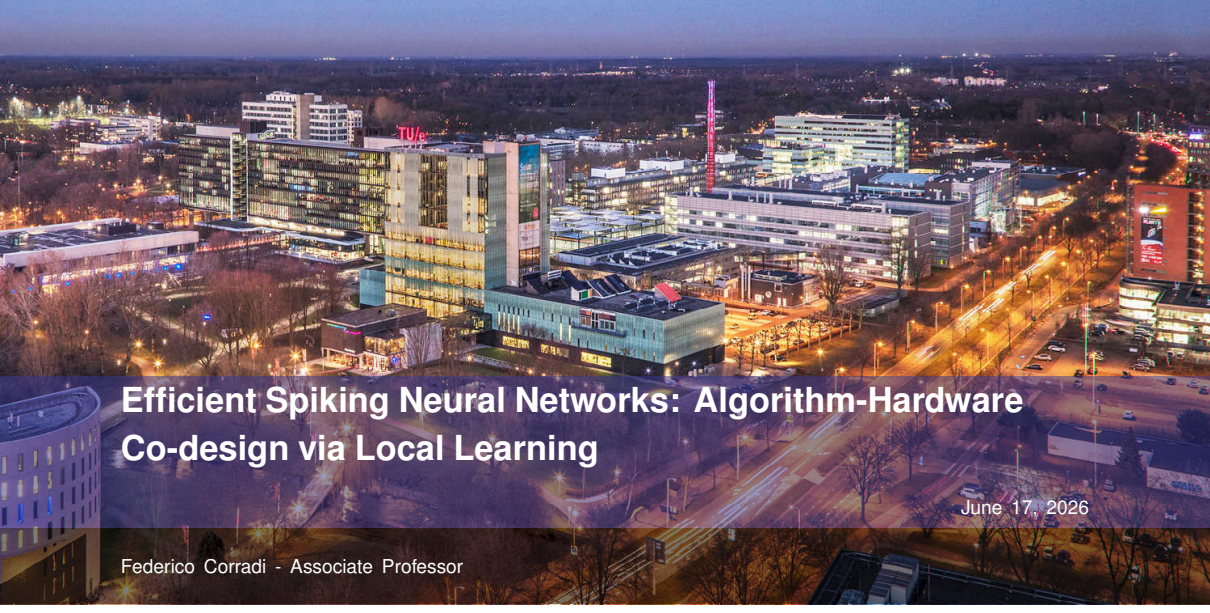


Efficient Spiking Neural Networks: Algorithm-Hardware Co-design via Local Learning

June 17, 2026

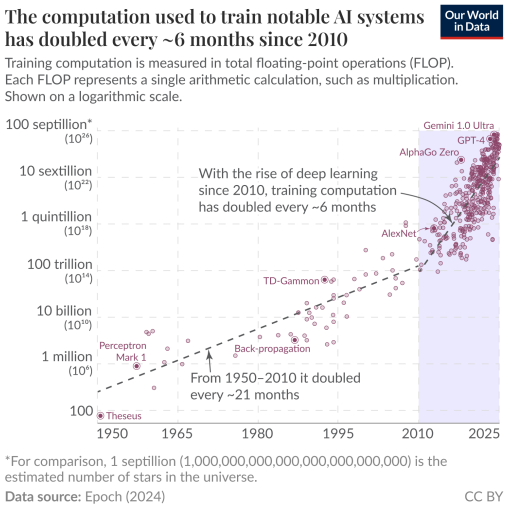
Federico Corradi - Associate Professor

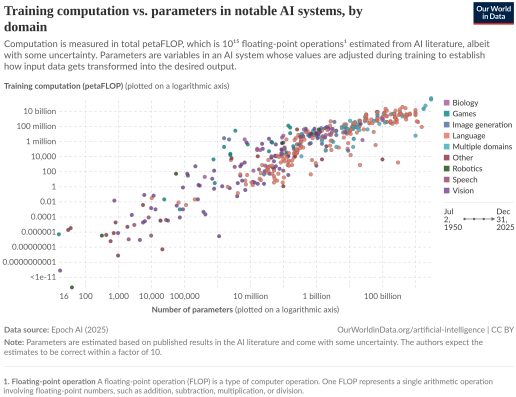
Eindhoven University of Technology, Neuromorphic Edge Computing Systems Lab

Neuromorphic Hardware and Algorithms 2026, University of Sussex, United Kingdom

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Deep Learning Seems to Have No Limits: Compute & Data





How much Training Data? [Epoch AI]

$C \sim 6ND$ (C compute, N param., D tokens)

The Efficiency Gap: LLMs vs Brains

State-of-the-Art LLM (Llama 3 405B)

1. Data Throughput

15T Tokens (Text-only)

$\approx$ 50 TB

2. Energy to Train

Cluster (54 days)

21,600 MWh *(Meta, "The Llama 3 Herd of Models", 2024)*

3. Result

Static Statistical Predictor

The 4-Year-Old Child (Vision)

1. Data Throughput (Vision only)

LeCun's: ~~1 PB~~ Optic nerve: $\sim$1 MB/s $\rightarrow$ $\sim$**60 TB**

(Koch et al., 2006; $\sim$12 h/day awake)

2. Energy (entire brain)

20 W $\times$ 4 Years

0.7 MWh *(incl. all functions)*

3. Result

Adaptive Multi-modal Learner

The Verdict: Orders-of-Magnitude Gap

Comparable data throughput (same order of magnitude), yet:

Biology uses $\sim$**30,000**$\times$ **less energy** (0.7 vs 21,600 MWh) even **counting all brain functions**, not just learning.

We need architectures and algorithms that extract meaning from continuous event streams.

The Opportunity: Neuromorphic Gatekeepers

The Problem: Blind Processing

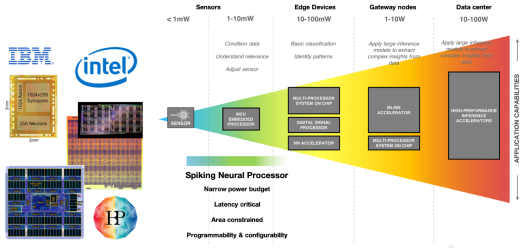
Today's Cloud AI is a "dumb pipe":

- **Redundant** raw data streams.
- **Kilowatts** on static information.
- Relies on **offline**, frozen models.

The Alternative: Intelligent Gatekeepers

The same principles that make biological learning efficient (**event-driven, local, online**) also enable efficient inference.

- **Reject Redundancy**.
- **Learn at the Edge**.
- **Continual adaptation**.

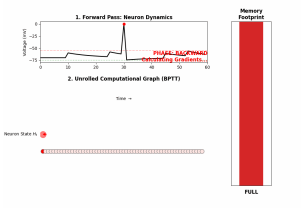


Spiking or Event-Driven NN seems the natural substrate — but how do we train them with local, online rules?

The Problem: Current Training for SNN is Inefficient

Standard: Backpropagation Through Time (BPTT)

- Global Error Propagation: Needs entire network state.
- Time Unrolling: Must store history of all activations.
- The Result: Massive memory footprint $\mathcal{O}(T \cdot H \cdot L)$.

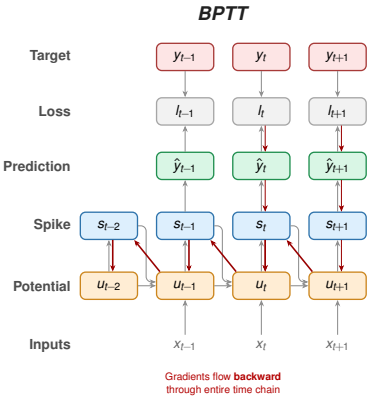


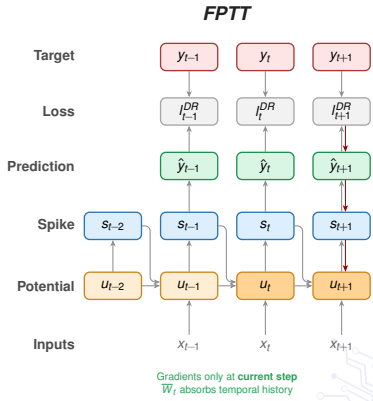
[Animation Link]

The Memory Bottleneck

- Memory is the primary constraint.
- Latency cannot tolerate backward locking.

BPTT vs Forward Propagation Through Time (FPTT)





Variables: u_t = membrane potential, s_t = spike, $\hat{y}_t$ = prediction

Key: FPTT replaces temporal backprop chain with dynamic regularization $l_t^{DR} = \mathscr{L}_t + \mathscr{R}(\overline{W}_t)$

[B. Yin, F. Corradi, S. Bohte, Nature Machine Intelligence, 2023]

FPTT: Dynamic Loss & Gradient Update

Dynamic Regularization

$$l_t^{DR} = \underbrace{\mathscr{L}_t}_{\text{Instantaneous Loss}} + \underbrace{\mathscr{R}(\overline{W}_t)}_{\text{Dynamic Regularization}} \quad (1)$$

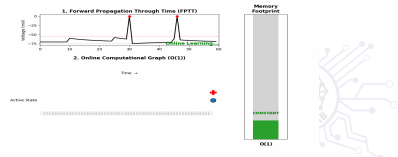
- **Instantaneous Loss**: standard t, no history needed.
- **Dynamic Regularization**: penalizes deviation from running weight average $\overline{W}_t$.

Gradient Update (no temporal chain)

$$\frac{\partial l_t^{DR}}{\partial W} = \frac{\partial l_t}{\partial \hat{y}_t} \cdot \frac{\partial \hat{y}_t}{\partial s_t} \cdot \frac{\partial s_t}{\partial u_t} \cdot \frac{\partial u_t}{\partial W} \quad (2)$$

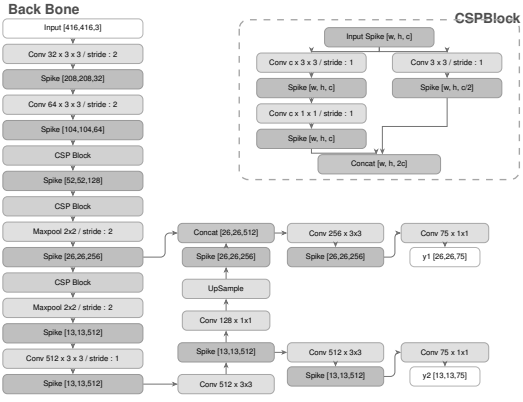
- Depends only on current states (u_t, s_t) — no chain through $u_{t-1}, u_{t-2}, \ldots$
- Eliminates vanishing/exploding gradients over time.

Result: Gradients depend only on current state $\rightarrow \mathscr{O}(H \cdot L)$ memory.



[B. Yin, F. Corradi, S. Bohte, NMI, 2023]

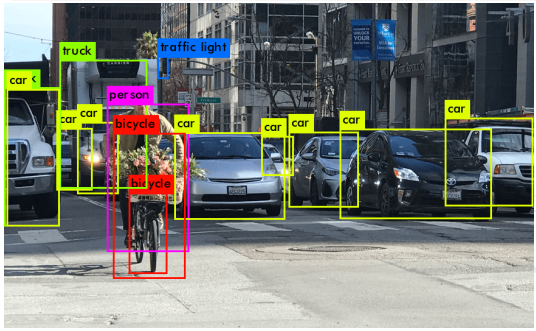
FPTT Performance: Scaling to Deep Networks



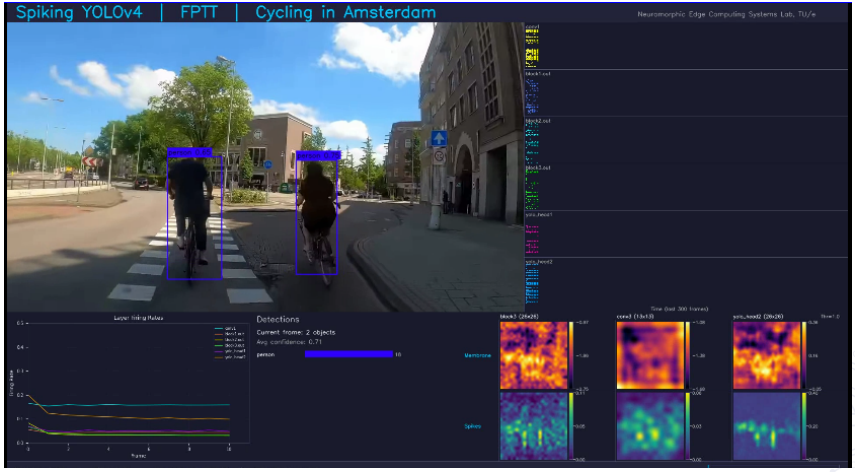
[Yin B., Corradi F., Bothe S., Nature Machine Intelligence, 2023]

SpyYoloV4 SNN

- 6.2M spiking neurons, 14M parameters
- Conv, FC, cross-stage partial subnets
- Trained end-to-end with FPTT
- Approaching DNN accuracy



FPTT Performance: Scaling to Deep Networks

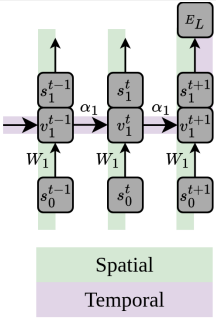


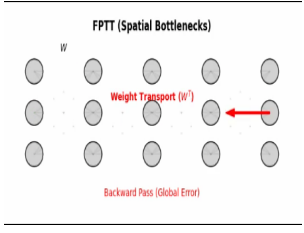
[B. Yin, F. Corradi, S. Bohte, 2023 NMI]
[Animation Link]

The Spatial Bottleneck

FPTT Solved Time, but...

- We still have **Spatial Credit Assignment** issues.
- Update Locking: Layers must wait for the forward pass to finish.
- Weight Transport: Backward weights must match forward weights (biologically implausible).

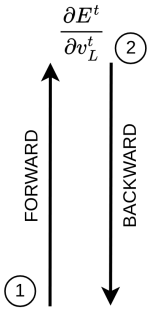




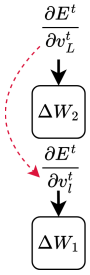
[Animation Link]

We need a solution that is Local in Time AND Space.

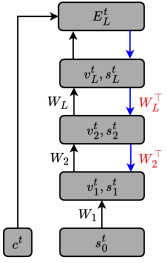
Spatial Credit Assignment Bottlenecks and Mitigations



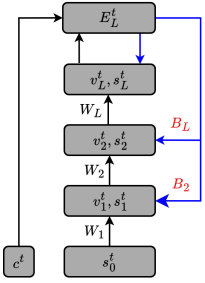
Update locking



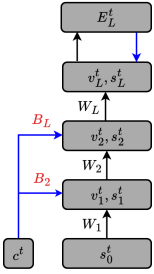
Sequential updates



Weight transport



Direct feedback alignment

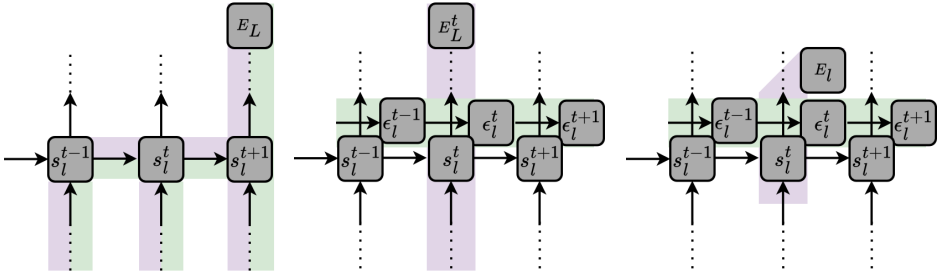


Direct-random-target projection

DFA [Nøkland A, 2016,] [Lee J. et al., 2020] , DRTP [Frenkel C. et al, 2021], showing potential but..

- Accuracy gap vs. BP $\rightarrow$ limited to small datasets
- Both require extra projection matrices (B_l)

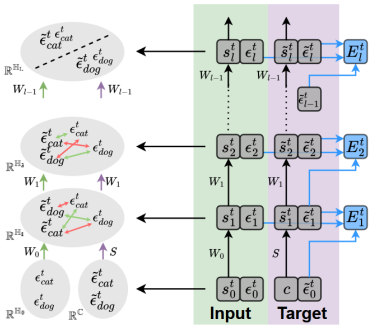
From BP to Traces Propagation (TP)



Fully Local Learning

- **No Backpropagation**: Forward-only passes.
- **Local Loss**: Layer-wise contrastive objectives.
- **Eligibility Traces**: Capture temporal dynamics at the synapse.

Traces Propagation: How it Works



spike: $s_l^t = \Theta(v_l^t - V_{th})$

trace: $e_l^t = \beta e_l^{t-1} + s_l^t$

spike: $\tilde{s}_l^t = \Theta(\tilde{v}_l^t - V_{th})$

trace: $\tilde{e}_l^t = \beta \tilde{e}_l^{t-1} + \tilde{s}_l^t$

$$E_l^t = CE(\underbrace{e_l^t(\tilde{e}_l^t)^T}_{y_t^l}, \underbrace{Softmax(\tilde{e}_{l-1}^t(\tilde{e}_{l-1}^t)^T)}_{z_t^l})$$

$$\frac{dL_l^t}{dW_{l-1}} = \underbrace{(y_l^t - z_l^t)}_{\text{Modulation signal}} \left[\left(e_l^t \Theta'(v_l^t - V_{th}) s_{l-1}^t\right) + \left(\tilde{e}_l^t \Theta'(\tilde{v}_l^t - V_{th}) \tilde{s}_{l-1}^t\right) \right]$$

Mechanism

- **Modulation Signal**: Drives input traces to align with target traces for same-class examples.
- **Divergence**: Pushes traces apart for different classes.

Traces Propagation: Local Update Rule

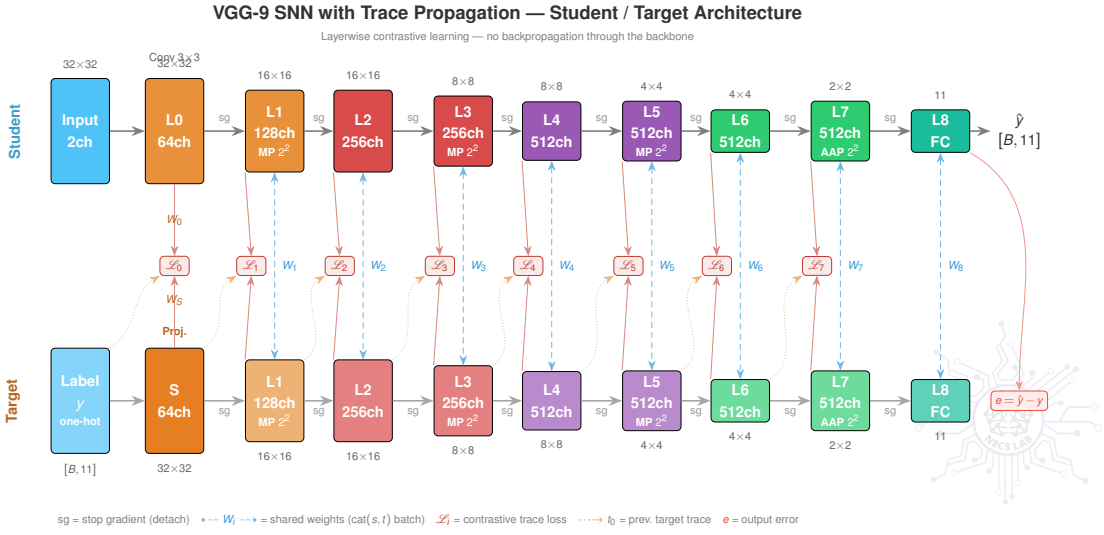
2. The Gradient Update

The weight update is computed using only locally available information (traces and spikes), eliminating the backward pass:

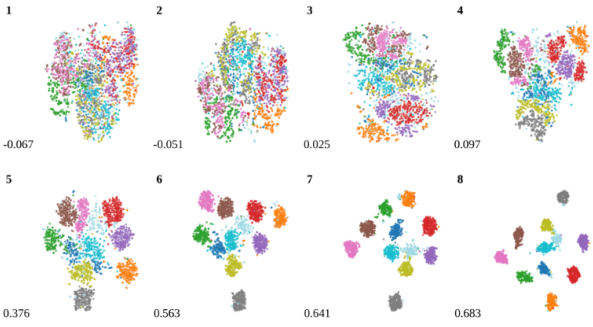
$$\Delta W_l \propto \underbrace{\left(\text{Softmax}(z_l^t) - y_l^t\right)}_{\text{Modulation Signal}} \cdot \underbrace{\tilde{\varepsilon}_l^t}_{\text{Post-syn. Trace}} \cdot \underbrace{\Theta'(v_l^t)}_{\text{Surrogate}} \cdot \underbrace{s_{l-1}^t}_{\text{Pre-syn. Spike}} \tag{3}$$

- **Modulation Signal:** Pushes input traces of the same class together and different classes apart (Green/Red arrows).
- **Three-Factor Rule:** The update combines a global (layer-wise) error signal with local pre- and post-synaptic activity.

Quantitative Results: TP on Dynamic Data and Deep SNNs



Intuition: Separation in Feature Space



[L. Pes, B. Yin, S. Stuijk, F. Corradi, IOP NCE, 2026]

t-SNE Visualization

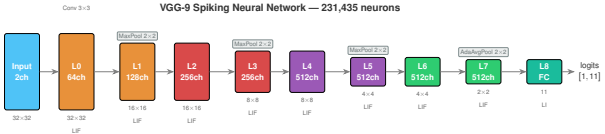
- Visualizing input traces at different layers.
- **Layer 1:** Mixed, hard to separate.
- **Layer 8:** Clear clusters for 11 classes.
- Experimental results: local rules can learn global representations.

Traces Propagation Complexity and Memory

Table: $L \rightarrow$ layers, $H \rightarrow$ neurons, $T \rightarrow$ seq. length, $O \rightarrow$ output classes.

Model	Update Locking	Weight Transport	Time Local	Space Local	Space Complexity	Time Complexity	Auxiliary Matrices
BPTT	✗	✗	✗	✗	TLH	TLH^2	–
E-prop [Bellec 2020]	✗	✗	✓	✗	LH^2	LH^2	–
E-prop(*rnd*) [Bellec 2020]	✗	✓	✓	✗	LH^2	LOH	LOH
OSTL [Ortner 2022]	✗	✗	✓	✗	LH^2	LH^2	–
OSTL(*rnd*) [Ortner 2022]	✗	✓	✓	✗	LH^2	LOH	LOH
S-TLLR [Apolinario 2023]	✗	✗	✓	✗	LH	LH^2	–
DECOLLE [Kaiser 2020]	✓	✓	✓	✓	LH	LOH	LOH
ETLP [Quintana 2024]	✓	✓	✓	✓	LH	LOH	LOH
OSTTP [Ortner 2023]	✓	✓	✓	✓	LH^2	LOH	LOH
TESS [Apolinario 2023]	✓	✓	✓	✓	LH	LOH	LOH
TP (ours)	✓	✓	✓	✓	LH	LH	OH

TP Results: Deep SNNs



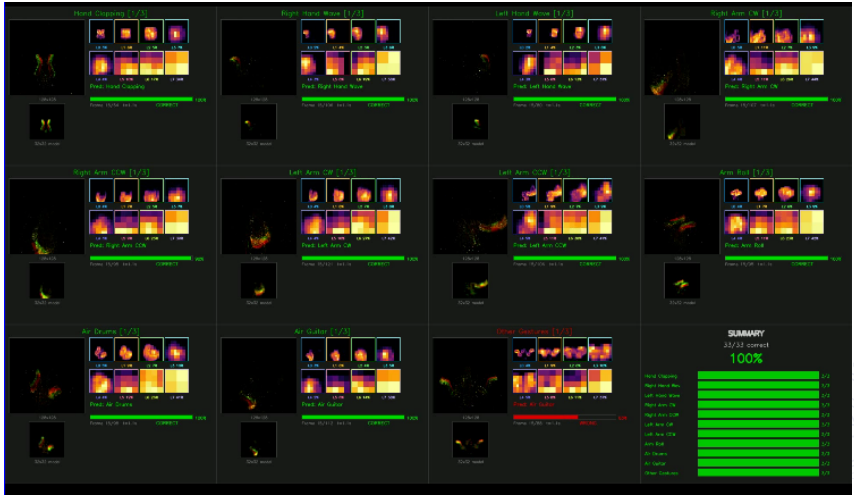
Model	Architecture Type	Neuron Type	Local Learning	Time Steps	Number of Epochs	Test Accuracy
			DVS-GESTURE			
BPTT (ours)	VGG-9	LIF	✘	20	200	98.56 ± 0.15
OTTT [7]	VGG-9	LIF	Partial (time)	20	200	96.88
S-TLLR [9]	VGG-9	LIF	Partial (time)	20	200	98.48 ± 0.37
DECOLLE [12]	3-CONV	LIF	✔	1800	200	95.54 ± 0.16
TESS [13]	**VGG-9**	**LIF**	✔	**20**	**200**	$\mathbf{98.56 \pm 0.31}$
TP(**ours**)	VGG-9	LIF	✔	20	200	98.18 ± 0.15
			DVS-CIFAR10			
BPTT [13]	VGG-9	LIF	✘	10	200	76.40 ± 0.66
OTTT [7]	VGG-9	LIF	Partial (time)	10	200	76.27 ± 0.05
S-TLLR [9]	VGG-9	LIF	Partial (time)	10	200	75.14 ± 1.37
TESS [13]	**VGG-9**	**LIF**	✔	**10**	**200**	$\mathbf{75.00 \pm 0.65}$
TP(**ours**)	VGG-9	LIF	✔	10	200	71.39 ± 0.19

[L. Pes, B. Yin, S. Stuijk, F. Corradi, IOP NCE, 2026]

DVS Gestures — VGG-9

- Competitive with BPTT on DVS Gestures
- Scalable to DVS CIFAR-10
- $\mathcal{O}(H \cdot L)$ memory, no update locking, no weight transport

TP Results: DVS Gestures VGG-9



[L. Pes, B. Yin, S. Stuijk, F. Corradi, IOP NCE, 2026]

Real-World Adaptation: Fine-Tuning and One-Shot Learning

[Animation Link]

[L. Pes, B. Yin, S. Stuijk, F. Corradi, IOP NCE, 2026]

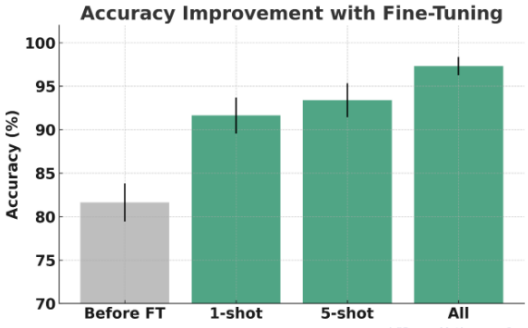
- TP-MLP: 1 hidden layer, 450 recurrent LIF neurons, 36 output classes
- Input: 40-bin mel-spectrogram + $1^{st}/2^{nd}$ order deltas = 120-dim, 101 time steps (1s audio)
- Pre-trained on 84,527 samples from 2,112 speakers → 81% on new speaker
- 1-shot adaptation: 26 samples (1 per class), 3 epochs with TP → **96%**
- Test set: same speaker, different recordings (80/20 split)
- GSC v0.02 dataset (35 words + unknown)

[Github link]

Real-World Adaptation: On-Device Fine-Tuning

The Use Case

- Pre-train on general data (Cloud).
- Deploy to Edge.
- **Adapt Locally**: Fine-tune for the specific user using Trace Propagation.



Example: Google Speech Commands adaptation recovers accuracy for new users with just a few shots.

[L. Pes, B. Yin, S. Stuijk, F. Corradi, IOP NCE, 2026]

Conclusions: From Global Error to Local Activity

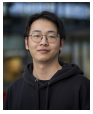
From Global Error to Local Activity and Learning

- **FPTT** eliminates temporal unrolling: $\mathcal{O}(T \cdot H \cdot L) \rightarrow \mathcal{O}(H \cdot L)$ memory, enabling deep online SNN training.
- **Trace Propagation** removes spatial backprop: forward-only, no update locking, no weight transport — fully local in space & time.
- **On-device fine-tuning** with TP enables few-shot adaptation at the edge (speech, gestures, bio-signals).

Next Step: Silicon

- **Algorithm-Hardware Co-Design**: local learning $\Leftrightarrow$ event-driven substrate
- **Exploiting Extreme Sparsity**: asynchronous vs. synchronous overhead
- **Native On-Chip Learning**: why is it still not at the edge?

Neuromorphic Edge Computing Systems Lab



Zhanbo Shen

Lorenzo Pes

Dr. Bojian Yin

Shimeng Ye

Yvonne Vullers

Stefano Chiavazza

Noah Marti

Dr. Roel Jordans

Dr. Federico Corradi

Collaborators

- **Prof. Sander Bohte** (CWI)
- **Dr. Bojian Yin** (Chen Institute)

CWI

imec

CHEN INSTITUTE TIANQIAO & CHRISSY

Interested? Reach out!

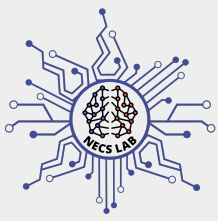
Neuromorphic Edge Computing Systems Lab

Federico Corradi

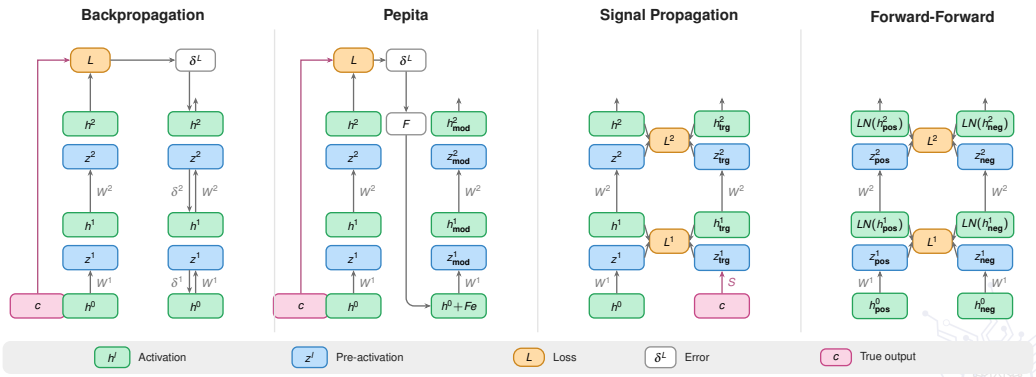
Eindhoven University of Technology

f.corradi@tue.nl, +31(0)402 472 556

Neuromorphic Edge Computing Systems Lab

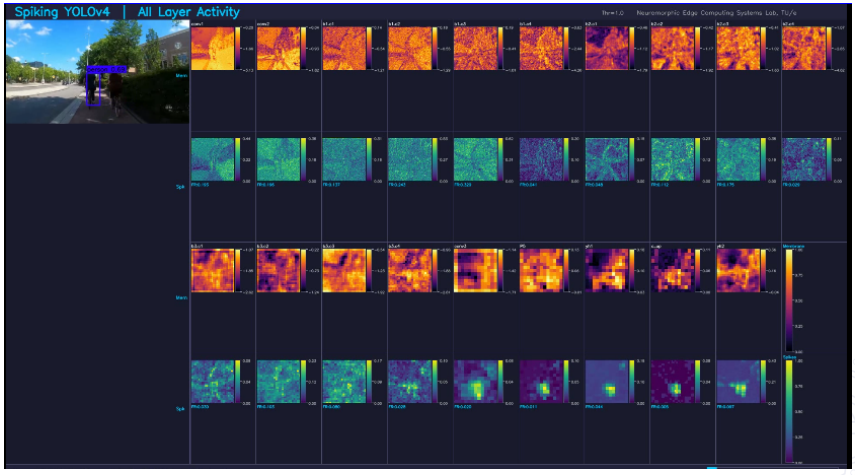


Spatial Credit Assignment: Forward-Only Algorithms (ANN)



Dellaferrera & Kreiman, ICML 2022 | Kohan et al., TNNLS 2023 | Hinton, arXiv 2022

FPTT Performance: Scaling to Deep Networks



[B. Yin, F. Corradi, S. Bohte, 2023 NMI]

[Animation Link]

Traces Propagation: Layer-wise Contrastive Loss

1. The Local Loss Function (E_l^t)

Trace Propagation (TP) defines a local loss at each layer l and time t, aiming to align input traces with target traces:

$$E_l^t = -\sum_{b,b'} y_l^t[b,b'] \log(\text{Softmax}(z_l^t[b,b'])) \tag{4}$$

- **Similarity Score (z_l^t):** Measures alignment between the input trace of example b and the target trace of example b' via dot product: $z_l^t = \varepsilon_l^t(\tilde{\varepsilon}_l^t)^T$.
- **Target Distribution (y_l^t):** A probability distribution derived from the previous layer's target traces, ensuring examples from the same class remain close.