Predictive Suppression Layers for Communication-Efficient Spiking Neural Networks

Aidin Attar
University of Padova
Department of Information Engineering
Padova, Italy
aidin.attar@phd.unipd.it

Michele Rossi
University of Padova
Dept. of Inf. Eng. and Dept. of Mathematics
Padova, Italy
michele.rossi@unipd.it

Abstract

Feedforward Spiking Neural Networks (SNNs) typically propagate every generated spike indiscriminately, disregarding whether the information is redundant from an information-theoretic perspective. This lack of selectivity induces high redundancy in inter-layer communication, creating an expensive overhead, e.g., in scenarios involving many-core neuromorphic hardware or communication-dominated Internet-of-Things (IoT) where features are transmitted wirelessly. To address this challenge, we trade localized processing for leaner network channels by introducing a minimal predictive coding framework for SNNs. We propose two layer variants sharing a predictor block: *error units*, which transmit signed spiking residuals, and *predictive suppression*, which uses residual magnitude to dynamically gate and forward only unpredictable, "surprising" activity. Evaluated on the N-MNIST and Spiking Heidelberg Digits (SHD) datasets using diagnostic metrics that decouple local processing from cross-layer communication, our new predictive coding layers achieve significant communication savings. Numerical results reveal a three-fold reduction in communicated activity, while increasing the task accuracy for both datasets. The latter finding is notable, and suggests that predictive coding layers not only minimize communication overhead, but also produce output feature vectors with a higher representation power.

CCS Concepts

• **Computing methodologies** → **Neural networks**; • **Cross-computing tools and techniques** → *Design*; • **Computer systems organization** → *Embedded and cyber-physical systems.*

Keywords

spiking neural networks, predictive coding, communication efficiency, neuromorphic computing, event-based sensing

1 Introduction

Spiking neural networks are a class of artificial neural networks that closely mimic the behavior of the brain. Their constituting neurons exchange information through sparse, discrete spikes instead of dense activations, mapping naturally onto neuromorphic hardware where activity, not idle time, entails energy consumption. This makes the *content* and *timing* of in-network communication central to their design. Yet, in a standard feedforward SNN every spike a layer produces is propagated forward indiscriminately, with no notion of whether that spike carries new information or merely repeats what the next layer could already have anticipated. The total spike count metric, commonly used to gauge energy efficiency, does not capture this distinction. Total spike count aggregates two qualitatively different costs: the *local* processing a layer performs internally, and the activity it *communicates* to downstream layers. On hardware where inter-core communication dominates the energy budget, these costs *are not interchangeable*, and a metric that conflates them may be inapt to come up with efficient designs. This same asymmetry appears in signal-processing and transmission in edge-systems: moving spike activity between devices is often *substantially more expensive* than performing local processing on a given device. In IoT, communication accounts for most of the energy consumed by IoT nodes and usually dominates local processing [8]. Under such communication-dominated regimes, we seek a mechanism that at the cost of an increased local processing leads to a leaner channel.

Predictive coding offers a useful design principle to control the local processing *vs* communication tradeoff. In its classical form, a cortical area transmits the part of its input that a higher area cannot predict, in place of the full input [5, 15]. Interpreted as a communication rule, this suggests forwarding "surprise" rather than the original signal. Translating this principle into an SNN, however, comes at a price. In fact, to know what is predictable, a layer must be in the position of computing a prediction, and this entails an increase in its local processing. Predictive coding therefore does not reduce the total number of spikes but rather increases it. The relevant question is whether local predictive computation (the price to pay) can be leveraged for a *more selective and informative* inter-layer (or inter-device, e.g., in an IoT setting) channel. In this paper, we propose minimal predictive coding as the mechanism to control this tradeoff. According to it, each layer locally predicts its own input and uses the resulting prediction error to gate the latent spikes it sends to the next layer. As a result of this gating mechanism, predictable spikes are suppressed, by only letting those carrying new information through. Fig. 1 contrasts the proposed predictive design with a plain feedforward baseline.

Specifically, we propose two minimal predictive coding variants. The first one, referred to as "error unit", forwards prediction errors in place of the original latent vector produced by a layer. The second variant, referred to as "predictive suppression", modulates the output latent vector via a gating mechanism driven by the error magnitude. Our numerical results confirm that both approaches increase the total spike count of a layer. However, the number of spikes that are communicated to the next layer decreases substantially, leading to a leaner channel. On N-MNIST [13], error unit layers improve classification accuracy over a matched feedforward baseline by concentrating communication resources on high-error (unpredictable) inputs. Predictive suppression provides a slightly

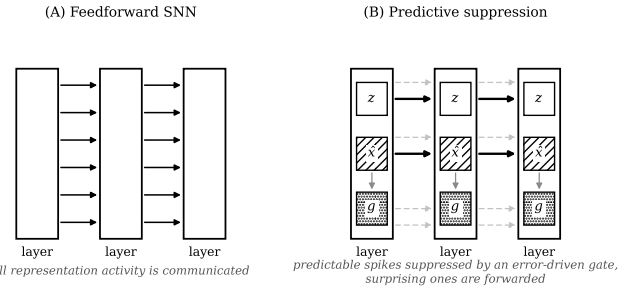

Figure 1: Predictive coding layer. (A) Plain feedforward SNN forwards the entire representation to the next layer. (B) Predictive suppression keeps the layers spiking but adds a local error-driven gate that suppresses predictable spikes and forwards the surprising ones, giving less communicated activity.

higher performance, still beating a non-predictive SNN baseline. A layer-wise linear-probe analysis shows that its compressed message remains as decodable as the full representation; a large share of the removable redundancy appears at the first communicated layer, which carries about 7× less weighted activity than the SNN baseline. On the more challenging SHD [2] dataset, predictive suppression layers still shine, by improving over the matched SNN baseline, whereas error units underperform. In general, predictive suppression is found to be the best design, delivering more stable performance across the considered datasets.

To summarize, in this work we introduce two minimal predictive coding layer designs for SNNs, performing a communication-oriented evaluation that separates local layer processing from inter-layer communication costs. The proposed predictive layers are evaluated on N-MNIST and SHD datasets, demonstrating their effectiveness in delivering surprise-selective, low-redundancy communication channels while retaining competitive task accuracy and signal reconstruction performance at a decoder. Typical results provide a three-fold reduction in communication activity, while the task accuracy increases for both datasets, suggesting that our designs not only minimize communication overhead, but also produce output feature vectors with a higher representation power.

2 Related work

Predictive coding was originally proposed as a theory of cortical processing where higher areas predict lower-level activity and only the unpredicted residual is propagated forward [15]. Rao and Ballard [15] used this principle to account for extra-classical receptive-field effects in the visual cortex, framing prediction errors as the informative signals exchanged between hierarchical areas. Friston [5] later embedded the same intuition in the free-energy principle, where perception and action are cast as processes that minimise prediction error under a generative model. More recently, predictive-coding networks have been studied as machine-learning models. Whittington and Bogacz [16] showed that local predictive-coding dynamics can approximate error backpropagation under suitable assumptions, while Millidge et al. [9] reviewed predictive coding as a possible route toward learning systems that are less dependent on backpropagation. In the present work we take a narrower interpretation: prediction error is used as a communication signal that gates what a spiking layer forwards, without involving iterative top-down inference or a new learning rule.

Deep spiking neural networks are commonly trained with surrogate gradients, which replace the non-differentiable derivative of the spike function with a smooth proxy during backpropagation. *SuperSpike* introduced one such surrogate-gradient formulation for multilayer SNNs [17], and Neftci et al. [10] surveyed how surrogate gradients make gradient-based optimisation practical for spiking models. Bellec et al. [1] addressed a related credit-assignment problem in recurrent SNNs with eligibility propagation, showing how local eligibility traces can support learning in temporal spiking networks. These works are primarily concerned with biologically plausible learning and credit assignment in spiking networks. Here, we use the supervised surrogate-gradient setting pragmatically, with the goal of designing and evaluating a communication-efficient predictive layer for edge-oriented SNNs.

A separate line of work studies efficiency in SNNs and neuromorphic systems. Neuromorphic processors such as INTEL's Loihi are designed to exploit sparse event-driven activity and on-chip communication [3], motivating models that reduce spike count, enforce sparse activity, or exploit conditional execution. More broadly, conditional-computation methods, such as adaptive computation time [6], learn to allocate computation selectively instead of applying the same processing everywhere. These works establish the importance of spiking activity and communication costs, but typically *report a single aggregate quantity* such as total spike count or executed operations. In contrast, we separate local predictive activity from inter-layer communication: a layer can spend more activity locally to predict its input while still reducing or restructuring the signal it sends forward.

Predictive coding in spiking networks is an active but fragmented line of work. Ororbia [14] proposed spiking neural predictive coding for continual learning from data streams, using predictive objectives to support adaptation in spiking models. N'dri et al. [12] recently surveyed predictive coding with SNNs, covering models where spiking dynamics, local errors, and predictive objectives are combined in biologically inspired learning systems. Difference Predictive Coding takes a more communication-oriented view, by transmitting sparse differences between predictions and inputs when training SNNs [7]. These works show that prediction errors can be represented or used in spiking systems, but they do not investigate whether a layer can trade local predictive computation for a more selective inter-layer message while preserving task information.

The closest prior art is Predictive Coding Light (PCL) [11], which builds a hierarchical spiking network where predictable spikes are suppressed and a compressed representation is forwarded, very close in spirit to our predictive-suppression mechanism. PCL is unsupervised, relies on spike-timing-dependent inhibitory plasticity, and is framed as a biologically plausible representation-learning model. Differently, our model is supervised and trained with surrogate gradients, the gate is explicitly driven by the prediction error, and the analysis is centred on communication: total activity is decomposed into local processing and communicated parts, weighted and event-based communication are kept separate, surprise selectivity and redundant communication are measured, and the communicated message is tested for linear decodability and task accuracy.

3 Methods

3.1 Predictive layer designs

Both predictive variants use the same local encoder-predictor skeleton, see Fig. 2. For an input $x_\ell[t]$ at layer ℓ and time step t, the layer computes a spiking latent representation and a local reconstruction:

$$z_\ell[t] = \mathrm{Enc}_\ell(x_\ell[t]), \qquad \hat{x}_\ell[t] = \mathrm{Pred}_\ell(z_\ell[t]), \tag{1}$$

where Enc_ℓ is a linear-leaky-integrate-and-fire (LIF) encoder and Pred_ℓ is a shallow LIF-based predictor mapping $z_\ell[t]$ back to the layer input dimensionality. The prediction residual and its scalar magnitude are

$$e_\ell[t] = x_\ell[t] - \hat{x}_\ell[t], \qquad m_\ell[t] = \frac{1}{D}\sum_{d=1}^{D} |e_\ell[t,d]|\,. \tag{2}$$

The two variants differ in what is communicated to the next layer.

Error units. The error-units variant follows the direct predictive-coding interpretation and forwards the residual

$$y_\ell[t] = \mathrm{LIF}([\ \mathrm{relu}(e_\ell[t]);\ \mathrm{relu}(-e_\ell[t])\ ])\,, \tag{3}$$

in order to take into account negative errors.

Predictive suppression. In this case, the residual is used as a control signal, whose magnitude defines an error-driven gate,

$$g_\ell[t] = \mathrm{clamp}\Big(1 - e^{-\alpha m_\ell[t]},\ g_{\min},\ 1\Big), \tag{4}$$

where clamp bounds the gate between $g_{\min}$ and 1. Hence, the layer communicates a gated version of its latent spike train:

$$y_\ell[t] = g_\ell[t] z_\ell[t]. \tag{5}$$

When the input is predicted well, the gate closes toward $g_{\min}$ and predictable spikes are suppressed; when the input is surprising, the gate opens and spikes are forwarded. We set $\alpha = 10$ after a preliminary sweep over $\alpha \in \{5, 7, 10\}$, which gave stable accuracy, and use $g_{\min} = 0.05$ as a small floor to avoid fully closing the communication channel.

The representation $z_\ell[t]$ is binary, while the gate $g_\ell[t]$ is continuous. Thus the communicated signal in Eq. (5) is *graded*: a predictive amplitude modulated spike train, not a pure binary event stream. The receiving layer remains spiking, integrating $y_\ell[t]$ and emitting its own spikes. Training minimises a task loss, a predictive (reconstruction) loss, and a min-rate regulariser that keeps the predictor from collapsing to silence,

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \beta\big(\mathcal{L}_{\text{pred}} + \lambda\,\mathcal{L}_{\text{rate}}\big), \tag{6}$$

with $\beta = 0.1$ and $\lambda = 2$, where $\mathcal{L}_{\text{rate}} = [\ 0.05 - \bar{r}_{\text{pred}}\ ]_+^2$ penalises predictor firing rates that fall below a target rate of 0.05.

3.2 Strictly event-driven predictive suppression

Predictive suppression in (5) uses a graded inter-layer channel. Here, we additionally define two binary variants that retain the same error-driven gate, while enforcing strictly event-based communication. A hard threshold with a straight-through estimator, termed **hard-STE**, emits $y_\ell[t] = \mathbf{1}[g_\ell[t] > \theta]\, z_\ell[t]$ in the forward pass, with gradients flowing through the underlying continuous gate. The second variant, termed **respike**, re-binarises the graded current through a LIF neuron, $y_\ell[t] = \mathrm{LIF}(\gamma\, g_\ell[t]\, z_\ell[t])$, yielding a more conventional SNN block. The hard-threshold is empirically set to $\theta = 0.3$ on N-MNIST and $\theta = 0.1$ on SHD; the respike gain is $\gamma = 4$.

4 Evaluation metrics

Our methodological approach is to separate where activity is spent between local processing and communication, instead of using a single total spike count. The total activity cost is thus

$$E_{\text{total}} = E_{\text{local}} + E_{\text{comm}}, \tag{7}$$

where E_{local} accounts for local processing activity (encoding, prediction), whereas E_{comm} is the activity communicated across layers. The unweighted sum is the special case where local and communicated activities have equal cost. To account for communication-dominated regimes, we consider the weighted proxy

$$C(\rho) = E_{\text{local}} + \rho\, E_{\text{comm}}, \tag{8}$$

where ρ represents the relative cost of communicated activity; ρ=1 recovers the unweighted total, while ρ>1 models regimes in which transmitting activity is more expensive than local computing. $C(\rho)$ is an activity-proxy sensitivity: it allows to gauge whether the local predictive cost is compensated for by the reduction in inter-layer communication. For the communicated part, two measures characterize the communicated activity:

$$E^{\text{w}}_{\text{comm}} = \sum_{t,d} g_\ell[t]\, z_\ell[t,d], \quad E^{\text{ev}}_{\text{comm}} = \sum_{t,d} \mathbf{1}[y_\ell[t,d] \neq 0]. \tag{9}$$

The weighted measure on the left captures the modulated activity of the proposed graded model. The event measure on the right counts how many binary spikes are actually transmitted, the natural quantity for the binary variants and the SNN baseline.

Beyond the activity decomposition we use three diagnostic metrics. Let $E_{\text{repr},\ell}$ be the activity of the local spiking representation (before suppression) at layer ℓ and $E_{\text{comm},\ell}$ the activity forwarded to the next layer. The *compression ratio*

$$\mathrm{Comp}_\ell = \frac{E_{\text{comm},\ell}}{E_{\text{repr},\ell}}$$

measures how much of the local representation at layer ℓ is communicated to the next layer; lower values indicate stronger suppression.

Compression alone says nothing about whether communication is suppressed selectively. We define two error-conditioned diagnostics. Within each batch, the (t, b) time-data batch pairs of a layer are ranked by their local prediction-error magnitude $m_\ell[t]$, and the top and bottom quartiles (25%) form the high- and low-error subsets, respectively. If C_{high} and C_{low} denote the mean communicated activity on these two subsets, the *surprise-selectivity index* (SSI) is defined as

$$\mathrm{SSI} = \frac{C_{\text{high}} - C_{\text{low}}}{C_{\text{high}} + C_{\text{low}}}.$$

Specifically, a high SSI (ideally approaching 1) means that communication activity across layers concentrates on surprising inputs, i.e., that C_{high} dominates C_{low}. The companion *redundant-communication fraction* (RCF)

$$\mathrm{RCF} = \frac{C_{\text{low,total}}}{C_{\text{total}}}$$

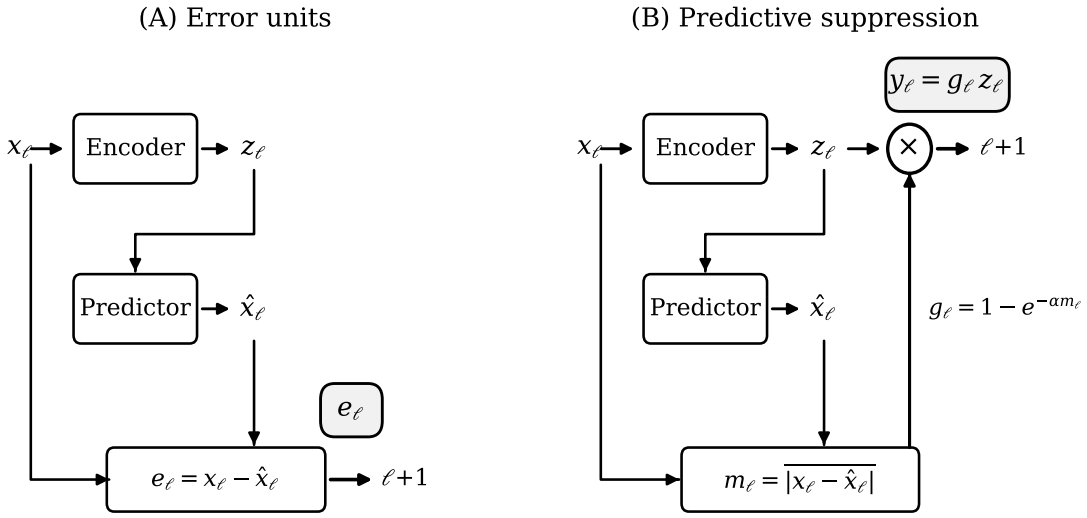

Figure 2: What each layer communicates. The two schemes share the same encoder/predictor skeleton; only the forwarded message differs. (A) The error-units variant forwards the prediction error $e_\ell = x_\ell - \hat{x}_\ell$. (B) Predictive suppression encodes z_ℓ, predicts its own input $\hat{x}_\ell$, and forwards $y_\ell = g_\ell z_\ell$, where the gate $g_\ell = 1 - e^{-\alpha m_\ell}$ is driven by the prediction-error magnitude.

measures the share of total communication spent on low-error inputs; low RCF means little activity is wasted on predictable inputs. Linear probes trained on the time-averaged communicated signal and, separately, on the full local representation check whether the compressed message remains linearly decodable into the class label.

5 Experimental setup

Testing is performed on two event-based benchmarks that are compatible with a temporal multilayer architecture: N-MNIST [13] (2×34×34, ten classes) and Spiking Heidelberg Digits [2] (700 channels, twenty classes). Events are binned into $T = 16$ frames. All models are multilayer spiking networks with matched hidden sizes (256-192-128) and identical preprocessing; the matched baseline is a plain feedforward LIF network of the same capacity. Training uses surrogate gradients via SpikingJelly [4] and the Adam optimiser, with five seeds. Preliminary sweeps over $\alpha \in \{5, 7, 10\}$ left N-MNIST accuracy stable (0.975-0.976, Table 1); we use $\alpha = 10$ as the main operating point. All reported cost metrics are activity proxies, not measured hardware energy.

6 Numerical Results

6.1 Predictive coding trades local layer processing for selective communication

Our results confirm that, on N-MNIST suppressing predictable spikes does not reduce task accuracy. In fact, the proposed predictive coding schemes improve over the matched SNN baseline while using less aggregate weighted inter-layer communication, see Table 1. As expected, the predictor adds local computation within a layer, but the signal sent between layers becomes leaner and more task-relevant. Fig. 3 summarises this trade-off and shows how it changes as communication becomes more expensive than local processing (increasing ρ).

Decodability of the communicated message. A compact channel is useful only if it preserves task information. To verify this, we train linear probes on two signal types at each layer: the compressed message communicated to the next layer, and the full local spiking representation before suppression. Across layers, the two probe accuracies nearly coincide, see Fig. 4. This demonstrates that predictive coding removes activity that the downstream decoder does not need, while preserving a linearly accessible class signal. At the first layer the prediction-driven gate removes the largest share of redundancy: the message carries 7× less weighted activity than the baseline (198 versus 1397) while remaining as linearly decodable as the full representation.

Prediction-error selectivity. Decodability shows that the message is sufficient, it does not show that the gate is prediction-driven. In fact, a random gate could preserve class information while still wasting activity on predictable inputs. Fig. 5 shows that this is not the case: the communicated activity monotonically increases with the prediction-error magnitude, the SSI is high and the RCF fraction is small. As expected from the error-driven gate, communication increases with prediction error: "surprising" inputs receive more communication activity, while predictable ones are strongly suppressed.

6.2 Event-driven variants and temporal transfer

Predictive suppression layers use a graded inter-layer channel because the gate is continuous. We now test whether the same gate can support strictly binary communication.

As shown in Table 1, training binary messages end-to-end recovers much of the graded model's performance, but the preferred binarization method depends on the dataset. On N-MNIST, **respike** is the best performing binary variant. On SHD, **hard-STE** is the preferred method, although it has the weakest compression. Predictive suppression, i.e., the graded channel option, is the most robust design overall.

Table 1: Summary of main results. Accuracy is mean ± standard deviation over five seeds. SSI and RCF are only reported for suppression-based communication. E_{comm} is computed on the actual transmitted signal y_ℓ; it is a summed magnitude for graded channels and an event count for binary channels.

Dataset	Model	Channel	Accuracy	E_{comm}	Compr.	SSI / RCF
N-MNIST	SNN baseline	events	0.941 ± 0.002	3006	1.00	—
	Error units	events	0.946 ± 0.004	855	0.33	—
	Predictive suppression	weighted	0.976 ± 0.001	821	0.52	0.770 / 0.050
	Respike	events	0.959 ± 0.002	943	0.81	0.995 / 0.001
	Hard-STE	events	0.950 ± 0.012	736	0.73	0.985 / 0.003
SHD	SNN baseline	events	0.621 ± 0.005	3094	1.00	—
	Error units	events	0.271 ± 0.012	151	0.17	—
	Predictive suppression	weighted	0.717 ± 0.004	747	0.51	0.920 / 0.020
	Hard-STE	events	0.698 ± 0.009	1119	0.89	0.930 / 0.020
	Respike	events	0.639 ± 0.009	931	0.75	0.950 / 0.013

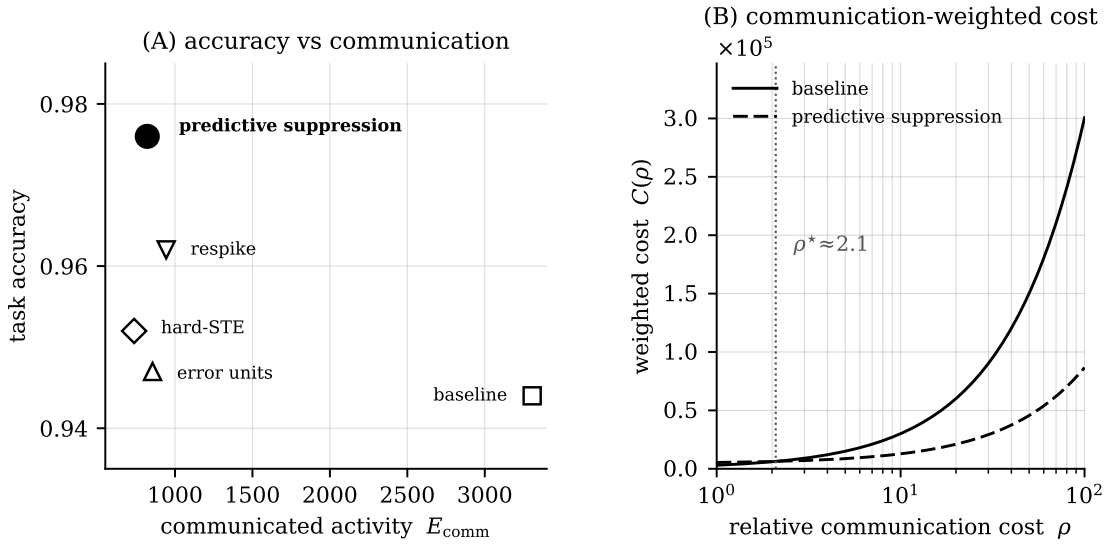

Figure 3: (A) Accuracy versus communicated activity proxy (summed over layers) on N-MNIST. Predictive suppression (filled circle) keeps high accuracy while communicating less than the matched baseline. (B) Communication-weighted cost sensitivity $C(\rho) = E_{\text{local}} + \rho\, E_{\text{comm}}$. As predictive suppression spends extra local processing to reduce communication, it becomes cheaper than the baseline once communication is weighted about $\rho^\star \approx 2.1$ times more than local computation.

Transfer to SHD. SHD provides a temporal test beyond event-based vision. Under the same simple feedforward architecture, predictive suppression again improves over the matched SNN baseline, with high selectivity and low redundant communication (Table 1). Instead, the error-units variant performs poorly in this setting. This suggests that the error is more useful as a control signal for communication than as the message itself.

7 Discussion

Predictive coding increases the total spiking activity via the addition of a local predictor. This allows changing what is sent between layers, attaining a compact, *surprise-oriented* and linearly decodable channel. This type of efficiency departs from spike-count minimisation and becomes relevant when communication is more expensive than local layer processing, as on hardware where moving activity between cores dominates the energy budget or IoT edge nodes where output features are to be transmitted over a wireless channel.

The cost curve in Fig. 3B plots the weighted cost $C(\rho)$ (see Eq. (8)) for a predictive suppression layer and a standard SNN layer as a function of factor ρ, gauging how much communication activity costs with respect to local processing. As can be seen, predictive suppression becomes convenient as it crosses the SNN baseline at $\rho^\star \approx 2.1$. In such regimes, predictive suppression becomes cheaper than the SNN baseline.

The continuous predictive suppression gate trains reliably and yields the most compact output channel. Its event-driven variants show that the approach can be pushed to binary, genuinely spike-based communication, although which variant works best depends on the dataset. Event-driven predictive suppression is feasible, but which binary mechanism is best is not universal.

8 Conclusions

In this paper we proposed novel *predictive coding* designs for SNNs. The new layer type integrates a predictor that is utilized to infer

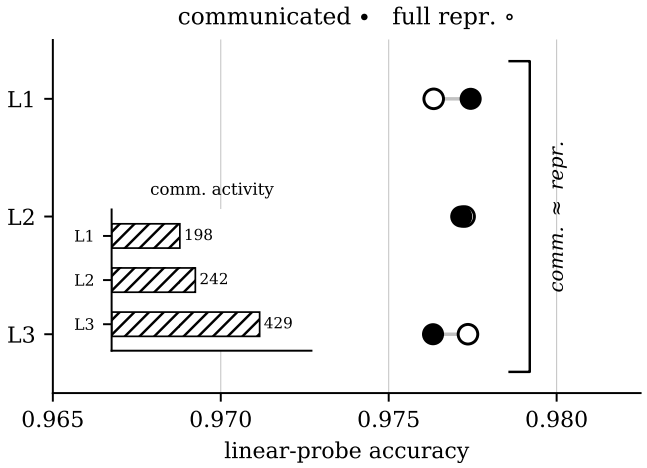

Figure 4: Linear-probe decodability for predictive suppression. At every layer the communicated message (filled) is essentially as decodable as the full representation (open); the paired dots nearly coincide. The first layer message carries about 7× less weighted activity than the baseline. Inset: the communicated activity that this message costs, per layer.

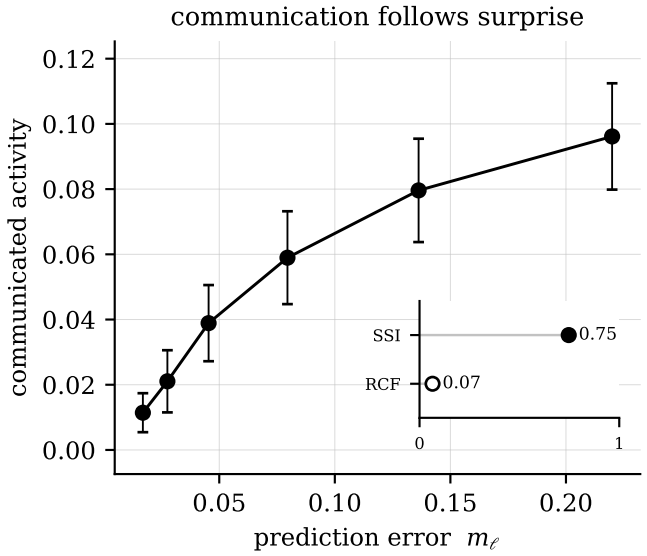

Figure 5: Communication is selective, not merely reduced: at the first layer of predictive suppression, communicated activity rises monotonically with the pred.-error magnitude m_ℓ. Inset: SSI (filled) is high and the RCF (open) is low.

and forward only non-predictable input to downstream layers. The rationale is that encoding "surprise" should lead to a leaner channel between layers, minimizing the amount of communicated activity. This is particularly appealing for exchanging features across processing cores or for edge IoT devices where SNN features are to be communicated via a wireless channel to a decision point. Our numerical results, obtained on N-MNIST and SHD datasets confirm the effectiveness of the proposed approach, namely, the channel becomes surprise-oriented and low-redundancy. Moreover, output features lead to better accuracies than with the matched SNN baseline and are more linearly decodable. Top-down and iterative extensions of the predictive suppression layers, convolutional gates to handle spatial event data, and direct hardware-energy measurements are natural extensions to our work.

Acknowledgments

The work in this paper is supported by the Multi-X project that has received funding from the Smart Networks and Services Joint Undertaking under the European Union's Horizon Europe R&I programme under GA No. 101192521.

References

[1] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. 2020. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications* 11, 1 (July 2020), 3625.

[2] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. 2022. The Heidelberg Spiking Data Sets for the Systematic Evaluation of Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 33, 7 (July 2022), 2744–2757.

[3] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* 38, 1 (Jan. 2018), 82–99.

[4] Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. 2023. SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances* 9, 40 (Oct. 2023), 1–70.

[5] Karl Friston. 2010. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience* 11, 2 (Feb. 2010), 127–138.

[6] Alex Graves. 2016. Adaptive Computation Time for Recurrent Neural Networks.

[7] Ville Karlsson, Nicklas Fianda, and Joni-Kristian Kämäräinen. 2026. Difference Predictive Coding for Training Spiking Neural Networks. In *Proceedings of the International Conference on Learning Representations (ICLR-26)*. Rio de Janeiro, Brazil, 1–20.

[8] J. Carlos López-Ardao, Raúl F. Rodríguez-Rubio, Andrés Suárez-González, Miguel Rodríguez-Pérez, and M. Estrella Sousa-Vieira. 2021. Current Trends on Green Wireless Sensor Networks. *Sensors* 21, 13 (Jan. 2021), 4281.

[9] Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. 2022. Predictive Coding: Towards a Future of Deep Learning beyond Backpropagation?. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)*. International Joint Conferences on Artificial Intelligence, Vienna, Austria, 5538–5545.

[10] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. 2019. Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-Based Optimization to Spiking Neural Networks. *IEEE Signal Processing Magazine* 36, 6 (Nov. 2019), 51–63.

[11] Antony W. N'dri, Thomas Barbier, Céline Teulière, and Jochen Triesch. 2025. Predictive Coding Light. *Nature Communications* 16, 1 (Oct. 2025), 8880.

[12] Antony W. N'dri, William Gebhardt, Céline Teulière, Fleur Zeldenrust, Rajesh P. N. Rao, Jochen Triesch, and Alexander Ororbia. 2026. Predictive coding with spiking neural networks: A survey. *Neural Networks* 196 (April 2026), 108371.

[13] Garrick Orchard, Ajinkya Jayawant, Gregory K. Cohen, and Nitish Thakor. 2015. Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Frontiers in Neuroscience* 9 (Nov. 2015), 437.

[14] Alexander Ororbia. 2023. Spiking neural predictive coding for continually learning from data streams. *Neurocomputing* 544 (Aug. 2023), 126292.

[15] Rajesh P. N. Rao and Dana H. Ballard. 1999. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience* 2, 1 (Jan. 1999), 79–87.

[16] James C. R. Whittington and Rafal Bogacz. 2017. An Approximation of the Error Backpropagation Algorithm in a Predictive Coding Network with Local Hebbian Synaptic Plasticity. *Neural Computation* 29, 5 (May 2017), 1229–1262.

[17] Friedemann Zenke and Surya Ganguli. 2018. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks. *Neural Computation* 30, 6 (June 2018), 1514–1541.