BRAIN-PLAUSIBLE LOCAL LEARNING IN SNNS: TRAINING HODGKIN–HUXLEY SPIKING NEURAL NETWORKS WITH PREDICTIVE CODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Almost all deep spiking neural networks (SNNs) use Leaky Integrate-and-Fire (LIF) neurons and are trained by backpropagation through time (BPTT) with surrogate gradients. Both choices are made for tractability, not biology: LIF throws away the ion-channel dynamics that generate real action potentials, and BPTT needs a global error signal no cortical mechanism is known to implement. Conductance-based models such as Hodgkin–Huxley (HH) are avoided because unrolling their coupled nonlinear dynamics through time is expensive and numerically fragile. We show the trade-off is avoidable. Our method trains a network of HH neurons for classification with no backpropagation, no surrogate gradients, and no unrolling, using predictive coding (PC) to supply layer-local errors. One design choice makes this work: a *rate proxy*, the spike count of each HH population divided by the simulation window, which gives PC the continuous states it needs. No gradient ever crosses spike generation, so the cost of learning does not depend on how complex the neuron is. Across MNIST, Fashion-MNIST, N-MNIST and the Spiking Heidelberg Digits, HH matches a LIF baseline trained under the same objective while firing fewer spikes in 26 of 28 window-matched comparisons. Our learning interface allows biophysically realistic neurons to retain competitive accuracy.

1 INTRODUCTION

Spiking neural networks compute with discrete events rather than continuous activations, which makes them a natural fit for event-driven neuromorphic hardware where energy is paid per spike (Maass, 1997; Davies et al., 2018). Two design decisions dominate current practice, and both are made for tractability rather than fidelity.

The first concerns the neuron. Almost all deep SNNs use the Leaky Integrate-and-Fire model, which reduces a neuron to a leaky accumulator with a threshold. The action potential itself, the regenerative interplay of voltage-gated sodium and potassium conductances that Hodgkin and Huxley characterised (1952), is replaced by a reset. What is discarded is not decorative: gating kinetics govern refractoriness, spike timing, and frequency adaptation, and these shape how much activity a network needs in order to represent a stimulus (Izhikevich, 2004; Gerstner & Kistler, 2002).

The second concerns the learning rule. SNNs are typically trained by backpropagation through time with a surrogate gradient standing in for the non-differentiable spike (Neftci et al., 2019; Wu et al., 2018). This works, but it requires storing and unrolling the entire temporal computation graph, and it propagates a single global error backwards through the network, a mechanism with no clear biological counterpart (Lillicrap et al., 2020).

The two decisions make each other worse. Training HH neurons with BPTT means differentiating through T steps of four coupled nonlinear variables per neuron, which is expensive and prone to the numerical blow-ups that stiff differential equations invite. The usual conclusion is that biophysical detail is a luxury nobody can afford at scale, so the field settles for LIF.

We argue that conclusion follows from the coupling, not from HH itself. If the learning rule never has to differentiate through the spiking dynamics, how complex the neuron is stops being a training

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

cost. Predictive coding provides exactly this: it casts learning as the minimisation of local prediction errors between adjacent layers, so each synapse updates from presynaptic activity and postsynaptic error alone (Rao & Ballard, 1999; Whittington & Bogacz, 2017). One obstacle remains: PC works on continuous states, while spiking layers emit binary events.

We bridge this with a **rate proxy**: the time-averaged spike count of each HH population, normalised to $[0, 1]$, taken as the continuous latent state for PC inference. The HH simulation runs forward once, in the manner it would on neuromorphic hardware; learning then operates entirely on the resulting rate representation. No gradient crosses the spike generation mechanism, so no surrogate is needed and no unrolling occurs.

The main contributions of this paper are summarised as follows:

- **Hodgkin–Huxley neurons trained end-to-end without backpropagation.** To our knowledge the first SNN built from full conductance-based HH neurons trained for classification without BPTT or surrogate gradients. The learning rule is Hebbian in form and layer-local; predictive coding supplies the error signal, so no unrolling is required.
- **Parity with LIF, at a lower spike count.** Against a LIF baseline trained under an identical PC objective, HH matches accuracy on every dataset while firing fewer spikes at almost every simulation window. We are explicit about the scope: HH's per-neuron-step integration is more expensive than LIF's, so the advantage lies in the spike domain rather than in total operations (Section 4.2).
- **The rate-proxy bridge.** The interface that makes the above possible: a time-averaged, normalised spike count taken as the continuous latent state for PC inference. Because no gradient crosses spike generation, the cost of learning is set by the size of the rate representation rather than by the complexity of the dynamics that produced it. This is precisely why a conductance-based neuron becomes affordable to train.

2 Related Work

Training spiking networks. Because the spike is a step function, SNN training centres on how to get a gradient through it. Surrogate-gradient methods substitute a smooth function in the backward pass and apply BPTT (Neftci et al., 2019; Wu et al., 2018; Lee et al., 2020); ANN-to-SNN conversion sidesteps the problem by training a rate network and transferring weights (Rueckauer et al., 2017); temporal-coding schemes differentiate spike times directly (Bohte et al., 2002). All retain a global backward pass. Our method has neither a backward pass nor a surrogate.

Biologically plausible credit assignment. Feedback alignment shows random backward weights suffice for learning (Lillicrap et al., 2016); dendritic microcircuits approximate backpropagation with cortical structure (Sacramento et al., 2018); the forward-forward algorithm replaces the backward pass with a second contrastive forward pass (Hinton, 2022), since extended to spiking systems (Ororbia, 2024; Ghader et al., 2026; Terres-Escudero et al., 2026). Predictive coding, in which each layer predicts the one below and learns from the residual (Rao & Ballard, 1999; Friston, 2005), is attractive because its weight update provably approximates backpropagation while remaining local (Whittington & Bogacz, 2017).

Predictive coding in SNNs. PC-SNN applies PC-based Hebbian plasticity to spiking networks with integrate-and-fire neurons (Wang et al., 2026); related work combines PC with STDP for compact visual codes (N'dri et al., 2025) and studies PC with spiking neurons and gist signalling (Lee et al., 2024). These use simplified neuron models. Our contribution is orthogonal to the choice of PC formulation: we show the same style of local objective supports a fully conductance-based neuron, and quantify what that buys.

Hodgkin–Huxley in machine learning. HH neurons appear mostly in simulation and hardware-emulation studies (Fang et al., 2021; Beaubois et al., 2024) rather than as trainable classifiers. Florio et al. (2023) compare biologically plausible neuron models for regression. The prevailing view holds richer models too costly for learning (Izhikevich, 2004); we show the cost is attributable to the training algorithm, not the neuron.

3 METHOD

Our network, PCHH-SNN, has two computational passes with distinct roles. A *spiking forward pass* propagates binary spikes through HH populations over T discrete steps. A *PC inference pass* operates on continuous states derived from that spiking activity. The rate proxy is what joins them (Figure 1).

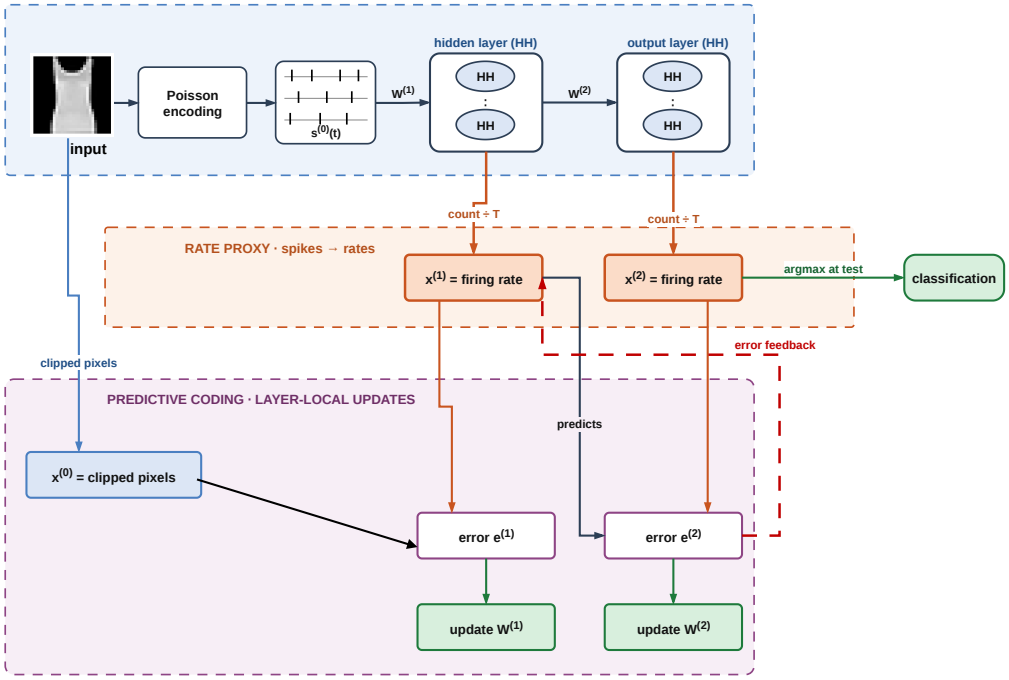


Figure 1: PCHH-SNN, one training step. The input image (here one image from Fashion-MNIST) is Poisson-encoded into spike trains that drive the hidden and output HH populations over T steps; no gradient is retained through this pass. Each population's spike count divided by T gives a firing-rate vector in $[0, 1]$, which initialises that layer's PC state. The input state $\boldsymbol{x}^{(0)}$ is the normalised image itself, taken directly from the pixels rather than from spikes. Each state is predicted from the layer below and the residuals $\boldsymbol{e}^{(l)}$ settle $\boldsymbol{x}^{(1)}$ over T_{infer} iterations (Eq. 4; only the top-down term from $\boldsymbol{e}^{(2)}$ is drawn). Each weight then updates from its own error and the activity below it. At test the class is $\arg\max_k x_k^{(2)}$ after settling; in training $\boldsymbol{x}^{(2)}$ is clamped to the target.

Figure 1 shows one training step. Reading top to bottom, the figure has three bands. In the blue band the image becomes a spike train and those spikes drive two HH populations for T steps, exactly as they would on event-driven hardware; nothing here is differentiated. The orange band is the bridge: each population's spikes are counted and divided by T, turning a binary event stream into a firing rate in $[0, 1]$. The purple band is where learning happens, and it sees only those rates. Each layer tries to predict the rate of the layer above it from the rate below; the mismatch is the prediction error, and errors are what both settle the hidden state and drive the weight update. The input is the one exception to the pattern: it is already continuous, so it enters the purple band directly as clipped pixels rather than through the spiking path.

3.1 TRAINING PROCEDURE

Algorithm 1 assembles the three phases. Phase 1 runs the spiking simulation forward once, retaining only accumulated spike counts. Phase 2 settles the latent states under the local objective. Phase 3 updates weights from quantities each synapse can access locally. No gradient crosses the boundary between Phase 1 and Phase 2.

Algorithm 1 PCHH-SNN training step (one mini-batch)

Require: Batch $\boldsymbol{X}$, targets $\boldsymbol{Y}$, batch size B, synaptic layers L, spiking steps T, inference steps T_{infer}, inference rate γ_x, gradient clip κ
Ensure: Updated weights $\{\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)}\}_{l=1}^{L}$

Phase 1: spiking forward pass ▷ *No gradient is retained*
$\boldsymbol{s}^{(0)}(1{:}T) \leftarrow \textsc{Encode}(\boldsymbol{X})$ ▷ *Input spike train*
$\boldsymbol{S}_l \leftarrow \mathbf{0}$, reset neuron states, $l = 1..L$ ▷ *Initialise spike counters*
$\boldsymbol{x}^{(0)} \leftarrow \text{clip}(\boldsymbol{X}, 0, 1)$ ▷ *Input state, held fixed*
for $t = 1$ **to** T **do**
  for $l = 1$ **to** L **do**
    $\boldsymbol{u}^{(l)}(t) \leftarrow \boldsymbol{W}^{(l)}\boldsymbol{s}^{(l-1)}(t) + \boldsymbol{b}^{(l)}$ ▷ *Receive spike input*
    $I^{(l)}(t) \leftarrow g_l\, \boldsymbol{u}^{(l)}(t) + I_l^{\text{bias}}$ ▷ *Convert to injected current*
    $\boldsymbol{s}^{(l)}(t) \leftarrow \textsc{NeuronStep}(I^{(l)}(t))$ ▷ *HH gating, Euler step, threshold*
    $\boldsymbol{S}_l \leftarrow \boldsymbol{S}_l + \boldsymbol{s}^{(l)}(t)$ ▷ *Accumulate spikes*
  end for
end for
$\boldsymbol{x}^{(l)} \leftarrow \boldsymbol{S}_l/T,\ l = 1..L$ ▷ *Rate proxies, Eq. (2)*
Phase 2: predictive-coding inference ▷ *Operates on rates only*
$\boldsymbol{x}^{(L)} \leftarrow \boldsymbol{Y}$ ▷ *Clamp output (training only)*
for $i = 1$ **to** T_{infer} **do**
  for $l = 1$ **to** L **do**
    $\boldsymbol{z}^{(l)} \leftarrow \boldsymbol{W}^{(l)}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$ ▷ *Predict layer l from layer $l-1$*
    $\boldsymbol{e}^{(l)} \leftarrow \boldsymbol{x}^{(l)} - f(\boldsymbol{z}^{(l)})$ ▷ *Prediction error*
  end for
  for $l = 1$ **to** $L - 1$ **do**
    $\boldsymbol{x}^{(l)} \leftarrow \text{clip}\big(\boldsymbol{x}^{(l)} - \gamma_x\big[\boldsymbol{e}^{(l)} - (\boldsymbol{e}^{(l+1)} \odot f'(\boldsymbol{z}^{(l+1)}))\boldsymbol{W}^{(l+1)}\big],\ 0,\ 1\big)$ ▷ *Settle, Eq. (4)*
  end for
end for
Phase 3: local parameter update
for $l = 1$ **to** L **do**
  $\boldsymbol{\delta}^{(l)} \leftarrow \boldsymbol{e}^{(l)} \odot f'(\boldsymbol{z}^{(l)})$ ▷ *Error-modulated signal*
  $\nabla_{\boldsymbol{W}^{(l)}} \leftarrow -\frac{1}{B}\,\boldsymbol{\delta}^{(l)}\boldsymbol{x}^{(l-1)\top}, \quad \nabla_{\boldsymbol{b}^{(l)}} \leftarrow -\frac{1}{B}\sum_{\text{batch}} \boldsymbol{\delta}^{(l)}$ ▷ *Hebbian, Eq. (5)*
end for
Clip global gradient norm to κ; Adam step ▷ *Update weights*

3.2 Architecture and Input Encoding

The network is feedforward with L=2 synaptic layers, $D_{\text{in}} \to H \to C$, with H=512 (400 on SHD), $D_{\text{in}} \in \{784, 2312, 7000\}$ dataset-dependent, and $C \in \{10, 20\}$. Every post-input layer is a population of HH spiking neurons. Weights $\boldsymbol{W}^{(l)}$ and biases $\boldsymbol{b}^{(l)}$ are the only learnable parameters; all HH biophysical constants are fixed at their standard values. Initialisation is Xavier uniform with gain 0.5 (Glorot & Bengio, 2010).

Static inputs (pixels, or binned spike counts on SHD) are converted to spikes by discrete-time Bernoulli (Poisson-like) rate coding. For input dimension i, the per-step firing probability is $p_i = \min(\alpha x_{0,i}, 1)$ with α=1.0, and spikes are drawn independently at each step, $s_{0,i}(t) \sim \text{Bernoulli}(p_i)$. N-MNIST is already event-based, so its binned events drive the first HH layer directly, without Poisson encoding. We also evaluate latency-first coding as an ablation.

3.3 Hodgkin–Huxley Spiking Dynamics

At each step, the presynaptic spike vector is projected through the synaptic weights to give a drive $\boldsymbol{u}^{(l)}(t) = \boldsymbol{W}^{(l)}\boldsymbol{s}^{(l-1)}(t) + \boldsymbol{b}^{(l)}$, converted to an injected current $I^{(l)}(t) = g_l\,\boldsymbol{u}^{(l)}(t) + I_l^{\text{bias}}$ with g_l=30 and I_l^{bias}=2. These are calibrated to hold the network in a non-silent, non-saturated regime (Table 3): too little gain and the rate proxies collapse to zero, too much and they saturate and carry no discriminative information.

Membrane voltage follows the classical conductance-based formulation (Hodgkin & Huxley, 1952) (per-step diagram in Figure 3):

$$C_m \frac{dV}{dt} = I(t) - I_{\text{Na}}(t) - I_{\text{K}}(t) - I_{\text{L}}(t), \tag{1}$$

with $I_{\text{Na}} = \bar{g}_{\text{Na}} m^3 h (V - E_{\text{Na}})$, $I_{\text{K}} = \bar{g}_{\text{K}} n^4 (V - E_{\text{K}})$, and $I_{\text{L}} = \bar{g}_{\text{L}} (V - E_{\text{L}})$. The gating variables $m, n, h \in [0, 1]$ follow first-order voltage-dependent kinetics $\dot{q} = \alpha_q(V)(1 - q) - \beta_q(V) q$ with the standard HH rate functions, and parameters C_m=1, $\bar{g}_{\text{Na}}$=120, $\bar{g}_{\text{K}}$=36, $\bar{g}_{\text{L}}$=0.3, E_{Na}=115, E_{K}=−12, E_{L}=10.6.

Integration uses forward Euler with $\Delta t = 0.03$ ms. Two numerical safeguards matter in practice. Voltage is clamped to $[-100, 100]$ mV before evaluating the rate functions, keeping their exponentials finite (App. D); and a smooth saturation $V \leftarrow V_{\max} \tanh(V/V_{\max})$ with $V_{\max}$=30 mV is applied after each step. The latter is near-linear within the normal spiking range, so it leaves HH dynamics undistorted while preventing floating-point overflow under large injected currents. Gating variables are initialised to their steady-state values $q_\infty(V_0)$ at V_0=0 so simulations do not open with a spurious burst.

A spike is emitted when voltage crosses threshold θ (1.2 mV; 0.8 mV on SHD) and the neuron is not refractory: $s(t) = \mathbb{I}[V(t) \geq \theta] \cdot \mathbb{I}[r(t)=0]$. On spiking, voltage resets to 0 mV and a refractory counter is set to $\tau_{\text{ref}}/\Delta t$ steps. With τ_{ref}=2 ms this is 67 steps, so for $T \leq 60$ on MNIST, Fashion-MNIST and N-MNIST each neuron fires at most once per sample and the rate proxy records whether a neuron fired, scaled by $1/T$ (at T=70 a second spike becomes possible, though only if the first occurs in the opening three steps). This prevents the all-spikes regime that output neurons would otherwise enter under strong class evidence. On SHD we use τ_{ref}=0.15 ms (5 steps), which permits repeated firing (Appendix H).

3.4 The Rate Proxy

PC inference needs continuous states; HH layers emit binary events. For layer l we define the rate proxy as the time-averaged spike count,

$$\boldsymbol{x}^{(l)}_{\text{proxy}} = \text{clip}\left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^{(l)}(t),\, 0,\, 1 \right) \in [0, 1]^{D_l}, \tag{2}$$

so each component is simply the fraction of steps in which that neuron fired; the clip is a safeguard, since a spike count divided by T already lies in $[0, 1]$. This small interface is what makes the rest of the method possible. Because PC operates on $\boldsymbol{x}^{(l)}_{\text{proxy}}$ rather than on individual spike events, the learning rule never differentiates through spike generation or through the HH state updates. The cost of learning is therefore set by the size of the rate representation, not by the complexity of the dynamics that produced it. This is precisely why HH becomes affordable.

3.5 Predictive Coding Inference

After the spiking pass, predictive coding asks whether each layer's firing rate can be predicted from the layer below. Where it cannot, the prediction error is used first to adjust the hidden rates and then to update the weights (Section 3.6). Every step uses only quantities from one layer and its neighbours, so no error travels back through the whole network.

States and errors. The network holds one state per layer, $\boldsymbol{x}^{(l)} \in [0, 1]^{D_l}$. The input state $\boldsymbol{x}^{(0)}$ is the normalised input ($\text{clip}(\boldsymbol{X}, 0, 1)$, or the time-averaged event train on N-MNIST) and stays fixed; every other state starts from its rate proxy, $\boldsymbol{x}^{(l)} \leftarrow \boldsymbol{x}^{(l)}_{\text{proxy}}$, so inference begins from what the HH neurons actually did. Each layer is predicted from the one below, $\hat{\boldsymbol{x}}^{(l)} = f(\boldsymbol{z}^{(l)})$ with $\boldsymbol{z}^{(l)} = \boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$ and $f(z) = \text{clip}(z, 0, 1)$, giving the error $\boldsymbol{e}^{(l)} = \boldsymbol{x}^{(l)} - \hat{\boldsymbol{x}}^{(l)}$. The total mismatch is the energy

$$E = \frac{1}{2} \sum_{l=1}^{L} \left\| \boldsymbol{e}^{(l)} \right\|_2^2, \tag{3}$$

which we report averaged over units and samples as the PC objective.

Settling. With weights fixed, each hidden state takes T_{infer} gradient steps on E:

$$\boldsymbol{x}^{(l)} \leftarrow \text{clip}\left(\boldsymbol{x}^{(l)} - \gamma_x\Big[\underbrace{\boldsymbol{e}^{(l)}}_{\text{own error}} - \underbrace{(\boldsymbol{e}^{(l+1)} \odot f'(\boldsymbol{z}^{(l+1)}))\,\boldsymbol{W}^{(l+1)}}_{\text{feedback from above}}\Big],\, 0,\, 1\right), \tag{4}$$

with γ_x=0.05. The first term pulls $\boldsymbol{x}^{(l)}$ toward the prediction from below; the second pulls it toward values that make the layer above easier to predict. The clip keeps every state a valid firing rate. During training the output is clamped to the target, $\boldsymbol{x}^{(L)} \leftarrow \boldsymbol{y}$ (one-hot; softened on SHD, Section 4). At test time it settles freely, $\boldsymbol{x}^{(L)} \leftarrow \text{clip}(\boldsymbol{x}^{(L)} - \gamma_x \boldsymbol{e}^{(L)}, 0, 1)$, and the class is $\arg\max_k x_k^{(L)}$. We use T_{infer}=100 in training and 50 at test.

3.6 Local Learning Rule

Once states have settled, weights update from the error-modulated signal $\boldsymbol{\delta}^{(l)} = \boldsymbol{e}^{(l)} \odot f'(\boldsymbol{z}^{(l)})$:

$$\frac{\partial E}{\partial \boldsymbol{W}^{(l)}} = -\frac{1}{B}\boldsymbol{\delta}^{(l)}\boldsymbol{x}^{(l-1)\top}, \qquad \frac{\partial E}{\partial \boldsymbol{b}^{(l)}} = -\frac{1}{B}\sum_{\text{batch}} \boldsymbol{\delta}^{(l)}. \tag{5}$$

This is an outer product between a postsynaptic error and presynaptic activity, a Hebbian-like update in which each synapse sees only its own pre- and post-synaptic quantities, with no global error signal from the network output (Whittington & Bogacz, 2017). The factor $f'(\boldsymbol{z}^{(l)})$ zeroes contributions from saturated units, preventing updates that would push already-clamped states further out of range. Gradients go to Adam (Kingma & Ba, 2015) with learning rate 2×10^{-4} and weight decay 10^{-4}; because Poisson encoding injects stochasticity into the rate proxies, global gradient-norm clipping at κ=1.0 is applied before each step.

4 Experiments

We evaluate on MNIST (LeCun et al., 1998), Fashion-MNIST (Xiao et al., 2017), N-MNIST (Orchard et al., 2015) and SHD (Cramer et al., 2022). SHD contains 20 spoken digits in English and German, recorded as spikes on 700 cochlear channels.

N-MNIST events are denoised and binned into 50 time bins per polarity, giving a binary 2312×50 event train that drives the network directly. SHD spikes are binned into a 50×140 time-by-channel count frame, flattened, and Poisson-encoded like pixel intensities, so the network sees each SHD clip as one static pattern rather than a temporal stream. For validation, MNIST, Fashion-MNIST and N-MNIST use a 90/10 train/validation split with a fixed seed, whereas SHD holds out two training speakers so that validation speakers are never seen in training. Unless noted, H=512, batch size 128 (N-MNIST: 64), up to 30 epochs, and T=60 (N-MNIST, SHD: T=50). SHD differs in four ways: H=400, τ_{ref}=0.15 ms, time-shift and channel-dropout augmentation, and soft targets (0.9 on the true class, $0.1/C$ elsewhere). Soft targets matter at C=20: a one-hot clamp pushes each output unit toward silence 19 times for every push toward firing, and a silent unit ($f(z)$=0) receives no gradient.

The controlled comparison throughout is against **LIF-PC**, an identical network with LIF neurons in place of HH. Encoding, architecture, PC inference, learning rule, optimiser and hyperparameters are all held fixed, so any difference comes from the neuron model. On SHD, LIF's bias is also set to the value on a bias grid that best matches HH's initial hidden firing rate (the lowest value tried; LIF still fires about 8% more at initialisation). This lowers LIF's drive and makes the spike comparison harder for HH. Full settings and preprocessing details are in Appendices B and E.

4.1 Classification Performance

Table 1 reports test accuracy at each dataset's operating point; macro-F1 tracks accuracy on every dataset (Appendix F). Our comparative claims rest on the seed-controlled HH/LIF comparison (Table 2).

The claim we defend is that a fully biophysical neuron can be trained locally *at no accuracy cost*, and the seed-controlled comparison is consistent with exactly that. Fashion-MNIST is the one dataset

Table 1: Test accuracy of PCHH-SNN at each dataset's operating point, compared with published SNN methods. T is each method's reported simulation window. Fashion-MNIST and N-MNIST baselines are fully connected; all N-MNIST entries use both polarity channels (2312 inputs). On SHD, both prior methods are recurrent; ours is feedforward. (LIF: leaky integrate and fire, IF: integrate and fire, HH: Hodgkin–Huxley, ALIF: adaptive leaky integrate and fire, BPTT: backpropagation through time, FF: forward-forward, PC: predictive coding).

Dataset	Method	Neuron	Learning	T	Acc. (%)
MNIST	Spiking MLP (STBP) (Wu et al., 2018)	LIF	BPTT	30	98.9
	PC-SNN (Wang et al., 2026)	IF	Local PC	256	98.1
	PCHH-SNN (ours)	HH	**Local PC**	60	97.7
Fashion-MNIST	FC SNN (Perez-Nieves & Goodman, 2021)	LIF	BPTT	–	82.2
	FC SNN (Ghader et al., 2026)	LIF	FF (local)	10	89.4
	PCHH-SNN (ours)	HH	**Local PC**	60	88.1
N-MNIST	FC SNN (Perez-Nieves & Goodman, 2021)	LIF	BPTT	–	92.7
	FC SNN (Ghader et al., 2026)	LIF	FF (local)	10	97.3
	PC-SNN (Wang et al., 2026)	IF	Local PC	256	97.5
	PCHH-SNN (ours)	HH	**Local PC**	50	93.3
SHD	RSNN (Cramer et al., 2022)	LIF	BPTT	–	83.2
	Adaptive SRNN (Yin et al., 2021)	ALIF	BPTT	–	90.4
	PCHH-SNN (ours)	HH	**Local PC**	50	87.7

Table 2: HH+PC vs. LIF+PC across five seeds (42, 123, 456, 789, 1011) at T=25 (SHD: T=50), mean $\pm$ sample std. Paired t-test on accuracy. PC obj. is the converged prediction error E (Eq. 3), averaged over units and samples; lower means the layers predict each other better. It measures training convergence, not physical energy. Accuracy differences are not significant on three of four datasets, and none survives Bonferroni correction. Per-seed values are in Table 10.

Dataset	HH Acc.	LIF Acc.	HH PC obj.	LIF PC obj.	p
MNIST	$0.9749 \pm .0007$	$0.9744 \pm .0009$	1.55×10^{-4}	2.00×10^{-4}	0.417
Fashion-MNIST	$0.8750 \pm .0037$	$0.8720 \pm .0041$	3.73×10^{-4}	3.98×10^{-4}	0.028
N-MNIST	$0.9158 \pm .0397$	$0.9345 \pm .0020$	1.49×10^{-3}	5.10×10^{-4}	0.354
SHD	$0.8396 \pm .0208$	$0.8381 \pm .0081$	1.42×10^{-3}	2.18×10^{-3}	0.859

where HH's advantage reaches nominal significance (t=+3.37, p=0.028), with HH ahead on all five seeds. We do not lean on it: the macro-F1 difference is not significant (p=0.116), the p-value does not survive Bonferroni correction across our four datasets (α=0.0125), and the effect is 0.3 accuracy points. On MNIST and SHD the two models are statistically indistinguishable (p=0.417 and p=0.859; Wilcoxon p=0.81 on SHD), with HH nominally ahead on both. N-MNIST is the one dataset where LIF leads on the mean, almost entirely because of one seed whose PC objective sits an order of magnitude above the rest, the signature of an initialisation failure (Appendix G); the other four seeds are indistinguishable. Parity, not superiority, is what the claim predicts, and parity is what we observe.

4.2 Spike Economy

Table 3 reports mean spike rate for both neuron models across simulation windows. HH fires fewer spikes than LIF in 26 of 28 comparisons: at every window on the three vision datasets and at five of seven on SHD. The reduction is largest at short windows (37.0%, 39.7%, 26.4% and 35.2% at T=10 on MNIST, Fashion-MNIST, N-MNIST and SHD).

The window dependence has a simple cause on the vision datasets. With τ_{ref}=2 ms a neuron fires at most once per sample for $T \leq 60$ (Section 3.3), so rate $\times T$ is the fraction of neurons that fire, and the fall in rate with T is largely the $1/T$ normalisation. On MNIST and Fashion-MNIST this fraction is nearly constant for HH, between 0.39 and 0.42 at every window, whereas LIF recruits far

Table 3: Mean spike rate (%) against the simulation window T, HH vs. LIF, single run per cell. Red. is the reduction of HH relative to LIF, computed from unrounded rates; positive means HH fires fewer spikes. HH fires less at every window on the vision datasets and at five of seven on SHD.

	MNIST			**Fashion-MNIST**			**N-MNIST**			**SHD**		
T	HH	LIF	Red.	HH	LIF	Red.	HH	LIF	Red.	HH	LIF	Red.
10	4.060	6.443	**37.0**	3.926	6.508	**39.7**	6.743	9.156	**26.4**	5.285	8.156	**35.2**
20	2.046	2.373	13.8	2.095	2.662	21.3	4.309	4.750	9.3	5.763	7.121	19.1
30	1.329	1.469	9.5	1.346	1.604	16.1	2.968	3.231	8.1	6.364	6.196	−2.7
40	0.996	1.064	6.3	1.020	1.205	15.3	2.324	2.454	5.3	6.463	6.841	5.5
50	0.798	0.833	4.1	0.808	0.938	13.8	1.901	1.978	3.9	6.427	7.378	12.9
60	0.668	0.696	4.0	0.659	0.769	14.3	1.605	1.653	2.9	6.641	6.770	1.9
70	0.585	0.685	14.7	0.566	0.794	28.8	1.389	1.424	2.5	6.705	6.388	−5.0

more neurons at short windows (0.64 and 0.65 at T=10, against 0.42 and 0.47 at T=50). The HH advantage therefore comes from LIF over-recruiting when the window is short, not from HH firing less at long windows. The same effect, together with the possibility of a second spike at T=70, explains the upturn there: the LIF fraction rises again (0.48 on MNIST, 0.56 on Fashion-MNIST) while HH stays near 0.40. On N-MNIST both models recruit most neurons at long windows (above 0.95 at T=50), which leaves little room for a difference. SHD uses a short refractory period that permits repeated firing, so this reading does not apply there. Its two exceptions (T=30, 70; HH fires 2.7% and 5.0% more) are single runs.

On event-driven neuromorphic hardware, where energy is paid per spike and conductance dynamics may be realised physically, spike count governs cost (Davies et al., 2018; Ostrau et al., 2022). We stop short of converting it to joules, which would need a per-operation energy model (Horowitz, 2014) and a hardware mapping we have not built. On digital simulation HH's per-neuron-step integration is substantially more expensive than LIF's, and the spike saving does not offset it.

4.3 Ablations

Table 4: Ablations, all at T=50 with a 512-unit hidden layer for 15 epochs (SHD: 400 units, up to 25 epochs, HH and LIF sharing one bias setting). All variants use full PC inference. Spiking dynamics dominate; the neuron model matters far less. Numbers differ from Table 1, which reports each dataset's tuned operating point. N-MNIST ablations use a single polarity-summed 34×34 frame (1156 inputs) encoded like a static image, not the event input of Table 1. (Analog: non-spiking perceptron units)

Dataset	**Neuron / Enc.**	**Acc.**	**F1**
MNIST	HH / Poisson	97.88	97.87
	LIF / Poisson	97.74	97.73
	Analog / Poisson	89.61	89.60
	HH / Lat.-first	97.75	97.73
	LIF / Lat.-first	97.81	97.79
F-MNIST	HH / Poisson	87.73	87.46
	LIF / Poisson	87.76	87.39
	Analog / Poisson	77.72	74.08
	HH / Lat.-first	87.64	87.40
	LIF / Lat.-first	87.80	87.56

Dataset	**Neuron / Enc.**	**Acc.**	**F1**
N-MNIST	HH / Poisson	93.61	93.53
	LIF / Poisson	93.70	93.62
	Analog / Poisson	58.41	51.08
	HH / Lat.-first	93.50	93.39
	LIF / Lat.-first	93.76	93.68
SHD	HH / Poisson	84.32	82.83
	LIF / Poisson	82.86	81.66
	Analog / Poisson	51.63	44.54
	HH / Lat.-first	86.26	85.38
	LIF / Lat.-first	87.06	85.52

Table 4 changes one design choice at a time. Removing the spikes hurts most. Replacing the HH neurons with analog units costs 8.3 points on MNIST, 10.0 on Fashion-MNIST, 32.7 on SHD and 35.2 on N-MNIST. The spiking substrate is therefore doing real work. The penalty is largest on N-MNIST and SHD, though on SHD the input is a flattened time-by-channel histogram, so this gap reflects the input distribution rather than temporal processing. Which spiking neuron we use matters far less: HH and LIF land within 0.3 points of each other on all three frame-based datasets, matching the parity in Table 2. Encoding matters little for accuracy once the input is a single frame: on MNIST, Fashion-MNIST and N-MNIST, latency-first and Poisson coding differ by at most 0.13

points. It matters for spike count. Latency-first coding makes almost every neuron fire once, while under Poisson coding HH fires 5.8–14.0% fewer spikes than LIF on these datasets. SHD is the exception: latency-first coding gains HH 1.9 points while halving its spike count. Spike counts for all variants are in Appendix H.

4.4 Few-Shot Generalisation

Table 5: Few-shot test accuracy. Bold marks the higher value in each pair. MNIST, Fashion-MNIST and N-MNIST are single runs; SHD is the mean of three repeats. The HH advantage is largest on MNIST, where it widens with k; SHD separates only at k=20. (k: number of training examples per class)

Dataset	Neuron	k=1	k=5	k=10	k=20
MNIST	HH	**0.095**	**0.320**	**0.625**	**0.785**
	LIF	0.084	0.265	0.569	0.575
Fashion-MNIST	HH	0.110	**0.502**	**0.616**	**0.687**
	LIF	**0.135**	0.397	0.482	0.644

Dataset	Neuron	k=1	k=5	k=10	k=20
N-MNIST	HH	**0.433**	0.589	0.680	**0.730**
	LIF	0.398	**0.591**	**0.681**	0.726
SHD	HH	0.208	0.452	0.582	**0.671**
	LIF	**0.237**	**0.455**	**0.600**	0.634

Under data scarcity the two neuron models can diverge. Training both from scratch on k-shot support sets (Table 5), HH leads LIF at every shot count on MNIST, and the gap grows with k: 1.1 points at k=1 and 21 points at k=20 (0.785 vs. 0.575) under identical budgets. Fashion-MNIST shows the same pattern from k=5 onward, with HH ahead by 10.5, 13.4 and 4.3 points at k=5, 10, 20, although LIF leads at k=1. On N-MNIST the two models are indistinguishable beyond k=1 (within 0.4 points), and SHD does not separate either: HH trails LIF at $k \leq 10$ and leads by 3.7 points only at k=20. We read this as a dataset-dependent advantage on static images rather than a general property of HH neurons. The evidence is limited: the vision results are single runs and SHD averages three repeats, so we treat the MNIST and Fashion-MNIST gaps as suggestive and leave a multi-seed few-shot study to future work.

5 Conclusion

We trained a spiking network of Hodgkin–Huxley neurons with predictive coding, using only local learning: no backpropagation, no surrogate gradients and no unrolling through time. The key step is simple: predictive coding operates on firing rates rather than individual spikes, so the learning rule never differentiates through the spiking dynamics. The cost of learning therefore no longer depends on the neuron model. Across four benchmarks, HH matches LIF in accuracy and usually fires fewer spikes.

The work has clear limits. The network has a single feedforward hidden layer, while predictive coding is naturally recurrent (Rao & Ballard, 1999; Friston, 2005). Only N-MNIST reaches the network as an event stream; SHD is flattened into a static histogram, so the temporal dynamics of HH are barely used. The evidence is also limited: we use five seeds and single runs in the sweeps, and the model is sensitive to gain, bias and threshold, so it needs careful tuning.

Several directions follow. Processing SHD step by step would test whether HH dynamics add real representational power. Recurrent and deeper networks would bring the model closer to standard predictive coding (Whittington & Bogacz, 2017). Simulating HH remains costly on digital hardware; cheaper neuron approximations (Izhikevich, 2004) or physical implementations (Fang et al., 2021; Beaubois et al., 2024) could close this gap, and neuromorphic hardware would turn our spike counts into measured energy (Davies et al., 2018).

Our main point is methodological. Detailed neuron models seem costly to *train* mostly because of how we train them. With a rule that does not differentiate through the dynamics, they become practical to train, and the remaining cost is that of simulation, not learning.

AI USE STATEMENT

In this work, we used generative AI tools to assist with writing formatting and editing of the manuscript. We have not used generative AI tools for research ideation, experimental design, or the generation of results, and autonomous agentic research is not applicable to this work. All experiments, analyses, and reported numbers were produced and verified by the authors. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including all text and claims.

ETHICS STATEMENT

This work uses only standard public benchmark datasets (MNIST, Fashion-MNIST, N-MNIST, SHD). SHD contains spike encodings of speech recorded from 12 speakers, who appear in the released files only as anonymised integer IDs; we use it under its public release terms. The research concerns fundamental learning mechanisms in spiking networks and we identify no specific pathway to harmful application. To the extent the work succeeds, its main societal implication is the potential for lower-energy inference on event-driven hardware.

REPRODUCIBILITY STATEMENT

We are committed to ensuring the reproducibility of our research. The complete PyTorch implementation of PCHH-SNN, including the Hodgkin–Huxley neuron model, the predictive-coding inference loop, and all training and evaluation scripts, will be made available as supplementary material. Our core method is detailed in Section 3, with the full algorithm given in Algorithm 1 and a step-by-step breakdown in the Appendix. The source code includes the exact configurations and hyperparameters used to generate every reported result, covering the classification accuracies (Table 1), the seed-controlled HH/LIF comparison (Table 2), the spike-economy analysis (Table 3), and the ablations (Table 4). All experiments use standard public benchmark datasets, and the data-processing pipelines for each are explicitly defined in our implementation and described in Appendix E. Every reported number comes from a single evaluation of the test set per configuration, using the checkpoint with the best validation score.

REFERENCES

Romain Beaubois, Jérémy Cheslet, Yoshiho Ikeuchi, Pascal Branchereau, and Timothée Levi. Real-time multicompartment hodgkin-huxley neuron emulation on soc fpga. *Frontiers in Neuroscience*, 18:1457774, 2024. doi: 10.3389/fnins.2024.1457774.

Sander M Bohte, Joost N Kok, and Han La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002. doi: 10.1016/S0925-2312(01)00658-0.

Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. The Heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2744–2757, 2022. doi: 10.1109/TNNLS.2020.3044364.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, and Shweta Jain. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359.

Xiaoyan Fang, Shukai Duan, and Lidan Wang. Memristive hodgkin-huxley spiking neuron model for reproducing neuron behaviors. *Frontiers in Neuroscience*, 15:730566, 2021. doi: 10.3389/fnins.2021.730566.

Mario De Florio, Adar Kahana, and George Em Karniadakis. Analysis of biologically plausible neuron models for regression with spiking neural networks, 2023. URL `https://arxiv.org/abs/2401.00369`.

Karl Friston. A theory of cortical responses. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1456):815–836, 2005. doi: 10.1098/rstb.2005.1622.

Wulfram Gerstner and Werner M Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002. doi: 10.1017/CBO9780511815706.

Mohammadnavid Ghader, Saeed Reza Kheradpisheh, Bahar Farahani, and Mahmood Fazlali. Backpropagation-free spiking neural networks with the forward–forward algorithm. *Scientific Reports*, 2026. doi: 10.1038/s41598-026-41671-4.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

Geoffrey E Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544, 1952. doi: 10.1113/jphysiol.1952.sp004764.

Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14. IEEE, 2014. doi: 10.1109/ISSCC.2014.6757323.

Eugene M Izhikevich. Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070, 2004. doi: 10.1109/TNN.2004.832719.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14:119, 2020. doi: 10.3389/fnins.2020.00119.

Kwangjun Lee, Shirin Dora, Jorge F Mejias, Sander M Bohte, and Cyriel M A Pennartz. Predictive coding with spiking neurons and feedforward gist signaling. *Frontiers in Computational Neuroscience*, 18:1338280, 2024. doi: 10.3389/fncom.2024.1338280.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:13276, 2016. doi: 10.1038/ncomms13276.

Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020. doi: 10.1038/s41583-020-0277-3.

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

Antony W. N'dri, Thomas Barbier, Céline Teulière, and Jochen Triesch. Predictive coding light. *Nature Communications*, 16:8880, 2025. doi: 10.1038/s41467-025-64234-z.

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

Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9:437, 2015. doi: 10.3389/fnins.2015.00437.

Alexander G. Ororbia. Contrastive signal-dependent plasticity: Self-supervised learning in spiking neural circuits. *Science Advances*, 10(43):eadn6076, 2024.

Christoph Ostrau, Christian Klarhorst, Michael Thies, and Ulrich Rückert. Benchmarking neuromorphic hardware and its energy expenditure. *Frontiers in Neuroscience*, 16:873935, 2022. doi: 10.3389/fnins.2022.873935.

Nicolas Perez-Nieves and Dan F. M. Goodman. Sparse spiking gradient descent. In *Advances in Neural Information Processing Systems*, volume 34, pp. 11795–11808, 2021.

Rajesh P N Rao and Dana H Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999. doi: 10.1038/4580.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017. doi: 10.3389/fnins.2017.00682.

João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in Neural Information Processing Systems*, 31, 2018.

Erik B. Terres-Escudero, Javier Del Ser, and Pablo García Bringas. Emerging neohebbian dynamics in forward-forward learning: Implications for neuromorphic computing. In Mattia Cerrato, Danguolė Kalinauskaitė, Mantas Lukoševičius, Mykola Pechenizkiy, and Kristina Šutienė (eds.), *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 151–165, Cham, 2026. Springer Nature Switzerland. ISBN 978-3-032-25311-8.

Haidong Wang, Xiaogang Xiong, Mengting Lan, Yinghao Chu, Zixuan Jiang, KC Santosh, Shimin Wang, and Renxin Zhong. Pc-snn: Predictive coding-based local hebbian plasticity learning in spiking neural networks. *Neurocomputing*, 683:133421, 2026. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2026.133421. URL https://www.sciencedirect.com/science/article/pii/S0925231226008180.

James C R Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural Computation*, 29(5): 1229–1262, 2017. doi: 10.1162/NECO_a_00949.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:331, 2018. doi: 10.3389/fnins.2018.00331.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.

Bojian Yin, Federico Corradi, and Sander M. Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10): 905–913, 2021. doi: 10.1038/s42256-021-00397-w.

A TABLE OF NOTATION

Table 6 lists the symbols used in the paper. Bold lowercase letters denote vectors, bold uppercase letters denote matrices, and superscript (l) indexes layers.

Table 6: Summary of notation.

Symbol	Meaning
Network and data	
L	Number of synaptic layers
D_l, D_{in}	Width of layer l; input dimension
H, C	Hidden layer size; number of classes
B	Batch size
$\boldsymbol{X}$, $\boldsymbol{Y}$	Input batch; target batch (one-hot or soft)
$\boldsymbol{W}^{(l)}$, $\boldsymbol{b}^{(l)}$	Weights and biases of layer l
k	Shots per class in few-shot training
Spiking forward pass	
T	Number of simulation steps
t	Simulation step index
$\boldsymbol{s}^{(l)}(t)$	Binary spike vector of layer l at step t
$\boldsymbol{S}_l$	Spike counter of layer l, summed over T steps
α, p_i	Poisson scale; per-step firing probability of input i
$\boldsymbol{u}^{(l)}(t)$	Synaptic drive, $\boldsymbol{W}^{(l)}\boldsymbol{s}^{(l-1)}(t) + \boldsymbol{b}^{(l)}$
$I^{(l)}(t)$	Injected current
g_l, I_l^{bias}	Current gain; bias current
Hodgkin–Huxley neuron	
V, C_m	Membrane voltage (mV); membrane capacitance
m, n, h	Gating variables in $[0, 1]$
$\alpha_q(V)$, $\beta_q(V)$	Opening and closing rates of gate $q \in \{m, n, h\}$
$q_\infty(V)$	Steady-state value of gate q
I_{Na}, I_{K}, I_{L}	Sodium, potassium and leak currents
$\bar{g}_{\text{Na}}$, $\bar{g}_{\text{K}}$, $\bar{g}_{\text{L}}$	Maximal conductances
E_{Na}, E_{K}, E_{L}	Reversal potentials
Δt	Euler integration step (ms)
θ	Spike threshold (mV)
τ_{ref}, $r(t)$	Refractory period; refractory counter
$V_{\max}$	Voltage saturation bound
Predictive coding	
$\boldsymbol{x}_{\text{proxy}}^{(l)}$	Rate proxy, spike count of layer l divided by T
$\boldsymbol{x}^{(0)}$	Input state, held fixed during inference
$\boldsymbol{x}^{(l)}$	PC latent state of layer l
$f(\cdot)$, $f'(\cdot)$	Prediction nonlinearity, $\text{clip}(z, 0, 1)$, and its derivative
$\boldsymbol{z}^{(l)}$	Pre-activation prediction, $\boldsymbol{W}^{(l)}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}$
$\hat{\boldsymbol{x}}^{(l)}$	Prediction of layer l, $f(\boldsymbol{z}^{(l)})$
$\boldsymbol{e}^{(l)}$	Prediction error, $\boldsymbol{x}^{(l)} - \hat{\boldsymbol{x}}^{(l)}$
E	PC energy, $\frac{1}{2}\sum_l \|\boldsymbol{e}^{(l)}\|_2^2$
T_{infer}, i	Inference iterations; inference iteration index
γ_x	Inference step size
$\boldsymbol{\delta}^{(l)}$	Error-modulated signal, $\boldsymbol{e}^{(l)} \odot f'(\boldsymbol{z}^{(l)})$
κ	Gradient-norm clipping threshold
$\odot$	Element-wise product

B HYPERPARAMETERS

Table 7: Hyperparameters of the operating-point runs (Table 1). Values are shared across datasets unless an override is noted.

Component	Parameter	Value
Architecture	Hidden size H	512 (SHD: 400)
	Input size D_{in}	784 (N-MNIST: 2312; SHD: 7000)
	Synaptic layers L	2
	Weight / bias init	Xavier uniform, gain 0.5 / 0
HH neurons	Simulation steps T	60 (N-MNIST, SHD: 50)
	Integration step Δt	0.03 ms
	Spike threshold θ	1.2 mV (SHD: 0.8 mV)
	Refractory time τ_{ref}	2 ms, 67 steps (SHD: 0.15 ms, 5 steps)
	Reset potential V_{reset}	0 mV
	Current gain g_l	30
	Bias current I_l^{bias}	2 (SHD: 1.5 hidden, 3.0 output)
	Voltage saturation $V_{\max}$	30 mV
	Voltage clamp (rate functions)	$[-100, 100]$ mV
	C_m; $\bar{g}_{\text{Na}}, \bar{g}_{\text{K}}, \bar{g}_{\text{L}}$	1; 120, 36, 0.3
	$E_{\text{Na}}, E_{\text{K}}, E_{\text{L}}$	115, -12, 10.6
Input encoding	MNIST, Fashion-MNIST, SHD	Poisson, α=1.0
	N-MNIST	raw events, 50 bins, denoise filter 10 ms
PC inference	Nonlinearity f	clip$(z, 0, 1)$
	Inference steps (train / eval)	100 / 50
	Inference step size γ_x	0.05
Optimiser (Adam)	Learning rate	2×10^{-4}
	Weight decay	1×10^{-4} (SHD: 2×10^{-4})
	Gradient clip norm κ	1.0
Training	Batch size B	128 (N-MNIST: 64)
	Max epochs	30 (SHD: 40)
	Early stopping	patience 4 on val. accuracy (SHD: 6 on val. macro-F1)
	Early-stopping tolerance	1×10^{-4}
	PC target	one-hot (SHD: 0.9 true class, $0.1/C$ others)
	Augmentation (SHD only)	time shift ± 4 bins, channel dropout 0.15
	Validation split	10% of train (SHD: 2 held-out speakers)
	Seed	42 (SHD: 123)

C INPUT ENCODING PIPELINE

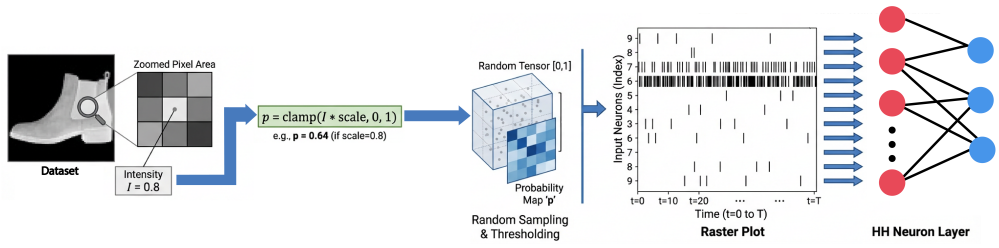


Figure 2: Poisson rate coding. Pixel intensity I maps to a per-step firing probability $p = \text{clamp}(I\alpha, 0, 1)$; independent Bernoulli draws at each of the T steps produce the binary spike raster fed into the first HH layer.

D HODGKIN–HUXLEY RATE FUNCTIONS

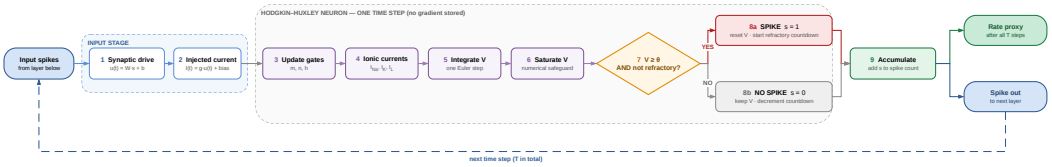


Figure 3: One HH time step, left to right. The synaptic drive $\boldsymbol{u}^{(l)}(t)$ sets the injected current $I^{(l)}(t)$ (steps 1–2). Inside the neuron, the gating variables m, n, h update first and the ionic currents follow from them; the membrane voltage is then integrated by one forward Euler step and saturated (steps 3–6). A spike is emitted only if the voltage is past threshold *and* the neuron is not refractory (step 7): on a spike the voltage resets and the refractory counter is set, otherwise the new voltage is kept and the counter decrements (step 8). Either way the spike variable is added to the running count that becomes the rate proxy (step 9), and the loop repeats for T steps.

The gating variables m, n, h follow $\dot{q} = \alpha_q(V)(1-q) - \beta_q(V)q$ with the standard HH rate functions, in the original HH convention where V is displacement from rest in mV:

$$\alpha_n(V) = \frac{0.01\,(10 - V)}{\exp\!\left(\frac{10-V}{10}\right) - 1}, \qquad \beta_n(V) = 0.125\,\exp\!\left(\frac{-V}{80}\right), \tag{6}$$

$$\alpha_m(V) = \frac{0.1\,(25 - V)}{\exp\!\left(\frac{25-V}{10}\right) - 1}, \qquad \beta_m(V) = 4\,\exp\!\left(\frac{-V}{18}\right), \tag{7}$$

$$\alpha_h(V) = 0.07\,\exp\!\left(\frac{-V}{20}\right), \qquad \beta_h(V) = \frac{1}{\exp\!\left(\frac{30-V}{10}\right) + 1}. \tag{8}$$

Gating variables are initialised at their steady-state values $q_\infty(V) = \alpha_q(V)/(\alpha_q(V) + \beta_q(V))$ evaluated at $V_0{=}0$, and clipped to $[0, 1]$ after each Euler step. α_n and α_m have removable singularities at $V{=}10$ and $V{=}25$ mV. The implementation does not substitute the limits 0.1 and 1.0; it adds 10^{-8} to both denominators. In 32-bit floats this returns 0 at the exact singular voltage and deviates from the limit only within about 10^{-3} mV of it, a region the continuous membrane voltage almost never occupies. The clamp to $[-100, 100]$ mV keeps the exponentials finite.

For reference, the latency-first encoding used in the ablations fires each input unit exactly once, at step $t_i = \max\!\big(1, \lceil T(1 - x_{0,i})\rceil\big)$, so latency falls linearly with intensity: the brightest inputs fire at the first step and zero-intensity inputs fire at the last, and the entire input is delivered within one pass of the simulation window.

E DATASETS, PREPROCESSING, AND IMPLEMENTATION

Table 8: Dataset splits and input dimensionality. The 90/10 train/validation division uses a fixed seed; the test split is the standard one for each benchmark and is evaluated exactly once per configuration. SHD's validation set instead holds out two of the ten training speakers; its official test split includes two speakers never seen in training.

Dataset	**Train**	**Val.**	**Test**	**Classes**	D_{in}
MNIST	54,000	6,000	10,000	10	784
Fashion-MNIST	54,000	6,000	10,000	10	784
N-MNIST	54,000	6,000	10,000	10	2312
SHD	6,796	1,360	2,264	20	7000

Preprocessing. Pixel values are scaled to $[0, 1]$ and flattened, $\boldsymbol{x}_0 = \mathrm{vec}(\boldsymbol{X}/255)$. N-MNIST events are filtered with `tonic`'s `Denoise` transform (10 ms) and binned with `ToFrame` into 50

time bins per polarity; any non-zero bin becomes a spike, giving a binary 2312×50 train fed directly to the first HH layer, and its time average serves as $\boldsymbol{x}^{(0)}$. For the ablations (Table 4) the two polarities are instead accumulated over the full window, summed, and min–max normalised into one 34×34 frame.

SHD. Each clip's spikes on the 700 cochlear channels are sum-pooled by 5 to 140 channels and binned into 50 bins over the first second (later events fall into the last bin). Counts are compressed with $\log(1+x)$ and divided by the per-sample maximum, giving a 7000-dimensional vector in $[0, 1]$ that is Poisson-encoded exactly like pixel intensities. Validation speakers are drawn with a fixed seed from the ten training speakers, so no validation speaker contributes training clips. During training only, each clip is circularly shifted in time by up to ±4 bins and 15% of its pooled channels are zeroed.

F Macro-F1 at the Operating Point

Table 9 reports macro-F1 alongside accuracy for the operating-point runs of Table 1. Macro-F1 averages the F1 score over classes with equal weight, so it drops below accuracy when errors concentrate in a few classes. The gap is at most 0.26 points on every dataset, so predictions are balanced across classes and no class is systematically sacrificed.

Table 9: Test accuracy and macro-F1 of PCHH-SNN at each dataset's operating point.

Dataset	Accuracy (%)	Macro-F1 (%)	Gap
MNIST	97.72	97.71	0.01
Fashion-MNIST	88.08	87.88	0.20
N-MNIST	93.29	93.18	0.11
SHD	87.68	87.42	0.26

G Per-Seed Results and Significance Tests

Table 10 gives the individual seed values summarised in Table 2, together with macro-F1 and its paired t-test. Standard deviations are sample standard deviations (ddof=1). Two observations that the aggregate hides. On Fashion-MNIST the accuracy difference is nominally significant (p=0.028) but the F1 difference is not (p=0.116), so the effect is weaker than the accuracy test alone suggests; it is 0.3 points and does not survive Bonferroni correction across four datasets (α=0.0125). We report it and do not lean on it. On N-MNIST, seed 1011 is an outlier in HH's disfavour (0.8450 against a 0.9305–0.9375 range across the other four seeds), coinciding with a PC objective an order of magnitude above them, the signature of an initialisation-induced instability rather than a neuron-model difference. That single seed accounts for almost all of the N-MNIST mean gap and for the F1 standard deviation of 0.0585, the largest dispersion anywhere in our results. On SHD the accuracy gap is $+0.15$ points (p=0.859; Wilcoxon p=0.813) with HH ahead on three of five seeds, while HH is lower on all five seeds in both spike rate (0.0575 vs. 0.0696, paired t, p=0.001) and converged PC objective (paired t, p=0.042). The SHD seed means sit below Table 1's single best run (87.68%); the seed study also holds out a different validation speaker pair, which leaves 332 fewer training clips.

Table 10: Per-seed test accuracy, macro-F1, and converged PC objective for HH+PC and LIF+PC at T=25 (SHD: T=50). Paired t-tests over the five seeds; p-values in the final rows.

Dataset	Seed	HH Acc.	LIF Acc.	HH F1	LIF F1	HH PC obj.	LIF PC obj.
MNIST	42	0.9753	0.9755	0.9752	0.9754	1.61×10^{-4}	1.68×10^{-4}
	123	0.9748	0.9741	0.9747	0.9740	1.43×10^{-4}	2.21×10^{-4}
	456	0.9738	0.9747	0.9737	0.9746	1.54×10^{-4}	1.57×10^{-4}
	789	0.9755	0.9731	0.9754	0.9730	1.45×10^{-4}	2.84×10^{-4}
	1011	0.9749	0.9744	0.9748	0.9743	1.72×10^{-4}	1.68×10^{-4}
	Mean	0.9749	0.9744	0.9747	0.9742	1.55×10^{-4}	2.00×10^{-4}
	Std .0007/.0009/.0007/.0009; Acc. t=+0.91, p=0.417; F1 t=+0.87, p=0.435						
F-MNIST	42	0.8767	0.8724	0.8758	0.8703	2.62×10^{-4}	4.53×10^{-4}
	123	0.8765	0.8749	0.8746	0.8738	3.42×10^{-4}	2.93×10^{-4}
	456	0.8757	0.8697	0.8732	0.8672	3.21×10^{-4}	4.76×10^{-4}
	789	0.8685	0.8663	0.8656	0.8630	6.33×10^{-4}	5.17×10^{-4}
	1011	0.8777	0.8765	0.8742	0.8753	3.06×10^{-4}	2.52×10^{-4}
	Mean	0.8750	0.8720	0.8727	0.8699	3.73×10^{-4}	3.98×10^{-4}
	Std .0037/.0041/.0041/.0050; Acc. t=+3.37, p=**0.028**; F1 t=+2.00, p=0.116						
N-MNIST	42	0.9305	0.9319	0.9296	0.9309	4.0×10^{-4}	3.7×10^{-4}
	123	0.9313	0.9350	0.9304	0.9341	5.4×10^{-4}	5.6×10^{-4}
	456	0.9345	0.9333	0.9335	0.9323	4.5×10^{-4}	5.5×10^{-4}
	789	0.9375	0.9371	0.9367	0.9363	4.1×10^{-4}	5.6×10^{-4}
	1011	0.8450	0.9351	0.8019	0.9342	5.65×10^{-3}	4.9×10^{-4}
	Mean	0.9158	0.9345	0.9064	0.9336	1.49×10^{-3}	5.1×10^{-4}
	Std .0397/.0020/.0585/.0021; Acc. t=−1.05, p=0.354; F1 t=−1.03, p=0.360						
SHD	42	0.8313	0.8339	0.8284	0.8169	1.46×10^{-3}	2.80×10^{-3}
	123	0.8352	0.8313	0.8312	0.8216	1.51×10^{-3}	2.92×10^{-3}
	456	0.8701	0.8511	0.8648	0.8383	1.34×10^{-3}	1.79×10^{-3}
	789	0.8472	0.8335	0.8429	0.8292	1.01×10^{-3}	1.58×10^{-3}
	1011	0.8140	0.8405	0.8056	0.8337	1.75×10^{-3}	1.82×10^{-3}
	Mean	0.8396	0.8381	0.8346	0.8279	1.42×10^{-3}	2.18×10^{-3}
	Std .0208/.0081/.0216/.0087; Acc. t=+0.19, p=0.859; F1 t=+0.73, p=0.507						

H Extended Ablations

Table 11 adds spike counts to the ablations of Table 4 for all four datasets. MNIST, Fashion-MNIST and N-MNIST use T=50, H=512 and 15 epochs (522 spiking neurons). SHD uses T=50, H=400 and up to 25 epochs, with HH and LIF sharing one bias setting.

Neuron model. Under Poisson coding HH fires fewer spikes than LIF at matched accuracy on every dataset: 5.8% fewer on MNIST, 14.0% on Fashion-MNIST, 8.7% on N-MNIST and 37.7% on SHD. With a shared bias on SHD, LIF fires 60% more than HH (0.091 vs. 0.057), against 21% more under the calibrated bias of Table 2.

Encoding. On the three frame-based datasets, latency-first coding makes almost every neuron fire once. Spike counts reach 503–522 of a possible 522 for both models, and the HH–LIF gap shrinks below 3%. Accuracy is unchanged, so Poisson coding reaches the same accuracy with fewer spikes: less than half on MNIST and Fashion-MNIST, and 78% on N-MNIST. SHD behaves differently. There, latency-first coding raises HH accuracy by 1.9 points and halves its spike count (1196 to 621), while LIF still fires 2.7 times more than HH.

SHD-specific factors. Table 12 changes one factor at a time from the HH+PC base on SHD. PC settling matters more on SHD than on the vision datasets: reading the class from the feedforward rate proxy costs HH 11.0 points and LIF 29.2. Soft targets and augmentation are both needed, worth

2.2 and 4.2 points. The refractory period barely affects accuracy but dominates spike count. At τ_{ref}=2 ms (67 steps, longer than the window) each neuron fires at most once, accuracy is unchanged, and HH emits 8.5 times fewer spikes. The two base rows use identical configurations and differ by 2.3 points, which sets the resolution of any single-run SHD comparison.

Table 11: Ablations with spike counts on all four datasets. SPS: mean spikes per sample; SR: mean spike rate. n/a marks quantities undefined for analog units. N-MNIST uses a single polarity-summed frame (1156 inputs).

Dataset	Neuron	Encoding	Acc. (%)	F1 (%)	SPS	SR
MNIST	HH	Poisson	**97.88**	**97.87**	217.8	0.0083
	LIF	Poisson	97.74	97.73	231.3	0.0089
	Analog	Poisson	89.61	89.60	n/a	n/a
	HH	Latency-first	97.75	97.73	518.0	0.0198
	LIF	Latency-first	97.81	97.79	519.8	0.0199
F-MNIST	HH	Poisson	87.73	87.46	206.0	0.0079
	LIF	Poisson	87.76	87.39	239.5	0.0092
	Analog	Poisson	77.72	74.08	n/a	n/a
	HH	Latency-first	87.64	87.40	520.7	0.0199
	LIF	Latency-first	**87.80**	**87.56**	522.0	0.0200
N-MNIST	HH	Poisson	93.61	93.53	394.5	0.0151
	LIF	Poisson	93.70	93.62	432.1	0.0166
	Analog	Poisson	58.41	51.08	n/a	n/a
	HH	Latency-first	93.50	93.39	503.3	0.0193
	LIF	Latency-first	**93.76**	**93.68**	518.3	0.0199
SHD	HH	Poisson	84.32	82.83	1196.3	0.0570
	LIF	Poisson	82.86	81.66	1918.9	0.0914
	Analog	Poisson	51.63	44.54	n/a	n/a
	HH	Latency-first	86.26	85.38	620.9	0.0296
	LIF	Latency-first	**87.06**	**85.52**	1696.3	0.0808

Table 12: SHD-specific factors at T=50, H=400, up to 25 epochs. FF: class read from the feedforward rate proxy without PC settling. Rows below the rule each change one factor from the HH+PC base.

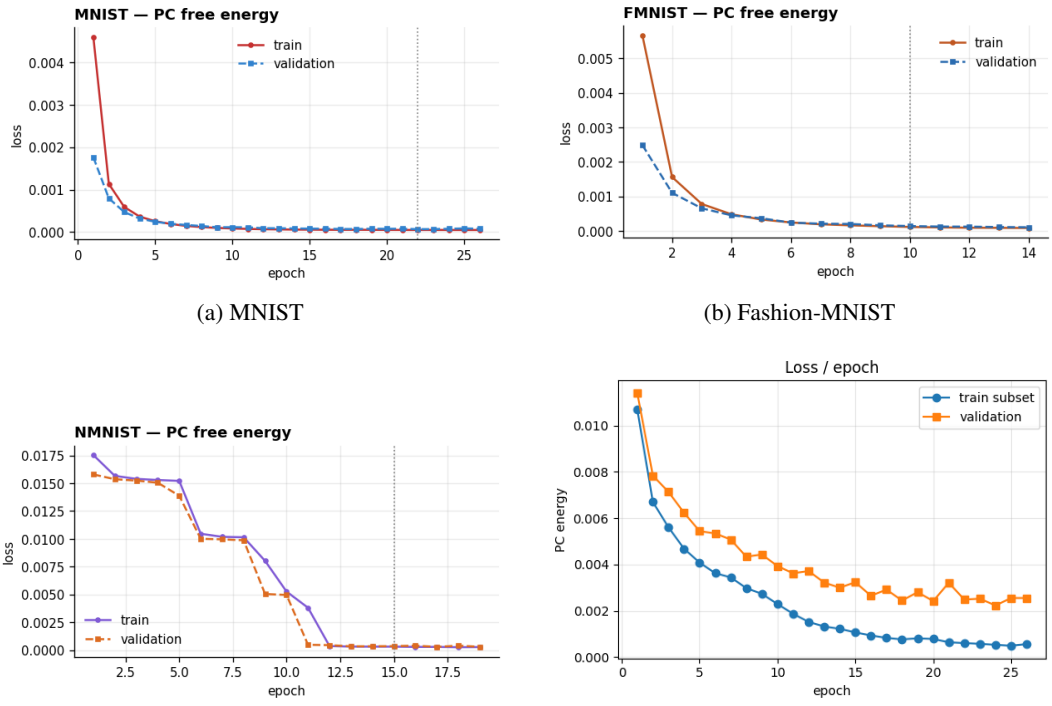
Variant	Acc. (%)	F1 (%)	AUC	SR	SPS
HH, Poisson, PC	84.32	82.83	0.9827	0.0570	1196.3
HH, Poisson, FF	73.37	70.28	0.9560	0.0559	1173.4
LIF, Poisson, PC	82.86	81.66	0.9814	0.0914	1918.9
LIF, Poisson, FF	53.67	53.07	0.9534	0.0921	1934.9
HH base (repeat run)	**86.66**	86.08	0.9898	0.0559	1173.9
hard one-hot targets	84.50	83.72	0.9892	0.0581	1220.0
no augmentation	82.51	82.24	0.9823	0.0573	1204.0
τ_{ref}=2 ms	86.62	**86.33**	**0.9910**	0.0065	137.5
uniform I^{bias}=2	85.42	85.46	0.9894	0.0658	1382.3

I Convergence of the Predictive-Coding Energy

Figure 4 shows the PC energy (3) over training on the three static datasets (MNIST, Fashion-MNIST, N-MNIST) and on SHD, respectively. On the static datasets, the energy falls steeply over the first five to ten epochs and then holds near a floor, with train and validation curves tracking each other closely throughout; the vertical dotted line in each panel marks the epoch selected by early stopping. This is the check that the objective is doing what the derivation claims: learning without a global

gradient still drives prediction error down monotonically, and the near-absence of a train/validation gap on these three datasets indicates the settling loop is not memorising the training set.

SHD behaves differently. Train and validation energy fall together for roughly the first five epochs, after which validation plateaus near 0.0025–0.003 while train continues decreasing toward the floor, leaving a persistent gap that does not close over the remaining epochs. We read this as evidence of a looser fit on the higher-dimensional, smaller SHD dataset rather than a failure of the local rule to converge; training energy still decreases monotonically throughout, which is the property we make claims about. We make no formal convergence claim in either case: the local updates approximate backpropagation only under conditions we do not verify here (Whittington & Bogacz, 2017), and the evidence for convergence in our setting is empirical.



(a) MNIST

(b) Fashion-MNIST

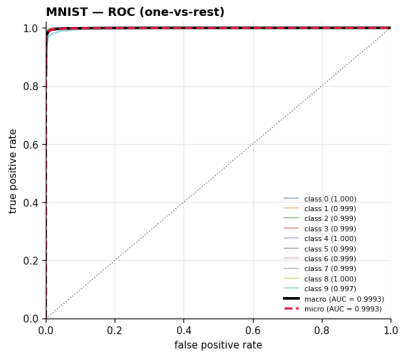
(c) N-MNIST

(d) SHD

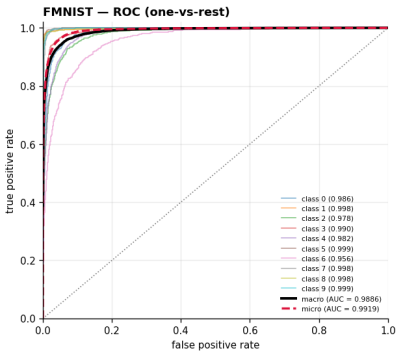
Figure 4: Train and validation PC energy, up to early stopping (dotted line marks the selected epoch). The local objective converges on all four datasets; on SHD validation plateaus while train continues to fall.

J Discriminability: ROC–AUC

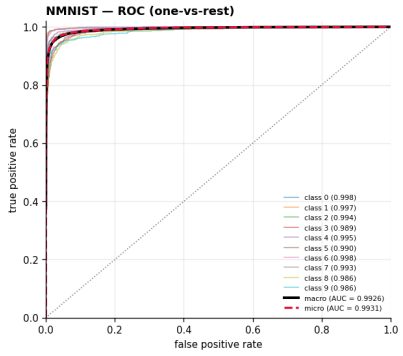
Figure 5 reports one-vs-rest ROC curves computed from the PC readout's output scores on the test set. Micro-average AUC is 0.9993 (MNIST), 0.9919 (Fashion-MNIST), and 0.9931 (N-MNIST); per-class AUCs are uniformly high on MNIST (all classes $\geq$0.997) and more variable on Fashion-MNIST, where class 6 (shirt) is the weakest at 0.956, consistent with its known visual confusability with classes 0 (T-shirt) and 2 (pullover) in the raw pixel space. On SHD we show only the macro-average curve for legibility across 20 classes; the macro-average AUC is 0.991 (Figure 5d), so the PC readout remains highly discriminative on SHD as well.



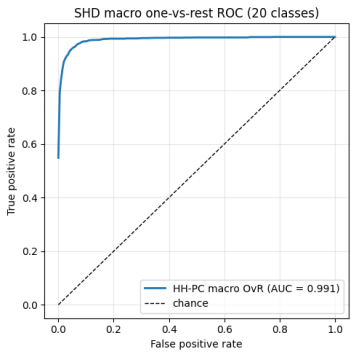
(a) MNIST



(b) Fashion-MNIST



(c) N-MNIST



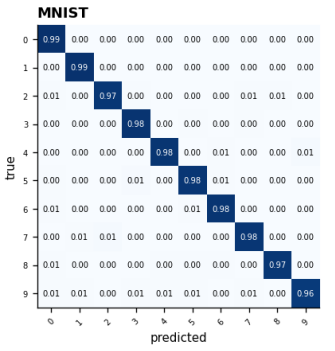
(d) SHD

Figure 5: One-vs-rest ROC curves of the PC readout on the test set. Per-class curves are shown alongside macro- and micro-averaged curves; the diagonal marks chance level. SHD shows the macro-average only, for legibility across 20 classes.

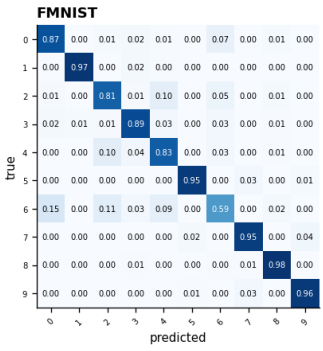
K Per-Class Error Structure

Figure 6 shows row-normalised confusion matrices (diagonal = per-class recall) on the test set. MNIST and N-MNIST are close to diagonal throughout, with per-class recall never dropping below 0.81 (N-MNIST, class 5) and 0.96 (MNIST, class 9). Fashion-MNIST is markedly less clean: class 6 (shirt) is recalled only 59% of the time and is confused with classes 0 (T-shirt, 15%), 2 (pullover, 11%), and 4 (coat, 9%), the classic Fashion-MNIST failure mode in which visually similar upper-body garment classes overlap in the learned representation.

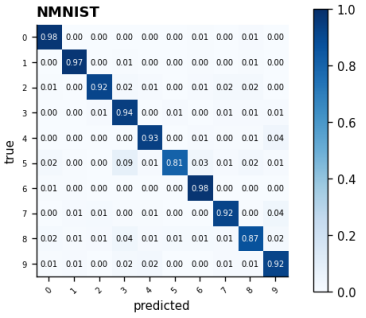
On SHD (Figure 6d, PC readout), most digit classes are recalled above 0.9 in both languages, but a few pairs stand out. English “one” is recalled only 58% of the time and is confused with German “neun” (nine) in 39% of trials; conversely, “neun” is misclassified as “one” in 25% of trials. English “three” is confused with German “fünf” (five, 16%), and German “null” (zero) is confused with German “vier” (four, 14%). Most of these confusions pair acoustically similar words across the two languages, which suggests they arise from the flattened time-by-channel input rather than from the learning rule.



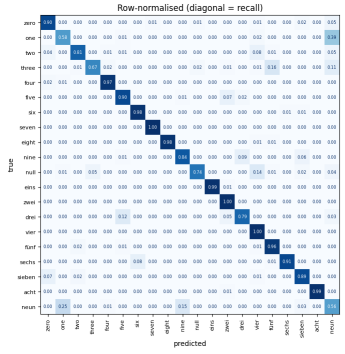
(a) MNIST



(b) Fashion-MNIST



(c) N-MNIST



(d) SHD

Figure 6: Row-normalised confusion matrices (diagonal = per-class recall) of the PC readout on the test set. SHD classes 0–9 are English, 10–19 German.