networks

We first prove the reduction of the weight-decay regularizer defined in §3.1 to its last-layer norms for a single network before proving Theorem 3.1.

A.1 Proof of Reduction to outer Norms

Theorem (Reduction to outer norms)**.** *Let $G(V, E, w)$ be a fully connected threshold network with amplitudes $\{s_v\}$ and weight-decay regularizer*

$$\mu_{p,2}(\Theta)^2 = \sum_{v \in V \setminus V_{\text{in}}} \Big(|s_v|^p + \sum_{(u,v) \in E} |w(u,v)|^p \Big)^{2/p}.$$

Then

$$\min_{\Theta} \mathcal{L}(f^{L,\Theta}(X), Y) + \frac{\beta_{\text{reg}}}{2} \mu_{p,2}(\Theta)^2 = \min_{\substack{\Theta \\ |s_{V_{\text{out}}}|=1}} \mathcal{L}(f^{L,\Theta}(X), Y) + \beta_{\text{reg}} \sum_{(u,V_{\text{out}}) \in E} \|W(u, V_{\text{out}})\|_p.$$

Proof. For a hidden node v, let $a_v = \big(\sum_{(u,v)\in E} |w(u,v)|^p\big)^{1/p}$ and $z_v = \sum_{(u,v)\in E} w(u,v)o(u)$, so $o(v) = \sigma_{s_v}(z_v) = s_v \mathbb{I}\{z_v \geq 0\}$. For $\alpha > 0$,

$$\sigma_{s_v}(\alpha z_v) = s_v \mathbb{I}\{\alpha z_v \geq 0\} = s_v \mathbb{I}\{z_v \geq 0\} = \sigma_{s_v}(z_v),$$

hence $f^{L,\Theta}(X)$ is invariant to a_v for every hidden v and to s_v for every $v \notin V_{\text{out}}$. The output retains only $s_{V_{\text{out}}} W(\cdot, V_{\text{out}})$. By this invariance we may push

$$a_v \to 0 \;\; (v \text{ hidden}), \qquad |s_v| \to 0 \;\; (v \notin V_{\text{out}}),$$

leaving

$$\min_{\Theta} \mathcal{L}(f^{L,\Theta}(X), Y) + \frac{\beta_{\text{reg}}}{2} \Big(|s_{V_{\text{out}}}|^2 + \|W(\cdot, V_{\text{out}})\|_p^2 \Big).$$

With $s_{V_{\text{out}}} \mapsto \alpha s_{V_{\text{out}}}$, $W(\cdot, V_{\text{out}}) \mapsto \alpha^{-1} W(\cdot, V_{\text{out}})$ fixing the output,

$$\frac{\beta_{\text{reg}}}{2} \Big(\alpha^2 |s_{V_{\text{out}}}|^2 + \alpha^{-2} \|W(\cdot, V_{\text{out}})\|_p^2 \Big) \geq \beta_{\text{reg}} \, |s_{V_{\text{out}}}| \, \|W(\cdot, V_{\text{out}})\|_p,$$

with equality at $\alpha^2 = \|W(\cdot, V_{\text{out}})\|_p / |s_{V_{\text{out}}}|$. Setting $|s_{V_{\text{out}}}| = 1$,

$$\min_{\substack{\Theta \\ |s_{V_{\text{out}}}|=1}} \mathcal{L}(f^{L,\Theta}(X), Y) + \beta_{\text{reg}} \sum_{(u,V_{\text{out}}) \in E} \|W(u, V_{\text{out}})\|_p. \qquad \square$$

A.2 Proof of Theorem 3.1

Theorem (Reduction to outer norms for parallel networks)**.** *Let $G(V, E, w)$ be a fully connected K-parallel threshold network with subnetworks $\{G_k(V_k, E_k, w_k)\}_{k=1}^K$ and additive weight-decay regularizer*

$$\mu_{p,2}(\Theta)^2 = \sum_{k=1}^{K} \mu_{p,2}(\Theta_k)^2.$$

Then the weight-decay regularized training problem is equivalent, under the gauge $|s_k^{(L-1)}| = 1$ for all k, to

$$\min_{\Theta} \mathcal{L} + \tfrac{\beta_{\text{reg}}}{2} \mu_{p,2}(\Theta)^2 = \min_{\Theta:\, |s_k^{(L-1)}|=1} \mathcal{L} + \beta_{\text{reg}} \sum_{k=1}^{K} \|W_{L,k}\|_p,$$

with $f^{L,K,\Theta}(X) = f^{L,K,\bar{\Theta}}(X)$ throughout.

Proof. Since $\mu_{p,2}(\Theta)^2 = \sum_{k=1}^K \mu_{p,2}(\Theta_k)^2$ and the G_k have disjoint parameters, the previous theorem applies per k: pushing $\{\|W_{l,k}\|_p\}_{l\le L-1} \to 0$ and $\{|s_k^{(l)}|\}_{l\le L-2} \to 0$ leaves

$$\frac{\beta_{\text{reg}}}{2}\left(|s_k^{(L-1)}|^2 + \|W_{L,k}\|_p^2\right) \ge \beta_{\text{reg}}\,|s_k^{(L-1)}|\,\|W_{L,k}\|_p,$$

with $|s_k^{(L-1)}| = 1$ giving $\beta_{\text{reg}}\|W_{L,k}\|_p$. With $f^{L,K,\Theta} = \sum_{k=1}^K f^{L,k,\Theta_k}$,

$$\min_{\Theta}\ \mathcal{L}(f^{L,K,\Theta}(X), Y) + \frac{\beta_{\text{reg}}}{2}\mu_{p,2}(\Theta)^2 = \min_{\substack{\Theta \\ |s_k^{(L-1)}|=1}} \mathcal{L}(f^{L,K,\Theta}(X), Y) + \beta_{\text{reg}}\sum_{k=1}^K \|W_{L,k}\|_p.$$

□

B Proof of Theorem 3.2

We restate the theorem for convenience.

Theorem (Bidual of feedforward parallel threshold networks)**.** *Let $p^{L,K}$ denote the weight-decay regularized training problem for a K-parallel feedforward threshold network with convex loss $\mathcal{L}$ and regularization parameter $\beta_{\text{reg}} > 0$. Let $\phi_k(X;\theta_k)$ denote the final hidden-layer feature map of subnetwork k after the reduction of Theorem 3.1. Then the dual problem is*

$$d^{L,K} = \max_{\lambda}\ -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\theta_k\in\bar{\Theta}_k} \|\lambda^\top \phi_k(X;\theta_k)\|_{p^*} \le \beta_{\text{reg}}, \qquad k \in [K].$$

The corresponding infinite bidual is

$$p_\infty^{L,K} = \min_{\mu}\left\{\mathcal{L}\left(\int_{\theta\in\bar{\Theta}} \phi(X;\theta)\,d\mu(\theta), Y\right) + \beta_{\text{reg}}\|\mu\|_{\text{TV}}\right\}.$$

Since $\beta_{\text{reg}} > 0$, strong duality holds:

$$d^{L,K} = p_\infty^{L,K}.$$

Moreover, if the number of parallel subnetworks satisfies $K \ge K^*$ for some $K^* \le n+1$, then

$$p^{L,K} = d^{L,K} = p_\infty^{L,K}.$$

For vector-valued outputs with k output coordinates, the Carathéodory bound becomes $K^* \le nk+1$ after flattening the output matrix into $\mathbb{R}^{nk}$.

Proof. After applying Theorem 3.1, the reduced training problem is

$$p^{L,K} = \min_{\{\theta_k,w_k\}_{k=1}^K} \mathcal{L}\left(\sum_{k=1}^K \phi_k(X;\theta_k)w_k, Y\right) + \beta_{\text{reg}}\sum_{k=1}^K \|w_k\|_p,$$

where $\theta_k \in \bar{\Theta}_k$ denotes the reduced hidden parameters of subnetwork k, and w_k denotes its output-layer weights.

Introduce an auxiliary variable

$$\hat{Y} = \sum_{k=1}^K \phi_k(X;\theta_k)w_k.$$

The equivalent constrained problem is

$$\min_{\hat{Y},\{\theta_k,w_k\}} \mathcal{L}(\hat{Y}, Y) + \beta_{\text{reg}}\sum_{k=1}^K \|w_k\|_p \quad \text{s.t.} \quad \hat{Y} = \sum_{k=1}^K \phi_k(X;\theta_k)w_k.$$

Let λ be the Lagrange multiplier for this equality constraint. The Lagrangian is

$$\mathcal{L}(\hat{Y}, Y) + \lambda^\top \hat{Y} - \lambda^\top \sum_{k=1}^{K} \phi_k(X;\theta_k) w_k + \beta_{\text{reg}} \sum_{k=1}^{K} \|w_k\|_p.$$

Taking the infimum over $\hat{Y}$ gives

$$\inf_{\hat{Y}} \left\{ \mathcal{L}(\hat{Y}, Y) + \lambda^\top \hat{Y} \right\} = -\mathcal{L}^*(-\lambda).$$

For each k, the infimum over w_k is

$$\inf_{w_k} \left\{ -\lambda^\top \phi_k(X;\theta_k) w_k + \beta_{\text{reg}} \|w_k\|_p \right\}.$$

By the conjugacy of the ℓ_p norm, this infimum is finite and equal to zero if and only if

$$\|\lambda^\top \phi_k(X;\theta_k)\|_{p^*} \le \beta_{\text{reg}};$$

otherwise it is $-\infty$. Since each subnetwork has its own hidden parameters, the dual must satisfy this constraint uniformly over all admissible hidden parameters. Therefore, the dual is

$$d^{L,K} = \max_{\lambda} \; -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\theta_k \in \bar{\Theta}_k} \|\lambda^\top \phi_k(X;\theta_k)\|_{p^*} \le \beta_{\text{reg}}, \qquad k \in [K].$$

If all subnetworks share the same admissible hidden-parameter class, this is equivalently a single uniform constraint over that class.

We next derive the bidual. Define the reduced atom set

$$\mathcal{A} = \left\{ \phi(X;\theta) a : \theta \in \bar{\Theta},\ \|a\|_p \le 1 \right\} \subseteq \mathbb{R}^n$$

for scalar outputs. For vector-valued outputs, we regard the output matrix as a vector in $\mathbb{R}^{nk}$ and define $\mathcal{A}$ analogously after vectorization.

The dual constraint can be written as

$$\sup_{a \in \mathcal{A}} \lambda^\top a \le \beta_{\text{reg}}.$$

The associated atomic gauge is

$$\gamma_{\mathcal{A}}(z) = \inf_{\mu} \left\{ \|\mu\|_{\text{TV}} : z = \int_{a \in \mathcal{A}} a \, d\mu(a) \right\}.$$

The bidual is therefore

$$p_\infty^{L,K} = \min_{z} \mathcal{L}(z, Y) + \beta_{\text{reg}} \gamma_{\mathcal{A}}(z),$$

or equivalently,

$$p_\infty^{L,K} = \min_{\mu} \left\{ \mathcal{L}\left(\int_{a \in \mathcal{A}} a \, d\mu(a), Y \right) + \beta_{\text{reg}} \|\mu\|_{\text{TV}} \right\}.$$

Reparameterizing the atoms $a = \phi(X;\theta)c$ gives the stated form over reduced hidden parameters.

Strong duality follows from Slater's condition. At $\lambda = 0$,

$$\max_{\theta \in \bar{\Theta}} \|0^\top \phi(X;\theta)\|_{p^*} = 0 < \beta_{\text{reg}},$$

since $\beta_{\text{reg}} > 0$. Hence the dual feasible set has nonempty relative interior, and

$$d^{L,K} = p_\infty^{L,K}.$$

Finally, the finite-width equivalence follows from Carathéodory's theorem. For scalar outputs, the atom set lies in $\mathbb{R}^n$, so any point in the convex hull of the atom set can be represented using at most $n+1$ atoms. Each atom corresponds to one parallel subnetwork. Therefore, if

$$K \geq K^* \qquad \text{for some} \qquad K^* \leq n+1,$$

the finite K-parallel primal can realize the same prediction vector as the infinite bidual. Hence

$$p^{L,K} = p_\infty^{L,K}.$$

Combining this with strong duality gives

$$p^{L,K} = d^{L,K} = p_\infty^{L,K}.$$

For vector-valued outputs with k output dimensions, the same argument applies in $\mathbb{R}^{nk}$, giving the bound $K^* \leq nk+1$. □

C Recurrent Threshold Network Theory

C.1 Unrolled DAG representation and recurrent reduction

A fixed-horizon recurrent threshold network is represented as a directed acyclic graph by unrolling over timesteps $t = 1, \ldots, T$: each recurrent unit is copied once per timestep, a hidden unit v becomes a time-indexed node v^t, and tied edges satisfy

$$w(u^t, v^t) = w(u^{t'}, v^{t'}) \qquad \text{for all } t, t' \in [T].$$

The difference from the feedforward case is that each recurrent hidden neuron has two incoming blocks, $P_{k,l,\text{in}}[:,i]$ and $P_{k,l,\text{rec}}[:,i]$, tied across time; the reduction scales them jointly. For $\alpha > 0$, set $P_{k,l,\text{in}}[:,i] \mapsto \alpha P_{k,l,\text{in}}[:,i]$ and $P_{k,l,\text{rec}}[:,i] \mapsto \alpha P_{k,l,\text{rec}}[:,i]$. The pre-activation

$$z_{k,l,i}^t = H_{k,l-1}^t P_{k,l,\text{in}}[:,i] + H_{k,l}^{t-1} P_{k,l,\text{rec}}[:,i]$$

obeys $z_{k,l,i}^t \mapsto \alpha z_{k,l,i}^t$, and

$$\sigma(\alpha\, z_{k,l,i}^t) = \sigma(z_{k,l,i}^t).$$

The spike output is unchanged, so every downstream computation is unchanged and $f^{L,T,K,\Theta}$ is invariant to α. Writing $a_{k,l,i} = \left(\|P_{k,l,\text{in}}[:,i]\|_p^p + \|P_{k,l,\text{rec}}[:,i]\|_p^p\right)^{1/p}$, we may push

$$\inf_{\alpha>0} \frac{\beta_{\text{reg}}}{2}\, \alpha^2 a_{k,l,i}^2 = 0$$

for every inner neuron ($l \leq L-1$), and likewise $|s_k^{(l)}| \to 0$ for $l \leq L-2$. The pair $(s_k^{(L-1)}, P_{k,\text{out}})$ merges by the arithmetic-geometric mean inequality under $|s_k^{(L-1)}| = 1$ as in Theorem A.1, giving

$$p^{L,T,K} = \min_{\substack{\Theta \\ |s_k^{(L-1)}|=1}} \mathcal{L}(f^{L,T,K,\Theta}(X^{1:T}), Y) + \beta_{\text{reg}} \sum_{k=1}^{K} \|P_{k,\text{out}}\|_p.$$

C.2 Recurrent bidual theorem

Theorem C.1 (Bidual of recurrent parallel threshold networks)**.** *Let $p^{L,T,K}$ denote the reduced weight-decay regularized recurrent training problem with convex loss $\mathcal{L}$ and regularization parameter $\beta_{\text{reg}} > 0$. Let $\phi_{T,k}(X^{1:T};\theta_k)$ denote the final hidden-layer feature map of subnetwork k at timestep T after the recurrent reduction. Then the dual problem is*

$$d^{L,T,K} = \max_{\lambda}\ -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\theta_k \in \bar{\Theta}_k} \|\lambda^\top \phi_{T,k}(X^{1:T};\theta_k)\|_{p^*} \leq \beta_{\text{reg}}, \qquad k \in [K].$$

The corresponding infinite bidual is

$$p_\infty^{L,T,K} = \min_\mu \left\{ \mathcal{L}\left(\int_{\theta \in \bar{\Theta}} \phi_T(X^{1:T};\theta)\, d\mu(\theta), Y \right) + \beta_{\text{reg}} \|\mu\|_{\text{TV}} \right\}.$$

Since $\beta_{\text{reg}} > 0$*, strong duality holds:*

$$d^{L,T,K} = p_\infty^{L,T,K}.$$

Moreover, for scalar outputs, if $K \geq K^*$ *for some* $K^* \leq n+1$*, then*

$$p^{L,T,K} = d^{L,T,K} = p_\infty^{L,T,K}.$$

For d_{out} *output coordinates, the same argument gives* $K^* \leq n d_{\text{out}} + 1$ *after vectorizing the predictions.*

Proof. Using Appendix C, the reduced recurrent training problem is

$$p^{L,T,K} = \min_{\{\theta_k, w_k\}_{k=1}^K} \mathcal{L}\left(\sum_{k=1}^K \phi_{T,k}(X^{1:T};\theta_k) w_k, Y \right) + \beta_{\text{reg}} \sum_{k=1}^K \|w_k\|_p,$$

where $\theta_k \in \bar{\Theta}_k$ denotes the reduced hidden recurrent parameters and $w_k = P_{k,\text{out}}$ denotes the output weights. Introduce an auxiliary prediction variable

$$\hat{Y} = \sum_{k=1}^K \phi_{T,k}(X^{1:T};\theta_k) w_k.$$

The constrained form is

$$\min_{\hat{Y}, \{\theta_k, w_k\}} \mathcal{L}(\hat{Y}, Y) + \beta_{\text{reg}} \sum_{k=1}^K \|w_k\|_p \quad \text{s.t.} \quad \hat{Y} = \sum_{k=1}^K \phi_{T,k}(X^{1:T};\theta_k) w_k.$$

Let λ be the Lagrange multiplier for this equality constraint. The Lagrangian is

$$\mathcal{L}(\hat{Y}, Y) + \lambda^\top \hat{Y} - \lambda^\top \sum_{k=1}^K \phi_{T,k}(X^{1:T};\theta_k) w_k + \beta_{\text{reg}} \sum_{k=1}^K \|w_k\|_p.$$

Taking the infimum over $\hat{Y}$ gives

$$\inf_{\hat{Y}} \left\{ \mathcal{L}(\hat{Y}, Y) + \lambda^\top \hat{Y} \right\} = -\mathcal{L}^*(-\lambda).$$

For each subnetwork k, the infimum over w_k is

$$\inf_{w_k} \left\{ -\lambda^\top \phi_{T,k}(X^{1:T};\theta_k) w_k + \beta_{\text{reg}} \|w_k\|_p \right\}.$$

By the conjugacy of the ℓ_p norm, this infimum is finite and equal to zero if and only if

$$\|\lambda^\top \phi_{T,k}(X^{1:T};\theta_k)\|_{p^*} \leq \beta_{\text{reg}}.$$

Otherwise, it is $-\infty$. Since the hidden parameters are optimized, the constraint must hold uniformly over the reduced recurrent parameter set. Therefore, the dual is

$$d^{L,T,K} = \max_\lambda \; -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\theta_k \in \bar{\Theta}_k} \|\lambda^\top \phi_{T,k}(X^{1:T};\theta_k)\|_{p^*} \leq \beta_{\text{reg}}, \qquad k \in [K].$$

Define the recurrent atom set

$$\mathcal{A}_{\text{RNN}} = \left\{ \phi_T(X^{1:T};\theta) a : \theta \in \bar{\Theta},\ \|a\|_p \leq 1 \right\} \subseteq \mathbb{R}^n$$

for scalar outputs. For vector-valued outputs, predictions are vectorized into $\mathbb{R}^{nd_{\text{out}}}$. The dual constraint is equivalently

$$\sup_{a \in \mathcal{A}_{\text{RNN}}} \lambda^\top a \leq \beta_{\text{reg}}.$$

The associated atomic gauge is

$$\gamma_{\mathcal{A}_{\text{RNN}}}(z) = \inf_{\mu} \left\{ \|\mu\|_{\text{TV}} : z = \int_{a \in \mathcal{A}_{\text{RNN}}} a \, d\mu(a) \right\}.$$

Thus, the bidual is

$$p_\infty^{L,T,K} = \min_z \mathcal{L}(z, Y) + \beta_{\text{reg}} \gamma_{\mathcal{A}_{\text{RNN}}}(z),$$

or equivalently,

$$p_\infty^{L,T,K} = \min_{\mu} \left\{ \mathcal{L}\left(\int_{a \in \mathcal{A}_{\text{RNN}}} a \, d\mu(a), Y \right) + \beta_{\text{reg}} \|\mu\|_{\text{TV}} \right\}.$$

Reparameterizing atoms as $a = \phi_T(X^{1:T}; \theta)c$ gives the stated measure-valued form over recurrent hidden parameters.

Strong duality follows from Slater's condition. At $\lambda = 0$,

$$\max_{\theta \in \bar{\Theta}} \|0^\top \phi_T(X^{1:T}; \theta)\|_{p^*} = 0 < \beta_{\text{reg}},$$

because $\beta_{\text{reg}} > 0$. Hence the dual feasible set has nonempty relative interior, and

$$d^{L,T,K} = p_\infty^{L,T,K}.$$

Finally, since L and T are fixed and the training set has n samples, every recurrent threshold activation pattern lies in $\{0,1\}^n$. Therefore, the recurrent feature generator is finite, and hence bounded, on the training set. For scalar outputs, the atom set lies in $\mathbb{R}^n$, so Carathéodory's theorem implies that any point in the convex hull of the atom set can be represented using at most $n+1$ atoms. Each atom is realizable by one parallel recurrent subnetwork. Hence, if $K \geq K^*$ for some $K^* \leq n+1$, the finite K-parallel recurrent network can realize the same prediction vector as the infinite bidual. Therefore,

$$p^{L,T,K} = p_\infty^{L,T,K}.$$

Combining with strong duality gives

$$p^{L,T,K} = d^{L,T,K} = p_\infty^{L,T,K}.$$

For d_{out} output coordinates, the atom set lies in $\mathbb{R}^{nd_{\text{out}}}$ after vectorization, giving the bound $K^* \leq nd_{\text{out}}+1$. □

C.3 Witnessed recurrent arrangement construction

Let $D^{l,t}$ denote the arrangement dictionary at layer l and timestep t. In the recurrent setting, this dictionary is witnessed: each activation column is stored together with the shared recurrent weights that generated the full trajectory. For layer l, the witness is

$$\omega_l = (P_{l,\text{in}}, P_{l,\text{rec}}),$$

which is chosen once and reused across all timesteps. Thus, $D^{l,t}$ is not an unconstrained marginal set of activation patterns at (l,t); it is a trajectory-consistent dictionary generated by shared recurrent weights.

The recurrent pre-activation can be written as a feedforward-style hyperplane over a stacked feature matrix:

$$H_l^t P_{l+1,\text{in}} + H_{l+1}^{t-1} P_{l+1,\text{rec}} = \begin{bmatrix} H_l^t & H_{l+1}^{t-1} \end{bmatrix} \begin{bmatrix} P_{l+1,\text{in}} \\ P_{l+1,\text{rec}} \end{bmatrix}.$$

For the first recurrent layer, the stacked feature matrix is

$$\begin{bmatrix} X^t & H_1^{t-1} \end{bmatrix}.$$

Therefore, the feedforward arrangement operator $A(\cdot)$ extends to recurrent networks by applying it to these stacked feature matrices, while restricting recurrent columns to those generated by the same witness.

At $t = 1$, the witness constraint is vacuous because no previous recurrent trajectory has been generated. The base construction is

$$D^{1,1} = A\big(\begin{bmatrix} X^1 & H_1^0 \end{bmatrix}\big).$$

For $l \geq 2$ at the first timestep,

$$D^{l,1} = \bigsqcup_{|\mathcal{S}_F| = m_{l-1}} A\Big(\begin{bmatrix} D^{l-1,1}_{\mathcal{S}_F} & H_l^0 \end{bmatrix}\Big).$$

Here $\mathcal{S}_F$ ranges over column subsets of size m_{l-1}, and $\bigsqcup$ denotes union with duplicate activation columns removed.

For $t > 1$, the recurrent input cannot be sampled freely from the marginal set $D^{l,t-1}$. It must come from the same shared-weight trajectory that generated the current forward column. Thus, for the first recurrent layer,

$$D^{1,t} = \bigsqcup_{\omega} A\big(\begin{bmatrix} X^t & D^{1,t-1}_{\omega} \end{bmatrix}\big),$$

where $D^{1,t-1}_{\omega}$ denotes the recurrent activation columns at timestep $t-1$ generated by the same witness $\omega = (P_{1,\text{in}}, P_{1,\text{rec}})$. For deeper layers,

$$D^{l,t} = \bigsqcup_{\omega} A\big(\begin{bmatrix} D^{l-1,t}_{\omega} & D^{l,t-1}_{\omega} \end{bmatrix}\big).$$

The notation means that both the forward and recurrent columns are selected from dictionaries generated by the same shared-weight witness.

This witnessed construction enforces recurrent parameter sharing at the dictionary level. If recurrent columns were sampled freely from the marginal set $D^{l,t-1}$, the construction could combine activation patterns generated by incompatible recurrent parameters. Such combinations may not be realizable by any single recurrent network.

At every layer–timestep pair (l, t), each activation column is a binary vector over the n training samples. Hence

$$D^{l,t} \subseteq \{0,1\}^n, \qquad |D^{l,t}| \leq 2^n.$$

Since L and T are fixed, the full witnessed collection

$$\{D^{l,t}\}_{l \in [L-1],\, t \in [T]}$$

is finite. The witness constraint can only reduce the admissible set of columns; it cannot increase the cardinality beyond 2^n at any fixed (l, t).

C.4 Proof of Theorem 3.4

Proof. By Theorem C.1, the recurrent dual is

$$d^{L,T} = \max_{\lambda} \; -\mathcal{L}^*(-\lambda) \quad \text{s.t.} \quad \max_{\Theta \in \bar{\Theta}_{\text{RNN}}} \big\| \lambda^\top \sigma\big(H_{L-2}^T P_{L-1,\text{in}} + H_{L-1}^{T-1} P_{L-1,\text{rec}}\big) \big\|_2 \leq \beta_{\text{reg}}.$$

The witnessed arrangement construction in Appendix C.3 ensures that every column of the final hidden-layer activation at $(L-1, T)$ is a column of

$$D^{L-1,T} = [d_1, \ldots, d_{P^{L-1,T}}].$$

Thus, for a width-m_{L-1} final hidden layer,

$$\left\|\lambda^\top \sigma\big(H_{L-2}^T P_{L-1,\text{in}} + H_{L-1}^{T-1} P_{L-1,\text{rec}}\big)\right\|_2 = \left(\sum_{j=1}^{m_{L-1}} (\lambda^\top d_j)^2\right)^{1/2},$$

where each d_j is a column of $D^{L-1,T}$. Since the final-layer hidden neurons are independently parameterized, maximizing over the hidden parameters gives

$$\max_{d_i \in D^{L-1,T}} \sqrt{m_{L-1}}|\lambda^\top d_i| \le \beta_{\text{reg}}.$$

This constraint is equivalent to the two-sided inequalities

$$\sqrt{m_{L-1}}\lambda^\top d_i - \beta_{\text{reg}} \le 0, \qquad -\sqrt{m_{L-1}}\lambda^\top d_i - \beta_{\text{reg}} \le 0, \qquad i \in [P^{L-1,T}].$$

At $\lambda = 0$, all inequalities are strict because $\beta_{\text{reg}} > 0$, so Slater's condition holds.

Introduce nonnegative Lagrange multipliers γ_i^+ and γ_i^- for the two inequalities. The Lagrangian of the dual problem is

$$-\mathcal{L}^*(-\lambda) - \sum_i \gamma_i^+(\sqrt{m_{L-1}}\lambda^\top d_i - \beta_{\text{reg}}) - \sum_i \gamma_i^-(-\sqrt{m_{L-1}}\lambda^\top d_i - \beta_{\text{reg}}).$$

Collecting terms gives

$$-\mathcal{L}^*(-\lambda) - \lambda^\top \sqrt{m_{L-1}} \sum_i (\gamma_i^+ - \gamma_i^-) d_i + \beta_{\text{reg}} \sum_i (\gamma_i^+ + \gamma_i^-).$$

Using the Fenchel identity with

$$v = -\sqrt{m_{L-1}} \sum_i (\gamma_i^+ - \gamma_i^-) d_i,$$

we obtain

$$\min_{\gamma^+,\gamma^- \ge 0} \mathcal{L}\left(-\sqrt{m_{L-1}} \sum_i (\gamma_i^+ - \gamma_i^-) d_i, Y\right) + \beta_{\text{reg}} \sum_i (\gamma_i^+ + \gamma_i^-).$$

Let $w_i = \gamma_i^+ - \gamma_i^-$. Minimizing over nonnegative γ_i^+, γ_i^- with fixed difference w_i gives

$$\gamma_i^+ + \gamma_i^- = |w_i|.$$

Therefore,

$$d^{L,T} = \min_{w \in \mathbb{R}^{P^{L-1,T}}} \mathcal{L}(-\sqrt{m_{L-1}} D^{L-1,T} w, Y) + \beta_{\text{reg}} \|w\|_1.$$

Finally, set $\widetilde{w} = -\sqrt{m_{L-1}} w$. Then

$$-\sqrt{m_{L-1}} D^{L-1,T} w = D^{L-1,T} \widetilde{w}, \qquad \|w\|_1 = \frac{1}{\sqrt{m_{L-1}}} \|\widetilde{w}\|_1.$$

Thus,

$$\widetilde{p}^{L,T} = \min_{\widetilde{w} \in \mathbb{R}^{P^{L-1,T}}} \mathcal{L}(D^{L-1,T} \widetilde{w}, Y) + \frac{\beta_{\text{reg}}}{\sqrt{m_{L-1}}} \|\widetilde{w}\|_1.$$

The equivalence to the original finite-width recurrent training problem follows from Theorem C.1 whenever $K \ge K^*$. □

C.5 Recurrent arrangement enumeration cost

The exact recurrent convex formulation is dominated by the construction of the final witnessed arrangement matrix $D^{L-1,T}$. We first bound the cost of a single call to the arrangement operator.

For an input matrix $Z \in \mathbb{R}^{n\times p}$ with rank $r_Z = \mathrm{rank}(Z) \leq \min(n,p)$, the operator $A(Z)$ enumerates sign patterns

$$\{\mathbb{I}\{Zu \geq 0\} : u \in \mathbb{R}^p\}.$$

The rows of Z define n hyperplanes in parameter space. Standard hyperplane-arrangement and linear-threshold counting bounds imply that the number of realizable sign patterns is at most $O(n^{r_Z})$. If each pattern is evaluated on all n samples in $O(np)$ time, then

$$\mathrm{cost}(A(Z)) = O(n^{r_Z+1}p).$$

At recurrent cell (l,t), the construction combines forward features from $(l-1,t)$ and recurrent features from $(l,t-1)$. Let $Q^{l,t}$ denote the number of spiked dictionary entries available at (l,t) before deduplicating identical binary columns, and define

$$r^{l,t} = \min(n, m_{l-1} + m_l).$$

The number of witnessed subset pairs is bounded by

$$\binom{Q^{l-1,t}}{m_{l-1}}\binom{Q^{l,t-1}}{m_l}.$$

For each subset pair, $A(\cdot)$ is applied to an $n \times (m_{l-1} + m_l)$ stacked feature matrix. Therefore,

$$\mathrm{cost}(D^{l,t}) = O\left(\binom{Q^{l-1,t}}{m_{l-1}}\binom{Q^{l,t-1}}{m_l} n^{r^{l,t}+1}(m_{l-1}+m_l)\right).$$

After deduplication, the binary arrangement matrix satisfies $|D^{l,t}| \leq 2^n$.

Let

$$m^* = \max_{l\in[L-1]} m_l, \qquad r^* = \min(n, 2m^*), \qquad Q^* = \max_{l,t} Q^{l,t}.$$

Bounding all LT recurrent cells by the worst-case cell gives

$$\mathrm{cost}(D^{L-1,T}) = O\left(LT\binom{Q^*}{m^*}^2 n^{r^*+1}m^*\right).$$

For a K-parallel network, the construction is performed independently for each subnetwork, so the cost scales linearly in K:

$$O\left(LTK\binom{Q^*}{m^*}^2 n^{r^*+1}m^*\right).$$

A simpler but looser bound is obtained by ignoring witness multiplicity and using $|D^{l,t}| \leq 2^n$:

$$O\left(LTK\binom{2^n}{m^*}^2 n^{r^*+1}m^*\right).$$

This bound is intentionally loose, but it captures the key point: exact witnessed arrangement enumeration is exponentially expensive in the number of training samples.

D LIF-SNN Theory

D.1 Typed LIF DAG and trainability settings

The fixed-horizon LIF-SNN can be represented as an unrolled DAG by introducing, for every layer l and timestep t, a membrane node U_l^t and a spike node S_l^t. The membrane node receives the three inputs appearing in the LIF recurrence:

$$U_l^t = S_{l-1}^t P_{l,\text{in}} + U_l^{t-1} B_l - S_l^{t-1} R_l, \qquad S_l^t = \sigma(U_l^t),$$

where $B_l = \text{Diag}(\beta_l)$, $\beta_l \in [0,1]^{m_l}$, and $R_l = \text{Diag}(U_{\text{thr}}^l)$. Thus, the corresponding edges are

$$S_{l-1}^t \to U_l^t \quad \text{with weight } P_{l,\text{in}}, \qquad U_l^{t-1} \to U_l^t \quad \text{with weight } B_l, \qquad S_l^{t-1} \to U_l^t \quad \text{with weight } -R_l.$$

The spike node is obtained by applying the threshold activation to the membrane node. Equivalently, the edge $U_l^t \to S_l^t$ has fixed weight 1, and the nonlinearity at S_l^t is $\sigma(\cdot)$.

We use two edge labels. The first label, $\tau : E \to \{0,1\}$, indicates whether an edge weight is optimized. Thus, $\tau(e) = 1$ denotes a trainable edge, while $\tau(e) = 0$ denotes a fixed structural edge. The second label, $\delta : E \to \{0,1\}$, indicates whether the edge is scale-bearing under the LIF membrane-rescaling symmetry. The input-to-membrane edges $P_{l,\text{in}}$ and reset-threshold edges $-R_l$ are scale-bearing, while the leak edges B_l are structural. The leak coefficients may be trainable or fixed depending on the configuration, but they do not carry the removable positive scale used in the weight-decay reduction.

In the structured-witness setting, at least one of β_l or U_{thr}^l may be optimized while generating the LIF witness. If U_{thr}^l is trainable, it is a scale-bearing parameter scaled jointly with the input block. If β_l is trainable, it remains a bounded structural recurrent coefficient and is not a scale-bearing trainable parameter. In the fixed-structural setting, both β_l and U_{thr}^l are fixed hyperparameters. They restrict the admissible LIF feature generator but are not optimized in the convex reconstruction stage.

D.2 Proof of Lemma 3.5

Proof. Let

$$A_l = \text{Diag}(a_{l,1}, \ldots, a_{l,m_l}), \qquad a_{l,i} > 0.$$

Define

$$\bar{U}_l^t = U_l^t A_l, \qquad \bar{P}_{l,\text{in}} = P_{l,\text{in}} A_l, \qquad \bar{R}_l = R_l A_l, \qquad \bar{B}_l = A_l^{-1} B_l A_l.$$

Starting from the LIF recurrence,

$$U_l^t = S_{l-1}^t P_{l,\text{in}} + U_l^{t-1} B_l - S_l^{t-1} R_l,$$

right-multiply both sides by A_l:

$$U_l^t A_l = S_{l-1}^t P_{l,\text{in}} A_l + U_l^{t-1} B_l A_l - S_l^{t-1} R_l A_l.$$

Using the definitions above,

$$U_l^t A_l = \bar{U}_l^t, \qquad P_{l,\text{in}} A_l = \bar{P}_{l,\text{in}}, \qquad R_l A_l = \bar{R}_l.$$

For the recurrent membrane term,

$$\bar{U}_l^{t-1} \bar{B}_l = U_l^{t-1} A_l (A_l^{-1} B_l A_l) = U_l^{t-1} B_l A_l.$$

Therefore,

$$\bar{U}_l^t = S_{l-1}^t \bar{P}_{l,\text{in}} + \bar{U}_l^{t-1} \bar{B}_l - S_l^{t-1} \bar{R}_l.$$

If B_l is diagonal, then A_l and B_l commute, so

$$\bar{B}_l = A_l^{-1} B_l A_l = B_l.$$

Finally, since A_l has strictly positive diagonal entries, each membrane coordinate is positively rescaled. The threshold activation is invariant under positive coordinate-wise scaling, so

$$\sigma(\bar{U}_l^t) = \sigma(U_l^t A_l) = \sigma(U_l^t).$$

Thus, positive diagonal rescaling of the membrane coordinates leaves all spike outputs unchanged. □

D.3 Reduction of the LIF weight-decay regularizer

The trainable parameters of a hidden LIF neuron i in layer l are $P_{l,\text{in}}[:,i]$ and, when trainable, $U^l_{\text{thr},i}$; both act only through U^t_l, which feeds σ. The leak B_l is structural (Lemma 3.5). For $A_l \succ 0$,

$$\sigma(U^t_l A_l) = \sigma(U^t_l),$$

so these magnitudes and the inner amplitudes are invisible to f, and we may push

$$\|P_{l,\text{in}}[:,i]\|_p \to 0 \;\; (l \le L-1), \qquad |s^{(l)}_k| \to 0 \;\; (l \le L-2),$$

which sends their $\mu_{p,2}$ terms to 0 while leaving B_l untouched. The pair $(s^{(L-1)}_k, P_{k,\text{out}})$ merges by the arithmetic-geometric mean inequality under $|s^{(L-1)}_k| = 1$ as in Theorem A.1, giving

$$\min_{\Theta} \mathcal{L} + \frac{\beta_{\text{reg}}}{2}\mu_{p,2}(\Theta)^2 = \min_{\substack{\Theta \\ |s^{(L-1)}_k|=1}} \mathcal{L} + \beta_{\text{reg}} \sum_{k=1}^{K} \|P_{k,\text{out}}\|_p.$$

D.4 Witnessed SNN arrangement construction and finiteness

The LIF-SNN reduction introduces a real-valued membrane state in addition to the binary spike state. Therefore, the SNN arrangement dictionary must track both spike patterns and the membrane values that generated them.

For a fixed LIF witness

$$\omega_l = (B_l, R_l, P_{l,\text{in}}, U^0_l),$$

the LIF recurrence unrolls as

$$U^t_l = U^0_l B^t_l + \sum_{\tau=1}^{t} \left(S^\tau_{l-1} P_{l,\text{in}} - S^{\tau-1}_l R_l\right) B^{t-\tau}_l.$$

Thus, under a fixed witness, U^t_l is a deterministic function of the previous-layer spike trajectory

$$(S^1_{l-1}, \ldots, S^t_{l-1})$$

and the same-layer recurrent spike trajectory

$$(S^0_l, \ldots, S^{t-1}_l).$$

Lemma D.1 (Finiteness of realizable membrane states for a fixed witness)**.** *Fix a finite training set of n samples, horizon T, and LIF parameter witness $\omega_l = (B_l, R_l, P_{l,\text{in}}, U^0_l)$. Then the set of membrane states realizable at layer–timestep pair (l,t) from a finite collection of spike trajectories is finite. More precisely,*

$$\left|\{U^t_l \text{ realizable under } \omega_l\}\right| \le \prod_{\tau=1}^{t} |D^{l-1,\tau}_{\text{SNN}}| \cdot \prod_{\tau=0}^{t-1} |D^{l,\tau}_{\text{SNN}}|.$$

Proof. The tuple

$$(S^1_{l-1}, \ldots, S^t_{l-1}, S^0_l, \ldots, S^{t-1}_l)$$

ranges over a finite Cartesian product of spike-arrangement sets. For a fixed witness ω_l, the unrolled recurrence maps each such tuple deterministically to a membrane state U^t_l. Therefore, the number of realizable membrane states is bounded by the product above. □

Across different LIF witnesses, the real-valued membrane values may vary. However, the projected spike dictionary remains finite after deduplication, because every spike column lies in $\{0,1\}^n$. Hence

$$D^{l,t}_{\text{SNN}} \subseteq \{0,1\}^n, \qquad |D^{l,t}_{\text{SNN}}| \le 2^n.$$

We now define the witnessed SNN arrangement operator. At layer–timestep pair (l, t), the SNN pre-activation can be written in stacked form:

$$U_l^t = \begin{bmatrix} S_{l-1}^t & U_l^{t-1} & S_l^{t-1} \end{bmatrix} \begin{bmatrix} P_{l,\text{in}} \\ B_l \\ -R_l \end{bmatrix}.$$

The structured arrangement operator $A_{\text{SNN}}(\cdot)$ applies the usual hyperplane-arrangement construction to the stacked state

$$\begin{bmatrix} S_{l-1}^t & U_l^{t-1} & S_l^{t-1} \end{bmatrix},$$

but only over LIF-structured parameter blocks

$$\begin{bmatrix} P_{l,\text{in}} \\ B_l \\ -R_l \end{bmatrix},$$

where B_l and R_l are diagonal and B_l has entries in $[0, 1]$.

At each cell (l, t), the forward input is drawn from the spike dictionary $D_{\text{SNN}}^{l-1,t}$. The recurrent input is drawn from the witnessed joint dictionary at $(l, t-1)$, whose entries are membrane–spike pairs

$$(U_l^{t-1}, S_l^{t-1})$$

generated by the same trajectory witness. Therefore, the recurrent subset is not an arbitrary Cartesian product of membrane and spike columns; choosing a recurrent spike column automatically chooses the membrane column associated with it under the same witness.

With this convention, the recursive construction is

$$D_{\text{SNN}}^{l,t} := \bigsqcup_{\substack{|\mathcal{S}_F|=m_{l-1},\ |\mathcal{S}_R|=m_l \\ \text{witness-consistent}}} A_{\text{SNN}}\left(\begin{bmatrix} D_{\text{SNN},\mathcal{S}_F}^{l-1,t} & U_{l,\mathcal{S}_R}^{t-1} & D_{\text{SNN},\mathcal{S}_R}^{l,t-1} \end{bmatrix}\right),$$

where $U_{l,\mathcal{S}_R}^{t-1}$ denotes the membrane columns associated with the selected recurrent spike columns under the same trajectory witness. The union $\bigsqcup$ removes duplicate spike columns after projection.

The construction is well-founded by diagonal induction on $l + t$, since each $D_{\text{SNN}}^{l,t}$ depends only on the previous-layer dictionary $D_{\text{SNN}}^{l-1,t}$ and the same-layer witnessed recurrent dictionary at $D_{\text{SNN}}^{l,t-1}$. Because every projected spike column lies in $\{0, 1\}^n$, the complete projected SNN dictionary is finite for fixed L and T.

D.5 Proof of Per-Timestep Loss

We prove the finite convex formulation for per-timestep supervision with a shared output layer. The reduced training problem is

$$p_{\text{all-}t}^{L,T,K} = \min_{\Theta \in \bar{\Theta}_{\text{SNN}}} \sum_{t=1}^{T} \mathcal{L}\left(\sum_{k=1}^{K} \sigma(U_{L-1,k}^t) P_{k,\text{out}}, Y^t\right) + \beta_{\text{reg}} \sum_{k=1}^{K} \|P_{k,\text{out}}\|_2.$$

The proof follows the finite-convex derivation of Theorem 3.4, with two changes. First, the dual variable has one block $\lambda_t \in \mathbb{R}^n$ per timestep. Second, the shared output layer couples all timestep losses through a single coefficient vector.

Introduce auxiliary variables

$$\widehat{Y}^t = \sum_{k=1}^{K} \sigma(U_{L-1,k}^t) P_{k,\text{out}}, \qquad t \in [T],$$

with multipliers $\lambda_t \in \mathbb{R}^n$. For each subnetwork k, the terms involving $P_{k,\text{out}}$ are

$$-\left(\sum_{t=1}^{T} \lambda_t^\top \sigma(U_{L-1,k}^t)\right) P_{k,\text{out}} + \beta_{\text{reg}} \|P_{k,\text{out}}\|_2.$$

Taking the infimum over $P_{k,\text{out}}$ gives zero if

$$\left\| \sum_{t=1}^{T} \lambda_t^\top \sigma(U_{L-1,k}^t) \right\|_2 \leq \beta_{\text{reg}},$$

and gives $-\infty$ otherwise. Hence the dual constraint is

$$\max_{\Theta \in \bar{\Theta}_{\text{SNN}}} \left\| \sum_{t=1}^{T} \lambda_t^\top \sigma(U_{L-1,k}^t) \right\|_2 \leq \beta_{\text{reg}}.$$

Under shared recurrent parameters, the columns selected across timesteps must form a trajectory-consistent tuple. Let $\mathcal{T}_{L-1}$ denote the set of trajectory-consistent final-layer tuples, and write

$$D^{\text{traj}} = \begin{bmatrix} D_{\mathcal{T}}^{L-1,1} \\ \vdots \\ D_{\mathcal{T}}^{L-1,T} \end{bmatrix} \in \{0,1\}^{nT \times |\mathcal{T}_{L-1}|}.$$

Then the dual constraint reduces to

$$\max_{i \in [|\mathcal{T}_{L-1}|]} \sqrt{m_{L-1}} \left| \sum_{t=1}^{T} \lambda_t^\top D_{\mathcal{T}}^{L-1,t}[:,i] \right| \leq \beta_{\text{reg}}.$$

Equivalently, for each i,

$$\sqrt{m_{L-1}} \sum_{t=1}^{T} \lambda_t^\top D_{\mathcal{T}}^{L-1,t}[:,i] - \beta_{\text{reg}} \leq 0,$$

and

$$-\sqrt{m_{L-1}} \sum_{t=1}^{T} \lambda_t^\top D_{\mathcal{T}}^{L-1,t}[:,i] - \beta_{\text{reg}} \leq 0.$$

At $\lambda_1 = \cdots = \lambda_T = 0$, all inequalities are strict because $\beta_{\text{reg}} > 0$, so Slater's condition holds.

Introduce nonnegative multipliers γ_i^+ and γ_i^- for the two inequalities. The Lagrangian terms involving λ_t become

$$-\sum_{t=1}^{T} \mathcal{L}^*(-\lambda_t) - \sqrt{m_{L-1}} \sum_{t=1}^{T} \lambda_t^\top \sum_i (\gamma_i^+ - \gamma_i^-) D_{\mathcal{T}}^{L-1,t}[:,i] + \beta_{\text{reg}} \sum_i (\gamma_i^+ + \gamma_i^-).$$

Define

$$v_t = -\sqrt{m_{L-1}} D_{\mathcal{T}}^{L-1,t} (\gamma^+ - \gamma^-).$$

Using the Fenchel identity for each timestep gives the primal equivalent

$$\min_{\gamma^+, \gamma^- \geq 0} \sum_{t=1}^{T} \mathcal{L}\Big(-\sqrt{m_{L-1}} D_{\mathcal{T}}^{L-1,t} (\gamma^+ - \gamma^-), Y^t\Big) + \beta_{\text{reg}} \sum_i (\gamma_i^+ + \gamma_i^-).$$

Let $w = \gamma^+ - \gamma^-$. For fixed w, minimizing over nonnegative γ^+, γ^- gives

$$\sum_i (\gamma_i^+ + \gamma_i^-) = \|w\|_1.$$

Therefore,

$$\min_{w \in \mathbb{R}^{|\mathcal{T}_{L-1}|}} \sum_{t=1}^{T} \mathcal{L}\Big(-\sqrt{m_{L-1}} D_{\mathcal{T}}^{L-1,t} w, Y^t\Big) + \beta_{\text{reg}} \|w\|_1.$$

Finally, set

$$\widetilde{w} = -\sqrt{m_{L-1}}\, w.$$

Then

$$-\sqrt{m_{L-1}}D_{\mathcal{T}}^{L-1,t}w = D_{\mathcal{T}}^{L-1,t}\widetilde{w}, \qquad \|w\|_1 = \frac{1}{\sqrt{m_{L-1}}}\|\widetilde{w}\|_1.$$

Thus,

$$\widetilde{p}_{\text{all-}t}^{L,T} = \min_{\widetilde{w}\in\mathbb{R}^{|\mathcal{T}_{L-1}|}} \sum_{t=1}^{T} \mathcal{L}(D_{\mathcal{T}}^{L-1,t}\widetilde{w}, Y^t) + \frac{\beta_{\text{reg}}}{\sqrt{m_{L-1}}}\|\widetilde{w}\|_1.$$

The generator now lies in $\mathbb{R}^{nT}$ because predictions are stacked across all T timesteps. Therefore, the Carathéodory bound becomes $K^* \leq nT + 1$ for scalar outputs.

D.6 Training-time cost for the SNN arrangement construction

The exact SNN convex formulation is dominated by the construction of the witnessed SNN dictionaries. The key difference from the generic recurrent threshold-network construction is that each SNN recurrent state contains both the membrane variable and the spike variable.

At cell (l, t), the SNN pre-activation uses the stacked feature matrix

$$\begin{bmatrix} S_{l-1}^t & U_l^{t-1} & S_l^{t-1} \end{bmatrix} \in \mathbb{R}^{n\times(m_{l-1}+2m_l)}.$$

Thus, one call to $A_{\text{SNN}}(\cdot)$ acts on an input matrix of width

$$p^{l,t} = m_{l-1} + 2m_l.$$

Let

$$r^{l,t} = \min(n, p^{l,t}).$$

By the same hyperplane-arrangement counting argument used for the recurrent dictionary, one call costs

$$O\Big(n^{r^{l,t}+1}(m_{l-1} + 2m_l)\Big).$$

Let $Q^{l,t}$ denote the number of spiked dictionary entries available at (l, t) before deduplicating identical spike columns. Although the stacked SNN state has three blocks, there are only two subset choices. The forward subset contains m_{l-1} spike columns from $D_{\text{SNN}}^{l-1,t}$. The recurrent subset contains m_l witnessed membrane–spike pairs from the same layer at timestep $t-1$. Choosing a recurrent spike subset automatically selects the associated membrane values under the same witness, so there is no third combinatorial factor for membrane subsets.

Hence, the number of subset choices is bounded by

$$\binom{Q^{l-1,t}}{m_{l-1}}\binom{Q^{l,t-1}}{m_l}.$$

For each subset pair, we apply $A_{\text{SNN}}(\cdot)$ to an $n \times (m_{l-1} + 2m_l)$ stacked feature matrix. Therefore, the cost of computing $D_{\text{SNN}}^{l,t}$ is

$$O\left(\binom{Q^{l-1,t}}{m_{l-1}}\binom{Q^{l,t-1}}{m_l} n^{r^{l,t}+1}(m_{l-1} + 2m_l)\right).$$

Let

$$m^* = \max_{l\in[L-1]} m_l, \qquad r^* = \min(n, 3m^*), \qquad Q^* = \max_{l,t} Q^{l,t}.$$

Bounding all LT cells by the worst-case cell gives

$$O\left(LT\binom{Q^*}{m^*}^2 n^{r^*+1}m^*\right).$$

Table 7: Training variants used in the experiments.

Method	Hidden dynamics	Output layer / training stage
SG	surrogate-gradient trained	surrogate-gradient trained
CVX	Gaussian sampled, frozen	convex reconstruction
SG-CVX	surrogate-pretrained, frozen	convex reconstruction
SG-SG	initialized from SG	surrogate-gradient finetuning
CVX-SG	initialized from CVX	surrogate-gradient finetuning

For a K-parallel SNN, the construction is performed independently for each subnetwork, so the cost scales linearly in K:

$$O\left(LTK\binom{Q^*}{m^*}^2 n^{r^*+1}m^*\right).$$

Ignoring witness multiplicity and using the projected-column bound $|D^{l,t}_{\mathrm{SNN}}| \leq 2^n$ gives the simpler loose bound

$$O\left(LTK\binom{2^n}{m^*}^2 n^{r^*+1}m^*\right).$$

For per-timestep supervision, the same spiked dictionaries are constructed. The additional step is assembling the trajectory-stacked dictionary

$$D^{\mathrm{traj}} = \begin{bmatrix} D^{L-1,1}_{\mathcal{T}} \\ \vdots \\ D^{L-1,T}_{\mathcal{T}} \end{bmatrix} \in \{0,1\}^{nT\times|\mathcal{T}_{L-1}|},$$

which costs

$$O(nT|\mathcal{T}_{L-1}|).$$

Thus, per-timestep supervision does not change the per-cell arrangement-construction cost. It increases the ambient dimension of the final convex program from n to nT and changes the scalar-output Carathéodory bound from $n+1$ to $nT+1$.

E Training Algorithms and Witness Regimes

This appendix gives the implementation details for the witness regimes used in Section 3.4. All convex variants follow the same template: fix a finite set of hidden LIF witnesses, roll each witness through the input sequence, construct the induced spike dictionary, and solve a convex output-layer reconstruction problem. The variants differ only in how the hidden witnesses are obtained. Surrogate-gradient variants instead update all trainable SNN parameters by backpropagation through time using a surrogate derivative for the spike activation.

E.1 Overview of implemented variants

We use five training variants as in Table 7. The two convex reconstruction methods are CVX and SG-CVX. In both cases, the hidden LIF witnesses are frozen before the convex solve, so the only optimization variables in the convex stage are the output coefficients. The finetuning methods SG-SG and CVX-SG are empirical hybrid baselines; once surrogate-gradient finetuning begins, the convex optimality guarantee no longer applies.

E.2 Complete versus sampled witnessed LIF dictionaries

This section formalizes the distinction between the complete spiked dictionary used in the exact theory and the finite sampled spiked dictionary used in the implementation.

Let $\mathcal{W}_{\text{LIF}}$ denote the LIF witness space. A witness

$$\omega = \{(P_{l,\text{in}}, B_l, R_l, U_l^0)\}_{l=1}^{L-1} \in \mathcal{W}_{\text{LIF}}$$

specifies the hidden LIF dynamics across all layers. The witness space may be continuous and therefore infinite. For a fixed training sequence $X^{1:T}$, define the final-time spike map

$$\psi_T(\omega) = S_{L-1}^T(X^{1:T}; \omega) \in \{0,1\}^n.$$

The complete projected LIF dictionary is the image of this map:

$$\mathcal{D}_{\text{SNN}}^{L-1,T} = \psi_T(\mathcal{W}_{\text{LIF}}) = \{\psi_T(\omega) : \omega \in \mathcal{W}_{\text{LIF}}\} \subseteq \{0,1\}^n.$$

Therefore,

$$|\mathcal{D}_{\text{SNN}}^{L-1,T}| \leq 2^n.$$

Thus, the projected dictionary is finite on a finite training set, even though the witness space itself may be infinite.

A complete spiked dictionary additionally stores a representative witness for each projected spike column. If

$$\mathcal{D}_{\text{SNN}}^{L-1,T} = \{d_1, \ldots, d_P\},$$

then the spiked dictionary stores pairs

$$D_{\text{SNN,dictionary}}^{L-1,T} = \{(d_i, \omega_i)\}_{i=1}^P, \qquad \psi_T(\omega_i) = d_i.$$

If multiple witnesses generate the same spike column, the witness fiber is

$$\Omega_i = \{\omega \in \mathcal{W}_{\text{LIF}} : \psi_T(\omega) = d_i\}.$$

The fiber Ω_i can be infinite. However, for reconstructing a network that matches the convex solution on the training set, it is sufficient to store one representative witness $\omega_i \in \Omega_i$.

The sampled witness regime replaces the complete witness space by a finite set

$$\widehat{\mathcal{W}}_M = \{\widehat{\omega}_1, \ldots, \widehat{\omega}_M\} \subset \mathcal{W}_{\text{LIF}}.$$

This finite set may be produced by Gaussian sampling, surrogate-gradient pretraining, or another witness-generation procedure. Once sampled, the witnesses are frozen. Rolling these witnesses through the LIF recurrence produces the finite sampled projected dictionary

$$\widehat{D}_{\text{SNN}} = [\psi_T(\widehat{\omega}_1), \ldots, \psi_T(\widehat{\omega}_M)].$$

Duplicate projected spike columns may be removed, provided that at least one generating witness is kept for every retained column.

Conditioned on the finite sampled witness set $\widehat{\mathcal{W}}_M$, the reconstruction problem is a deterministic finite convex program:

$$\widehat{p}_{\text{SNN}}^{L,T,M} = \min_{\widetilde{w} \in \mathbb{R}^{|\widehat{\mathcal{W}}_M|}} \mathcal{L}(\widehat{D}_{\text{SNN}}\widetilde{w}, Y) + \frac{\beta_{\text{reg}}}{\sqrt{m_{L-1}}}\|\widetilde{w}\|_1.$$

Thus, a continuous sampling distribution does not create an infinite optimization problem. The randomness appears only in the construction of the finite dictionary. After conditioning on the draw, the convex program optimizes only the output coefficients and is globally optimal for the sampled spiked dictionary.

E.3 Reconstructing an LIF-SNN from the convex solution

The complete spiked dictionary stores pairs (d_i, ω_i), where $d_i \in \{0,1\}^n$ is a spike column and ω_i is at least one LIF witness that realizes it on the training set. Therefore, after solving the finite convex program, no inverse realization problem is needed: each active coefficient retrieves a stored witness and becomes one parallel LIF subnetwork.

If witness metadata is discarded and only projected spike columns are stored, reconstruction becomes a separate trajectory-realization problem. One must recover membrane states and LIF parameters that reproduce the selected spike trajectory across all layers and timesteps. This problem is substantially harder than output-layer convex reconstruction and can become nonconvex when leak parameters are optimized, because the recurrence contains products between membrane variables and leak coefficients. Our theory avoids this inverse problem by defining the complete dictionary as witnessed.

Algorithm 1: Witnessed LIF-SNN Reconstruction from Convex Solution

Input: spiked dictionary $D_{\text{SNN}}^{L-1,T} = [d_1, \ldots, d_P]$ with witness map $i \mapsto \omega_i$; convex solution $\widetilde{w}^* \in \mathbb{R}^P$; final hidden width m_{L-1}

Output: K-parallel LIF-SNN parameters $\Theta = \{(\omega_k, P_{k,\text{out}})\}_{k=1}^K$ realizing the convex predictor on the training set

Set $\mathcal{S} \leftarrow \{i : \widetilde{w}_i^* \neq 0\}$;

`// There exists an equivalent scalar-output solution with` $|\mathcal{S}| \leq n+1$.

Set $K \leftarrow |\mathcal{S}|$;

for $k = 1, \ldots, K$ **do**

  Let i_k be the k-th active index in $\mathcal{S}$;

  Retrieve the stored witness $\omega_k \leftarrow \omega_{i_k}$;

  Set subnetwork k's hidden LIF parameters to ω_k;

  Set

$$P_{k,\text{out}} \leftarrow \frac{\widetilde{w}_{i_k}^*}{m_{L-1}} \mathbf{1} \in \mathbb{R}^{m_{L-1}}.$$

return $\Theta = \{(\omega_k, P_{k,\text{out}})\}_{k=1}^K$;

E.4 Surrogate-gradient pretraining: SG

SG is the standard surrogate-gradient LIF-SNN baseline. It trains the hidden LIF dynamics and the output layer jointly by backpropagation through time using a surrogate derivative for the spike activation.

Algorithm 2: SG

Input: Training data $\{X^t\}_{t=1}^T$, labels Y, LIF-SNN architecture, surrogate loss $\mathcal{L}_{\text{sg}}$, number of epochs E

Output: Pretrained LIF-SNN parameters Θ_{LIF}

Initialize hidden parameters $\{P_{l,\text{in}}, B_l, R_l\}_{l=1}^{L-1}$ and output weights P_{out};

for $e = 1, \ldots, E$ **do**

  Initialize membrane and spike states $\{U_l^0, S_l^0\}_{l=1}^{L-1}$;

  for $t = 1, \ldots, T$ **do**

    for $l = 1, \ldots, L-1$ **do**

      Compute

$$U_l^t = S_{l-1}^t P_{l,\text{in}} + U_l^{t-1} B_l - S_l^{t-1} R_l.$$

      Compute spikes $S_l^t \leftarrow \sigma(U_l^t)$ using the surrogate-gradient spike function;

  Compute prediction $f_\Theta(X^{1:T})$ using the chosen readout rule;

  Compute surrogate-gradient loss $\mathcal{L}_{\text{sg}}(f_\Theta(X^{1:T}), Y)$;

  Update all trainable parameters Θ by BPTT with surrogate gradients;

return Θ_{LIF};

E.5 Gaussian spiked dictionaries: CVX

CVX follows the Convex-Lasso approximation of Ergen et al. (2023). Instead of enumerating the complete witnessed dictionary, we sample a finite set of hidden LIF witnesses and solve the convex problem on the induced spike dictionary. For each layer l and subnetwork k, the sampled witness consists of input-to-membrane weights and a LIF-structured recurrent block,

$$G_{l,k,\mathrm{in}} \in \mathbb{R}^{m_{l-1,k} \times m_{l,k}}, \qquad G_{l,k,\mathrm{rec}} = \begin{bmatrix} B_{l,k} \\ -R_{l,k} \end{bmatrix}.$$

The same witness is reused across all timesteps; recurrent weights are not resampled independently at each timestep. Related capacity and separation results motivate Gaussian witnesses as high-capacity random features (Vershynin, 2020; Dirksen et al., 2022), but they do not prove completeness of Gaussian recurrent or LIF-SNN dictionaries.

Algorithm 3: CVX

Input: Training data $\{X^t\}_{t=1}^T$, labels Y, widths $\{m_{l,k}\}$, number of subnetworks K, regularization β_{reg}

Output: Gaussian-sampled dictionary $\widehat{D}_{\mathrm{G}}$ and convex output weights $\widetilde{W}$

for $k = 1, \ldots, K$ **do**
  Set $S_{0,k}^t \leftarrow X^t$ for all $t \in [T]$;
  for $l = 1, \ldots, L-1$ **do**
    Sample and fix Gaussian input weights $G_{l,k,\mathrm{in}} \in \mathbb{R}^{m_{l-1,k} \times m_{l,k}}$;
    Sample or fix LIF-structured recurrent parameters $B_{l,k} = \mathrm{Diag}(\beta_{l,k})$ and $R_{l,k} = \mathrm{Diag}(U_{\mathrm{thr},l,k})$;
    Initialize $U_{l,k}^0 \leftarrow 0$ and $S_{l,k}^0 \leftarrow 0$;
    for $t = 1, \ldots, T$ **do**
      Compute
$$U_{l,k}^t = S_{l-1,k}^t G_{l,k,\mathrm{in}} + U_{l,k}^{t-1} B_{l,k} - S_{l,k}^{t-1} R_{l,k}.$$
      Compute $S_{l,k}^t \leftarrow \sigma(U_{l,k}^t)$;
  Store final-layer spike features $\{S_{L-1,k}^t\}_{t=1}^T$;
Construct the sampled spiked dictionary $\widehat{D}_{\mathrm{G}}$ from the stored final-layer spike features;
Solve
$$\widetilde{W} \leftarrow \arg\min_{W} \ \mathcal{L}(\widehat{D}_{\mathrm{G}} W, Y) + \beta_{\mathrm{reg}} \|W\|_1.$$
return $\widehat{D}_{\mathrm{G}}, \widetilde{W}$;

E.6 Surrogate-pretrained spiked dictionaries: SG-CVX

SG-CVX tests whether surrogate-gradient training can produce a more useful spiked dictionary than unstructured Gaussian sampling. We first train a LIF-SNN on the same training distribution, extract its hidden dynamics, freeze them, and solve convex reconstruction over the resulting spike dictionary. We do not claim that surrogate-pretrained witnesses have higher separation capacity than Gaussian witnesses; empirically, they may be better aligned with the task distribution and more stable under length extrapolation.

Algorithm 4: SG-CVX

Input: Pretrained LIF-SNN parameters Θ_{LIF}, training data $\{X^t\}_{t=1}^T$, labels Y, regularization β_{reg}
Output: Pretrained LIF dictionary $\widehat{D}_{\mathrm{LIF}}$ and convex output weights $\widetilde{W}$

Extract hidden LIF parameters $\{\widehat{P}_{l,\mathrm{in}}, \widehat{B}_l, \widehat{R}_l\}_{l=1}^{L-1}$ from Θ_{LIF};
Freeze the extracted hidden dynamics;
for $k = 1, \ldots, K$ **do**
  Set $S_{0,k}^t \leftarrow X^t$ for all $t \in [T]$;
  for $l = 1, \ldots, L-1$ **do**
    Assign the frozen pretrained hidden block $(\widehat{P}_{l,k,\mathrm{in}}, \widehat{B}_{l,k}, \widehat{R}_{l,k})$;
    Initialize $U_{l,k}^0 \leftarrow 0$ and $S_{l,k}^0 \leftarrow 0$;
    for $t = 1, \ldots, T$ **do**
      Compute
$$U_{l,k}^t = S_{l-1,k}^t \widehat{P}_{l,k,\mathrm{in}} + U_{l,k}^{t-1} \widehat{B}_{l,k} - S_{l,k}^{t-1} \widehat{R}_{l,k}.$$
      Compute $S_{l,k}^t \leftarrow \sigma(U_{l,k}^t)$;
  Store final-layer spike features $\{S_{L-1,k}^t\}_{t=1}^T$;
Construct the pretrained spike dictionary $\widehat{D}_{\mathrm{LIF}}$ from the stored final-layer spike features;
Solve
$$\widetilde{W} \leftarrow \arg\min_{W} \ \mathcal{L}(\widehat{D}_{\mathrm{LIF}} W, Y) + \beta_{\mathrm{reg}} \|W\|_1.$$
return $\widehat{D}_{\mathrm{LIF}}, \widetilde{W}$;

E.7 Surrogate-gradient finetuning from LIF: SG-SG

SG-SG is the standard finetuning baseline initialized from a surrogate-pretrained LIF-SNN. It is included to separate gains from pretraining alone from gains due to convex reconstruction.

Algorithm 5: SG-SG

Input: Pretrained LIF-SNN parameters Θ_{LIF}, finetuning data $\{X_{\mathrm{ft}}^t\}_{t=1}^T$, labels Y_{ft}, surrogate loss $\mathcal{L}_{\mathrm{sg}}$, finetuning epochs E_{ft}
Output: Finetuned LIF-SNN parameters $\Theta_{\mathrm{FT}\leftarrow\mathrm{LIF}}$

Initialize $\Theta \leftarrow \Theta_{\mathrm{LIF}}$;
for $e = 1, \ldots, E_{\mathrm{ft}}$ **do**
  Initialize membrane and spike states $\{U_l^0, S_l^0\}_{l=1}^{L-1}$;
  for $t = 1, \ldots, T$ **do**
    Roll out the LIF-SNN on X_{ft}^t;
  Compute prediction $f_\Theta(X_{\mathrm{ft}}^{1:T})$;
  Compute surrogate-gradient loss $\mathcal{L}_{\mathrm{sg}}(f_\Theta(X_{\mathrm{ft}}^{1:T}), Y_{\mathrm{ft}})$;
  Update all trainable parameters using surrogate gradients;
Set $\Theta_{\mathrm{FT}\leftarrow\mathrm{LIF}} \leftarrow \Theta$;
return $\Theta_{\mathrm{FT}\leftarrow\mathrm{LIF}}$;

E.8 Surrogate-gradient finetuning from CVX: CVX-SG

CVX-SG tests the reverse direction: whether a convex output-layer solution gives a useful initialization for subsequent surrogate-gradient training. Since the finetuning stage updates hidden recurrent weights, this method is an empirical hybrid rather than a convex optimization method.

Algorithm 6: CVX-SG

Input: CVX sampled hidden witnesses, convex output weights $\widetilde{W}$, finetuning data $\{X_{\text{ft}}^t\}_{t=1}^T$, labels Y_{ft}, surrogate loss $\mathcal{L}_{\text{sg}}$, finetuning epochs E_{ft}

Output: Finetuned LIF-SNN parameters $\Theta_{\text{FT}\leftarrow\text{CVX}}$

Initialize hidden LIF weights using the sampled dictionary witnesses from CVX;
Initialize the output layer using the convex solution $\widetilde{W}$;
for $e = 1, \ldots, E_{\text{ft}}$ **do**
  Initialize membrane and spike states $\{U_l^0, S_l^0\}_{l=1}^{L-1}$;
  for $t = 1, \ldots, T$ **do**
    Roll out the initialized LIF-SNN on X_{ft}^t;
  Compute prediction $f_\Theta(X_{\text{ft}}^{1:T})$;
  Compute surrogate-gradient loss $\mathcal{L}_{\text{sg}}(f_\Theta(X_{\text{ft}}^{1:T}), Y_{\text{ft}})$;
  Update all trainable parameters, including hidden recurrent weights and output weights;
Set $\Theta_{\text{FT}\leftarrow\text{CVX}} \leftarrow \Theta$;
return $\Theta_{\text{FT}\leftarrow\text{CVX}}$;

F Arithmetic Addition Experimental Details

Compute Resources All pure convex reconstruction runs were trained primarily on a local MacBook M4 Pro with 24 GB of RAM. Surrogate-gradient training and finetuning runs were performed on A100 GPU partitions, including A100x4 and A100x8 configurations.

Why addition and autoregressive rollout? We use arithmetic addition as a controlled probe of temporal algorithmic generalization. Prior work on sequence models and Transformers uses arithmetic in the same spirit: models can fit short training lengths while failing to extrapolate to longer digit sequences, and successful length generalization often requires architectural or supervision choices that expose the task structure (Jelassi et al., 2023; Cho et al., 2024a;b). Addition is therefore useful not because the input-output map is intrinsically difficult, but because correct extrapolation requires a reusable carry-transition rule.

Our main protocol uses a carry-augmented recurrent formulation. At each timestep, the model predicts both the sum digit and the next carry. This teacher-forced supervision gives direct access to the latent state that mediates the recurrent computation. At evaluation time, we use autoregressive rollout, where the model must reuse its own predicted carry over longer horizons. Thus, the benchmark separates short-horizon fitting from stable recurrent extrapolation. This is also the setting most aligned with our spiked dictionary theory: a useful LIF feature is not merely a static output pattern, but a recurrent trajectory generated by hidden membrane and spike dynamics.

We also evaluate a direct operand-to-output variant without explicit latent carry supervision. That setting serves as a protocol ablation: it asks whether convex reconstruction improves local digit prediction even when the model is not forced to learn an interpretable carry state. In our results, the direct variant improves token-level prediction and delays first errors, but does not replace the recurrent rollout benchmark as the main test of length generalization.

Task and splits. We train on addition in bases $b \in \{2, 3, 5\}$ with $n_{\text{digits}} = 5$, so the training horizon is $T = n_{\text{digits}} + 1 = 6$. Each timestep input is the normalized tuple $(a_t, b_t, c_t^{\text{in}})$, where a_t and b_t are the operand digits and c_t^{in} is the carry into the current digit column. The model predicts the sum digit y_t^{sum} and the next carry y_t^{carry} at every timestep. All bases are averaged over seeds $\{0, 1, 2\}$. Each run uses separate pretraining and finetuning splits: $n_{\text{train,pre}} = 2304$, $n_{\text{val,pre}} = 512$, $n_{\text{train,ft}} = 2304$, and $n_{\text{val,ft}} = 512$. The held-out ID test set has $n_{\text{test}} = 1024$, and each OOD split has $n_{\text{test,ood}} = 1024$. For base 2, we evaluate $n_{\text{digits}} \in \{10, 20, 50\}$; for bases 3 and 5, we evaluate $n_{\text{digits}} \in \{10, 25, 50\}$.

Architecture and training variants. All addition runs use a K-parallel LIF-SNN with

$$L = 3, \qquad P_{\text{rec}} = 256, \qquad P_{\text{last}} = 512, \qquad K = 2.$$

Both SG and CVX use final-layer spike readout, so the convex dictionary is built from binary spike features. This matches the witnessed spike-dictionary formulation in the theory. For bases $b > 2$, the hidden dictionary remains binary, while the output head is multiclass and maps spike features to one of b sum digits. The five training variants are exactly those defined in Appendix E: SG, SG-CVX, SG-SG, CVX, and CVX-SG.

Losses and hyperparameter sweeps. The carry-augmented objective is

$$\mathcal{L}_{\text{joint}} = \lambda_{\text{sum}}\mathcal{L}_{\text{sum}} + \lambda_{\text{carry}}\mathcal{L}_{\text{carry}},$$

with $\lambda_{\text{sum}} = 1$. We sweep

$$\lambda_{\text{carry}} \in \{0.125, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 2.0, 4.0, 6.0, 8.0, 10.0\}.$$

The surrogate-gradient stages use Adam for 100 pretraining epochs and 100 finetuning epochs. The SG learning-rate grid is

$$\{10^{-3}, 5 \cdot 10^{-3}, 10^{-2}, 10^{-1}\},$$

and the SG and CVX regularization grids are both

$$\{10^{-2}, 10^{-1}, 0.5, 1.0, 5.0, 10.0\}.$$

Because spike readout is used, the CVX bias grid is fixed to $\{0.0\}$. Both SG and CVX use ramped timestep weighting and the joint teacher-forcing objective during training.

Evaluation metrics. Every trained stage is evaluated under both teacher forcing and autoregressive rollout. Unless otherwise stated, the main paper reports autoregressive evaluation. The primary metric is joint-token accuracy:

$$\frac{1}{nT}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbf{1}\{\hat{y}_{i,t}^{\text{sum}} = y_{i,t}^{\text{sum}}\}\mathbf{1}\{\hat{y}_{i,t}^{\text{carry}} = y_{i,t}^{\text{carry}}\}.$$

A timestep is counted as correct only when both the sum digit and carry prediction are correct. We also track sum-token accuracy, carry-token accuracy, joint-sequence accuracy, and first-error timestep statistics.

Carry-loss sensitivity. The appendix figures report autoregressive joint-token accuracy as a function of λ_{carry} for ID and OOD splits. These plots show the tradeoff between enforcing the latent carry variable and maintaining stable rollout behavior at longer horizons. The main tables report validation-selected configurations aggregated over seeds.

Aggregation. The main tables aggregate results over seeds after validation selection over the corresponding hyperparameter grid and λ_{carry} sweep. The sensitivity plots in this appendix show the full λ_{carry} dependence before selecting the final reported operating point.

G Broader Impact

This research may serve as a stepping stone toward training more energy-efficient and biologically plausible AI models at scale. Beyond SNNs, the reconstruction viewpoint is also naturally aligned with reservoir computing, where internal recurrent dynamics are fixed and only the final readout layer is trained, similar in spirit to our CVX-G and SG-CVX algorithms. More broadly, globally optimized readout reconstruction can provide a principled way to reuse fixed or pretrained dynamical systems without relying entirely on end-to-end gradient-based training. If developed further, this could reduce the computational and energy costs of training temporal models, while also offering a clearer diagnostic framework for understanding when failures arise from optimization, representation, or the choice of internal dynamics.

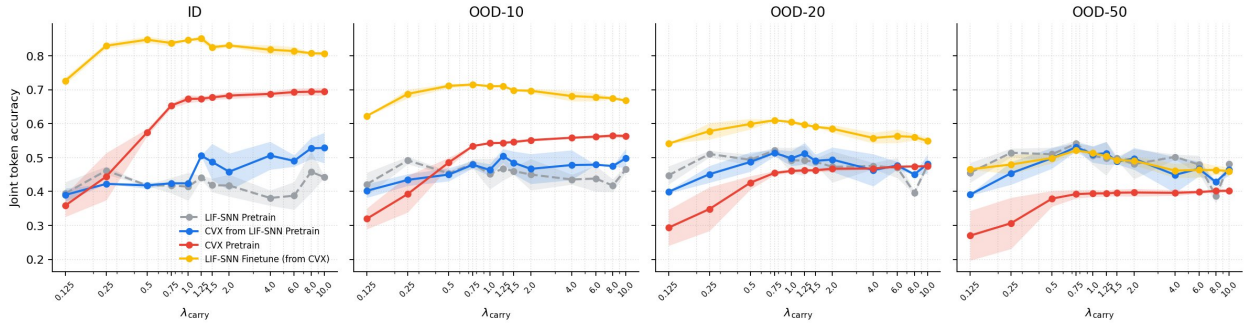

Figure 1: Base-2 addition: effect of λ_{carry} on autoregressive joint-token accuracy for ID and OOD splits. Results are averaged over three seeds. The architecture is $L = 3$, $P_{\text{rec}} = 256$, $P_{\text{last}} = 512$, and $K = 2$, with final-layer spike readout for both SG and CVX. OOD lengths are $n_{\text{digits}} \in \{10, 20, 50\}$.

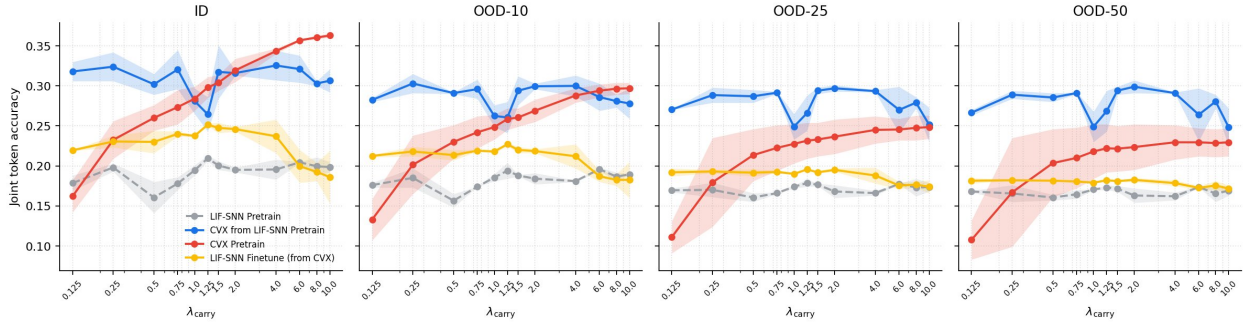

Figure 2: Base-3 addition: effect of λ_{carry} on autoregressive joint-token accuracy for ID and OOD splits. Results are averaged over three seeds. The architecture and readout match Figure 1; OOD lengths are $n_{\text{digits}} \in \{10, 25, 50\}$.

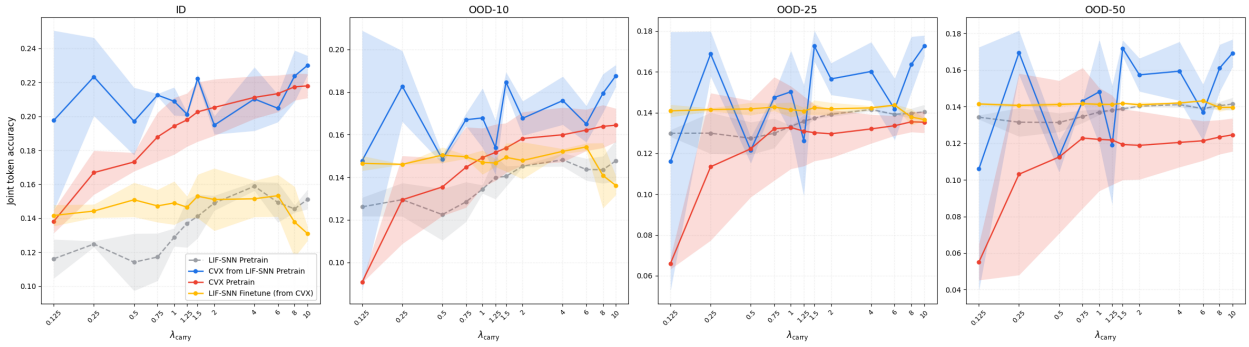

Figure 3: Base-5 addition: effect of λ_{carry} on autoregressive joint-token accuracy for ID and OOD splits. Results are averaged over the available two seeds. The architecture is the same as the base-2 and base-3 experiments: $L = 3$, $P_{\text{rec}} = 256$, $P_{\text{last}} = 512$, and $K = 2$, with final-layer spike readout for both SG and CVX. OOD lengths are $n_{\text{digits}} \in \{10, 25, 50\}$.