Unsupervised Particle Tracking with Neuromorphic Computing

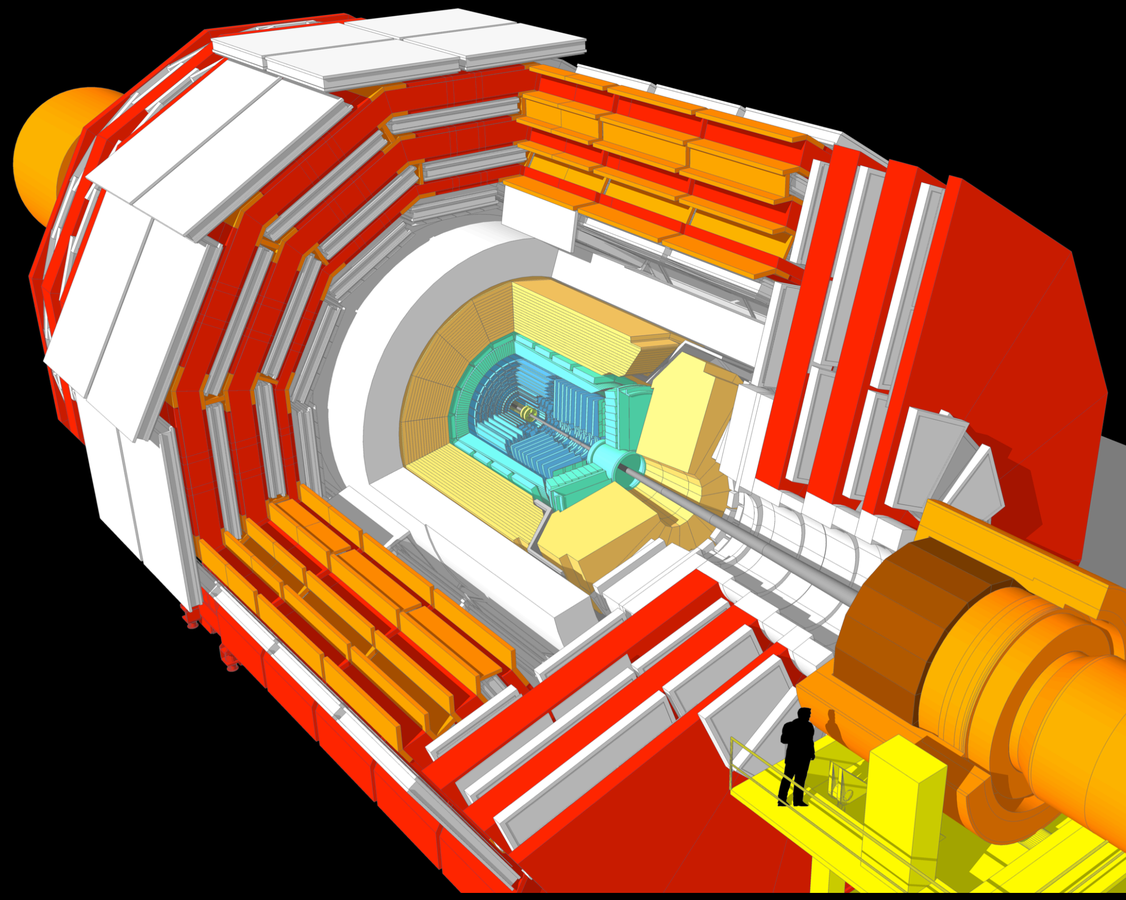
Fabio Cufino, E. Coradin, T. Dorigo, F. Sandin, M. Tosi, M. Awais, E. Lupi, J. Raji, E. Porcu

Fabio Cufino - 17 Jun 2025

Compact Muon Solenoid

CMS detector

- **Muon Detectors**: Capture muon tracks
- Superconducting magnet (3.8 T)
- **Hadronic Calorimeter** (**HCAL**): Measures energy of hadrons
- **Electromagnetic Calorimeter (ECAL)**: Measures energy of photons and electrons
- **Silicon Tracker**: Detects charged particles, reconstructs trajectories



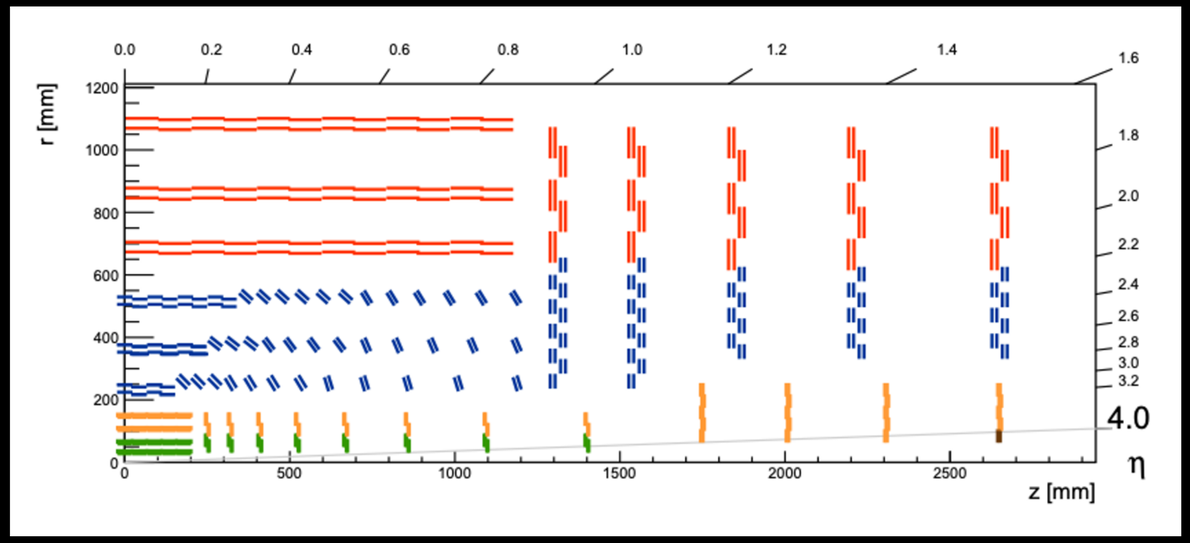
Compact Muon Solenoid

High Lumi LHC

- **HL-LHC:** 200 p-p collisions per bunch crossing (pile-up)
- **Need**: precise detection and fast data processing for high luminosity

Phase 2 Tracker

- **Silicon detectors:** excellent spatial resolution $O(\mu m)$
- **Outer, Inner Tracker**
- **<u>Challenge</u>:** fast & low-power pattern recognition

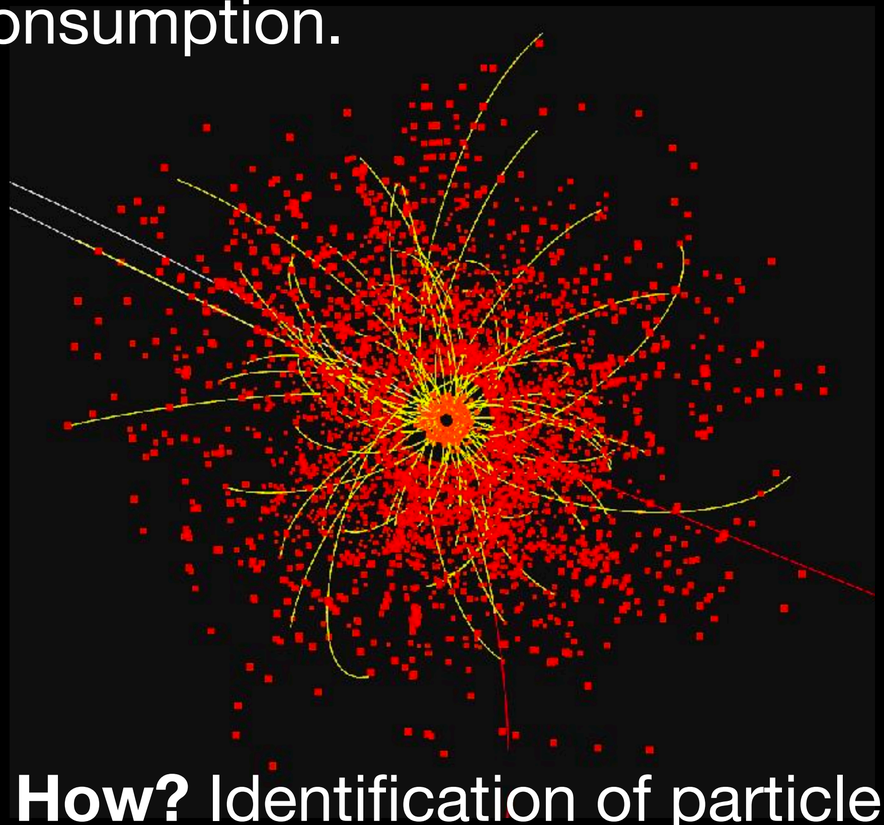


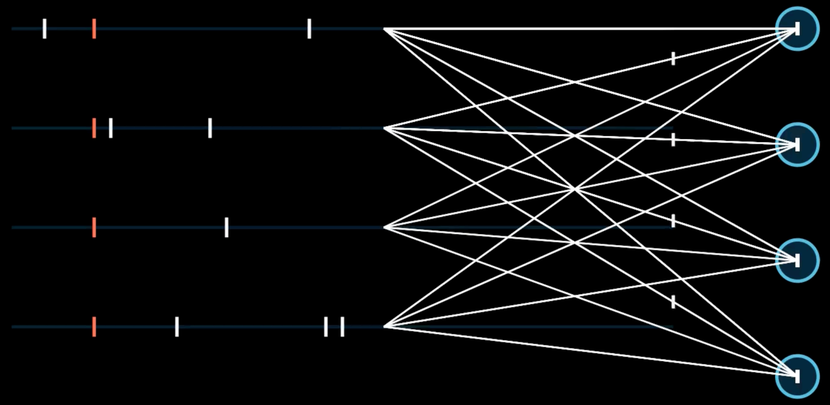
Silicon sensors within a single sector of the Phase-2 CMS tracker [CMS collaboration]

The Goal

Spiking Neural Networks for pattern recognition

- **Use of Spiking Neural Networks**: allow high efficiency with low power consumption.



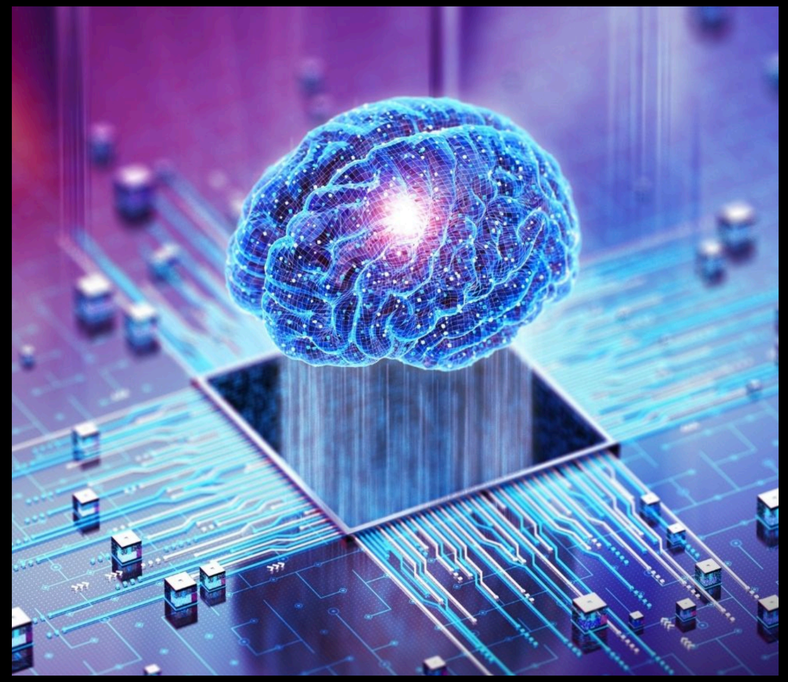


- **How?** Identification of particle tracks via unsupervised pattern recognition

Neuromorphic Computing

What is neuromorphic computing?

- **Emulates** the structure and function of the human brain
- **Spiking Neural Networks (SNNs):** core model used in NC
- SNNs use **discrete spikes** to transmit signals in response to incoming (**discrete)** stimuli.
- **Neurons** and **synapses** are implemented in specialized hardware
- **Distinctive features:**
  - **Energy Efficiency:** Consumes minimal power during operation
  - **Low Latency:** Enables real-time processing

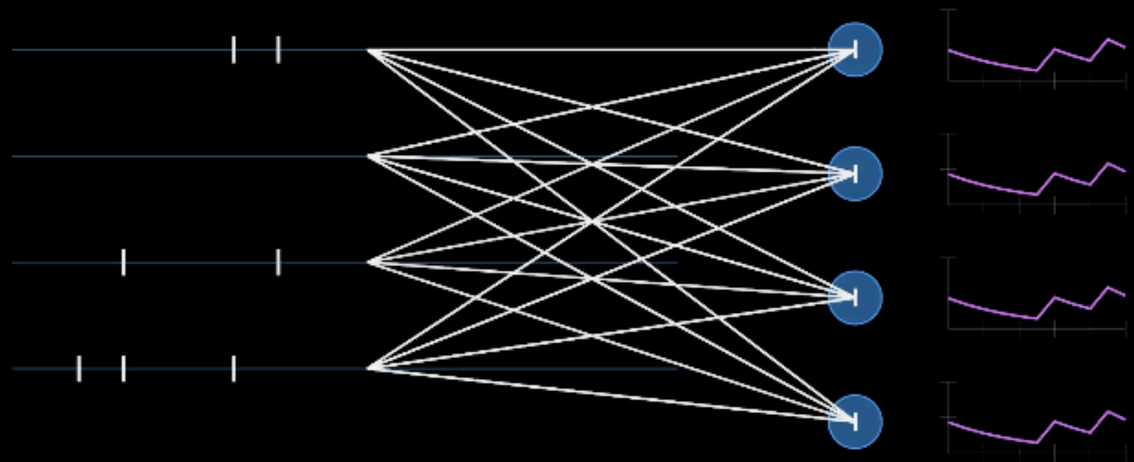


Model of a biological neuron

Leaky integrate-and-fire neuron (LIF)

- **Synapses**: Transfer discrete electrical impulses (**spikes**) to the neuron
- **Neurons**: model the behavior of Membrane Potential (V_{mem})

  - Integrate incoming signals — **fire** when threshold is reached
  - Include a **leaky component** — past inputs decay over time



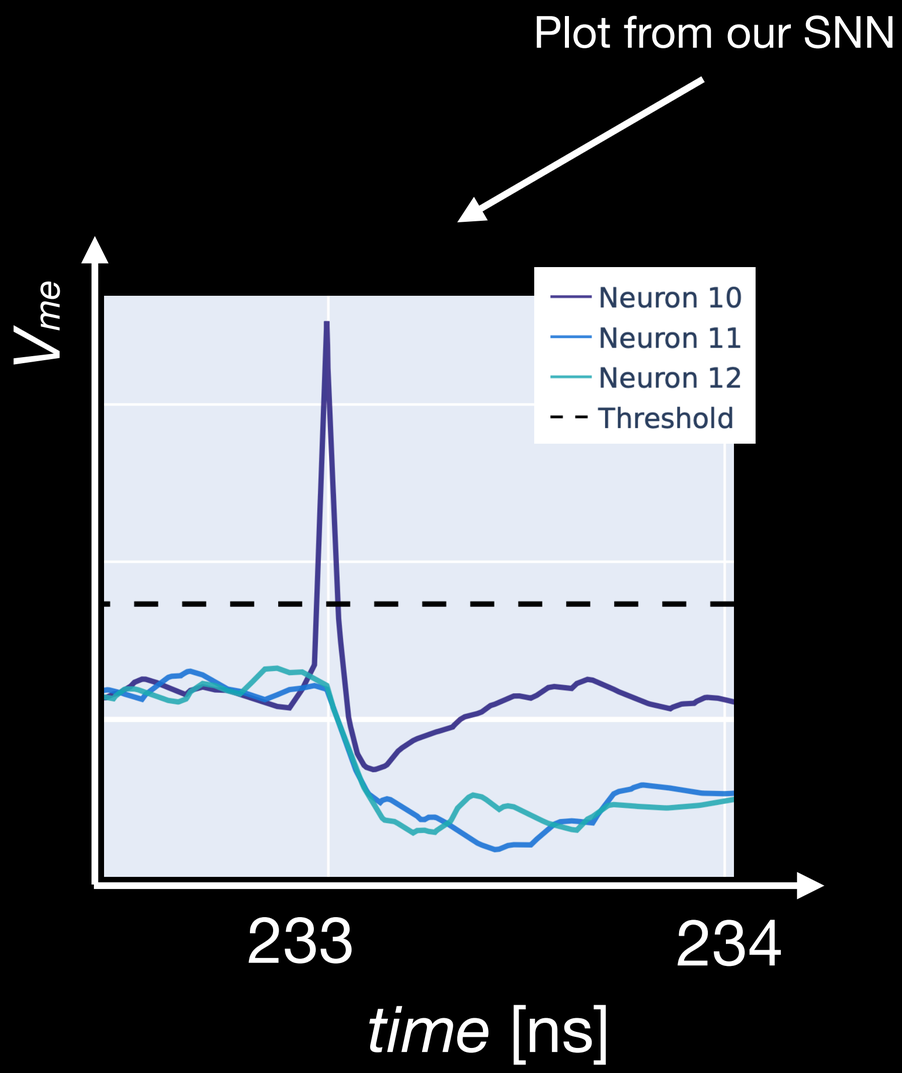
Afferents

V_{mem}

Model of a biological neuron

Leaky integrate-and-fire neuron (LIF)

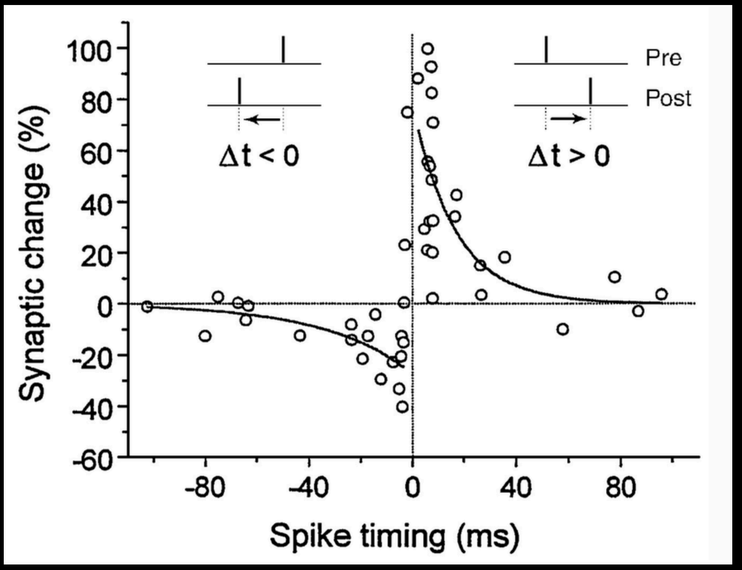
- Membrane Potential (V_{mem}):
  - **Excitatory Postsynaptic Potential (EPSP):**
    - Input spikes increase membrane potential
  - **Neuron activation:**
    - Behavior after V_{mem} exceeds threshold *T*
  - **Inhibitory Post-Synaptic Potential (IPSP):**
    - Competition between neurons of same layer -> *Specialization*



Learning algorithm: STDP

Biological process: Spike timing-dependent plasticity

- **Synaptic strength:** based on the **relative timing between input spikes** and **neuron firing**
  - **Δt > 0:**
    - Synapse **strengthened** → Long-Term Potentiation (LTP)
    - "*Causal*": incoming spike contributed the firing
  - **Δt < 0:**
    - The synapse is **weakened** → Long-Term Depression (LTD)
    - "*Non-causal*": incoming spike didn't help the firing



[arXiv:1907.09126]

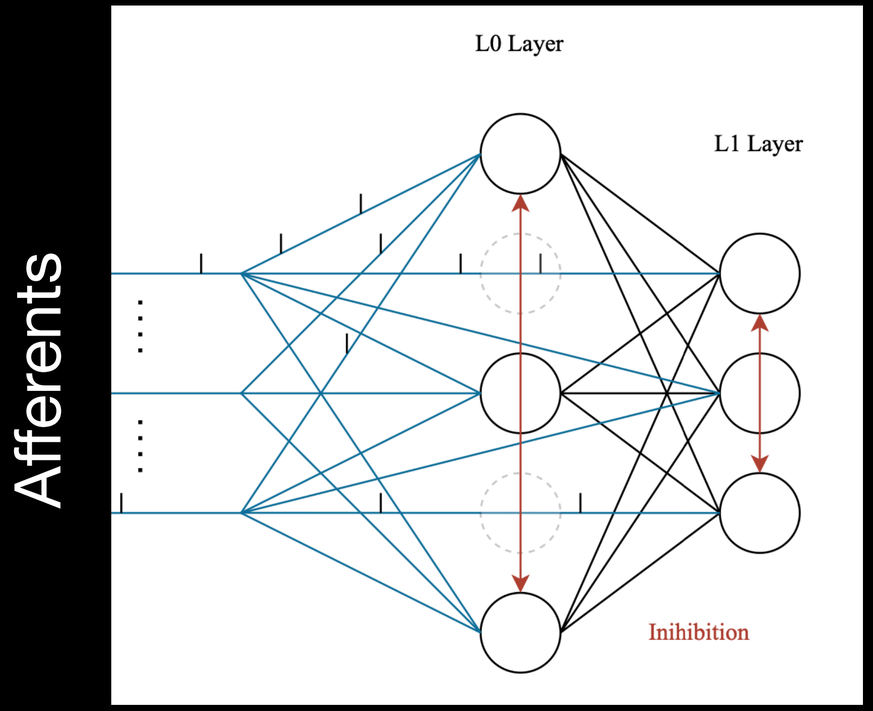
Time-sensitive: Neuron respond to patterns that **precede** firing

P.S. Simple model, we have learned delays…

Network architecture

Spiking Neural Network for Detector Signal Processing

- **10 Afferents**: Fibers that carry electrical signals from the detector layers to the neural network
  - All connect to both **L0** and **L1** neurons
- **Two layers of neurons:** 6 L0, 6 L1
  - Supports **dropout** for regularization
- 20 hyperparameters to be optimize

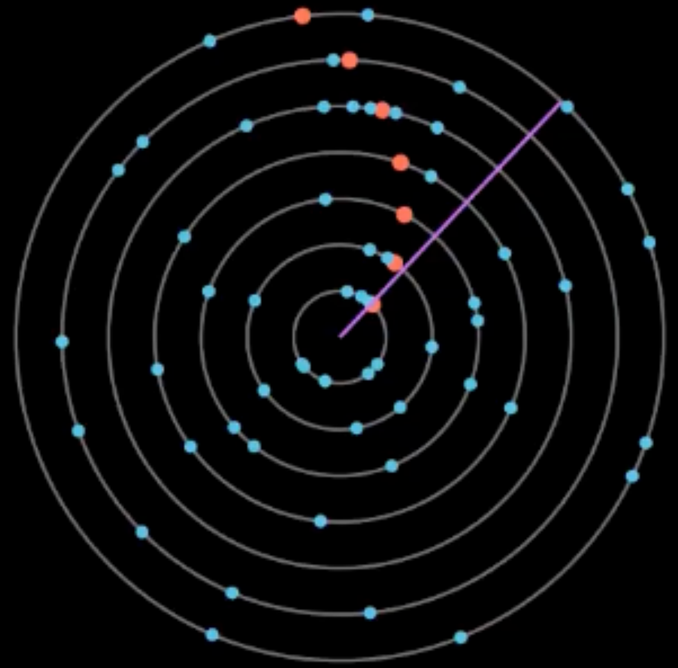


Architecture with layered connectivity & lateral inhibition

Information Encoding

From the detector to the SNN

- **Concept Overview:**
  - Project the 3D event in 2D
  - **Scanning**: to encode spatial geometry as time-dependent signals
  - Detector layers → mapped to afferents
- **Readout frequency:** $f = 40\ MHz$, $\omega = (2\pi + \delta) \cdot f$

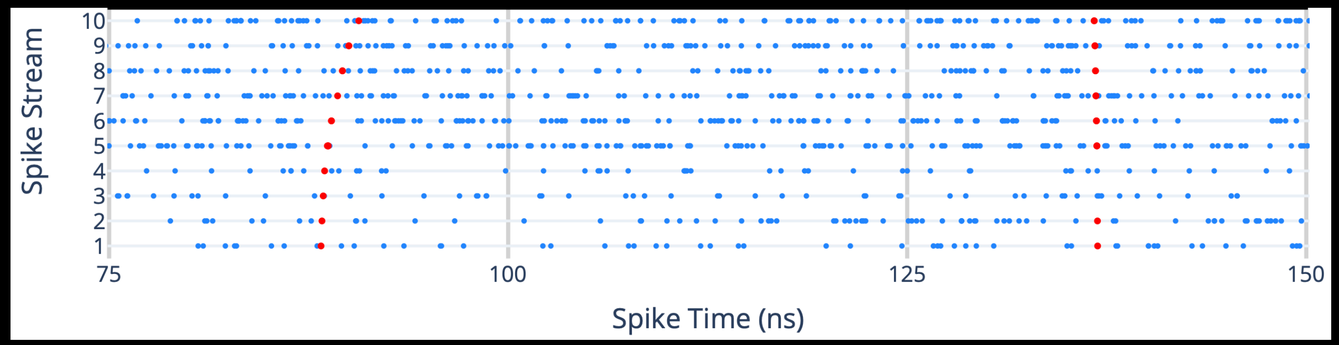


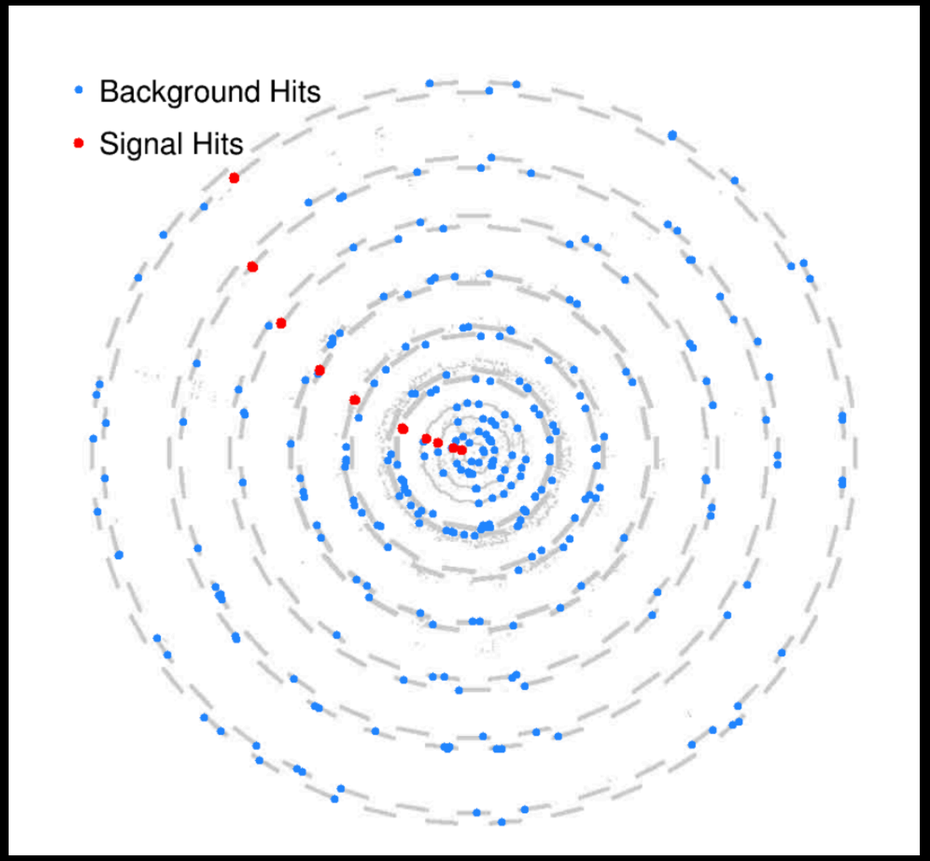
δ: handle border effects

Information Encoding

From the detector to the SNN

- A bit more difficult…
- **Right**: Example of event used for SNN
- **Bottom**: Time-encoded events with and without tracks



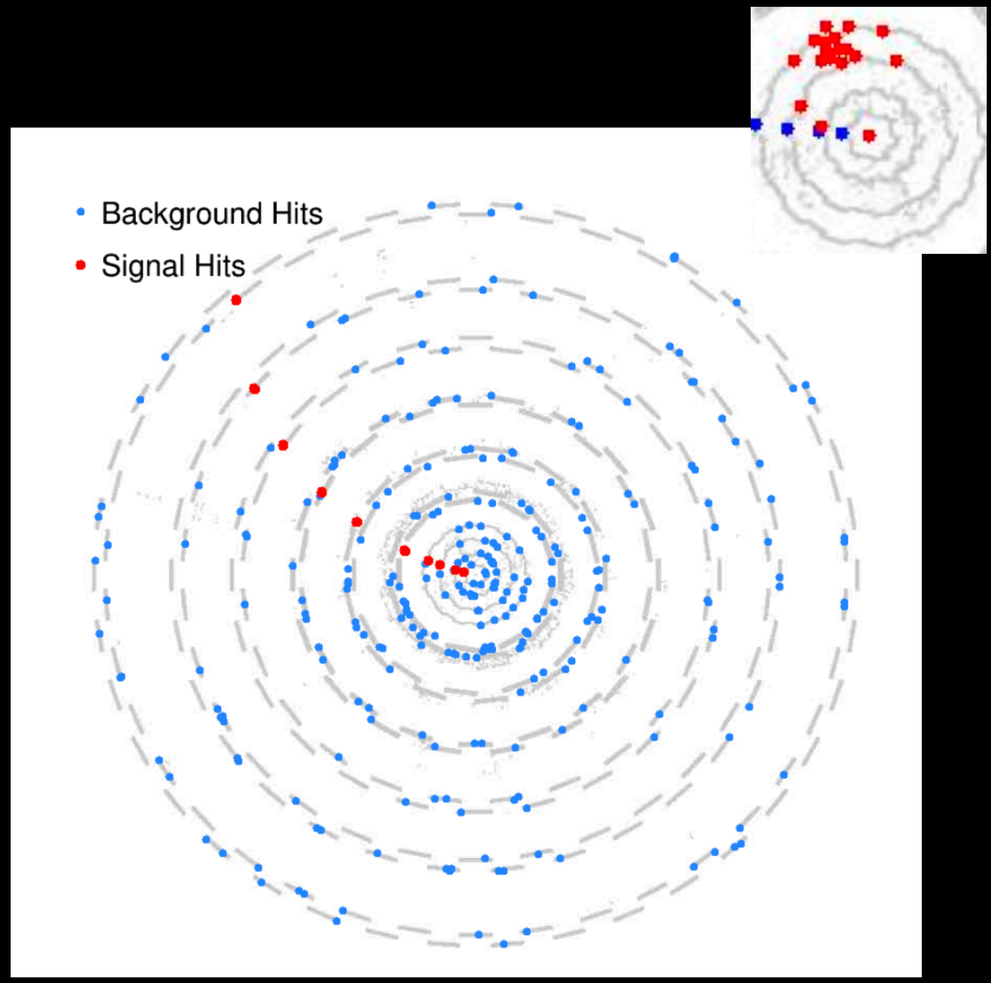


Transverse plane projection of an event with an anti-muon, $p_T = 1\ GeV$ with $Nhit = 300$

Dataset

Monte Carlo simulations

- **For training:** 1 particle per event
  - Muons: q = –1, $p_T \in \{1, 3, 10\}\ GeV$
  - Anti-muons: q = +1, $p_T \in \{1, 3, 10\}\ GeV$
- Contains some interactions with the tracker material
- We superimpose a Poissonian background
  - $\sim N_{BGK}{}_{hit} = 300$, $\sim N_{SIG}{}_{hit} = 10$
  - Simulated 100K events



Transverse plane projection of an event with an anti-muon, $p_T = 1\ GeV$ with $Nhit = 300$

Evaluation functions

Acceptance, Fake Rate, Selectivity

- **Acceptance per class:**
  - Fraction of signal events in which at least one neuron is activated

$$A_{q,p_t} = \frac{N\,SEv(q,p_t) \text{ in which at least 1 neuron was activated}}{\text{TOT } SEv(q,p_t)}$$

- **Fake Rate**:
  - Quantifies how often the SNN activates during **background-only events**.

$$F = \frac{N\,BEvs \text{ in which at least 1 neuron was activated}}{\text{TOT } BEvs}$$

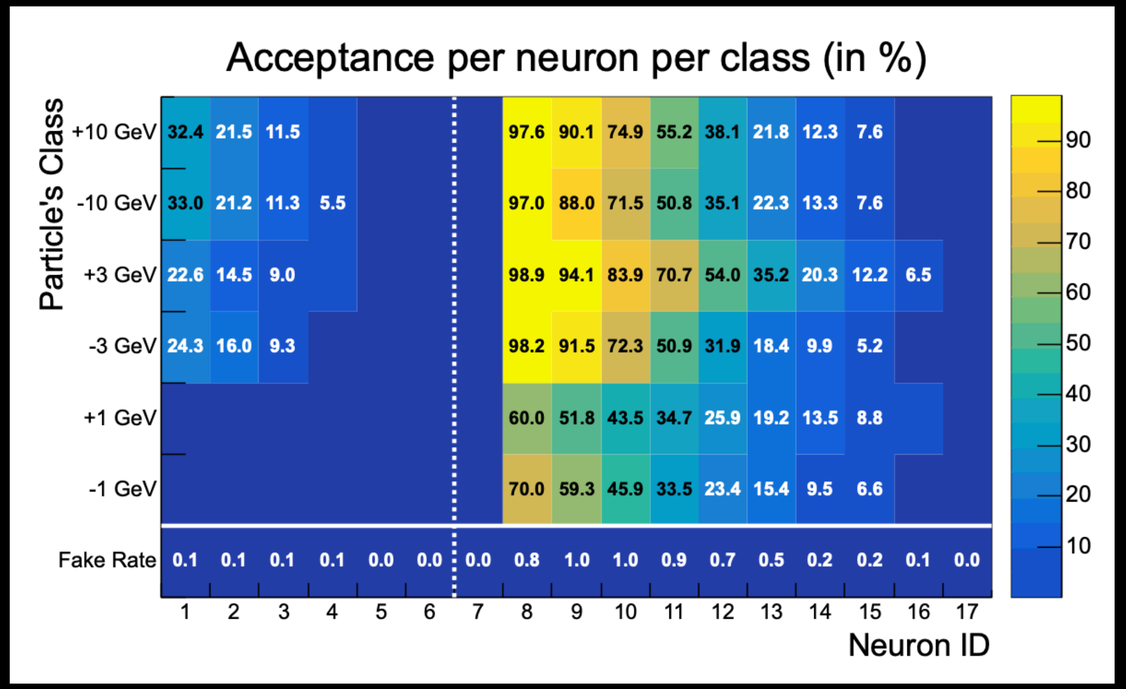
- **Selectivity**:
  - Neuron's ability to discriminate particle classes (e.g. 1 GeV vs. 10 GeV muons).
  - Based on **mutual information**: higher selectivity = more task-specific neurons.

Results

Performance of Network on Test Dataset

- **Test dataset:** *25K events*
- **Architecture:** Increased network complexity
  - 10 neurons in **L1 layer**
- Performance Highlights:
  - **High acceptance** observed, in **N8**
  - **Low fake rate** across all neurons
  - ⚠️ **Limited class specialization**
    → Most neurons activate across multiple particle classes (see heatmap)

Heatmap of neuron activations across particle classes



Genetic Algorithm

Efficiently optimize network configuration

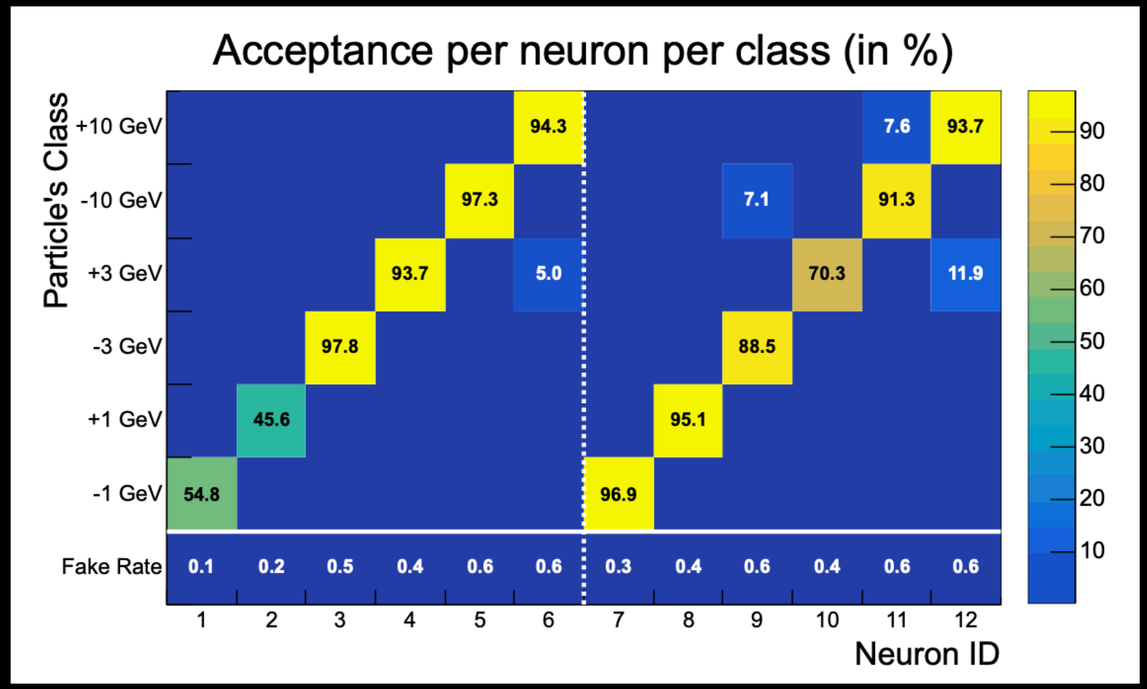
- **Explore** the high-dimensional **hyperpar space**
- NSGA-II (Non-dominated Sorting Genetic Algorithm II), pyGAD library
- Suited for problems with multiple conflicting objectives
  - **maximizing network *Acceptance*, *Selectivity***
  - **minimizing the *Fake Rate***
- **Goal**: find the best ***pareto*** solutions given the 3 objectives

Results after Genetic Algorithm

Performance of Network on Test Dataset

- **Test dataset:** *25K events*
- **Architecture:** 6-6, no need to increase network complexity
- **High acceptance** in all signal classes
- 🎯 **Strong class-to-neuron specialization:**
- Each particle class is primarily detected by a **specific neuron**
  - Neuron 4 → +3 GeV, Neuron 5 → –3 GeV
- **Low fake rate** across all neurons (~0.1–0.6%)

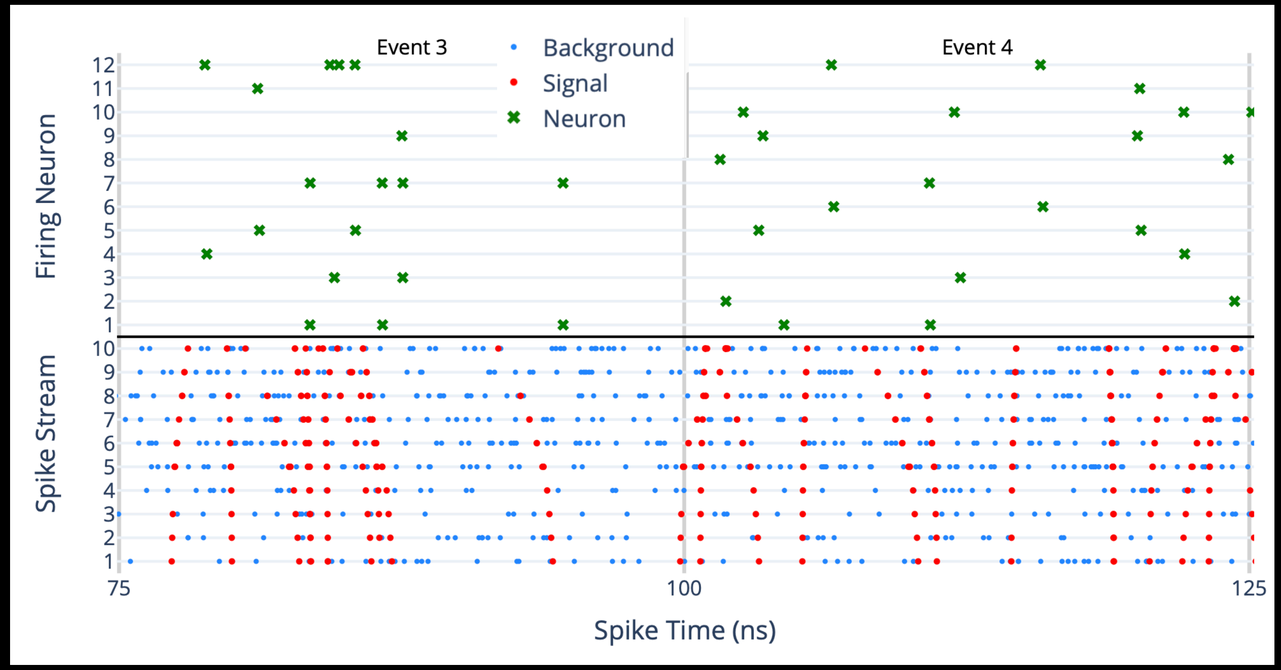
Heatmap of neuron activations across particle classes

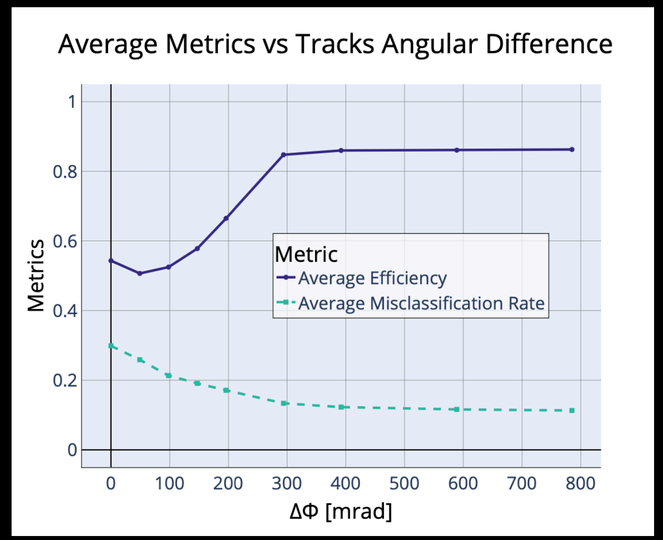


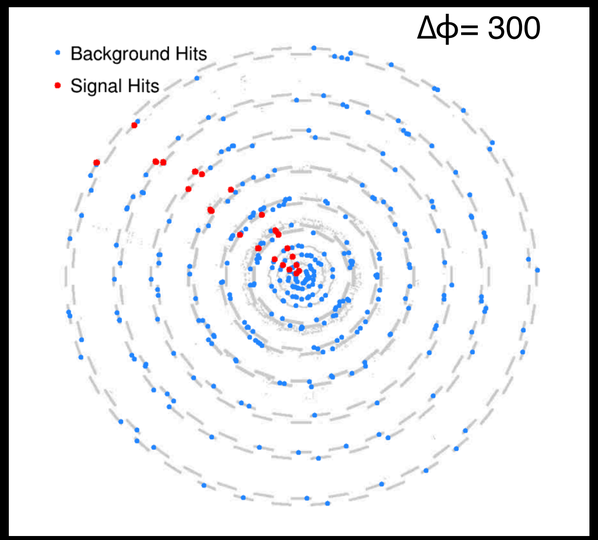
Multi Tracks

Multi-particle events

- Events containing 10 track
- **Top:** Good neurons activations to the input tracks (1st, 2nd tracks)
- Affecting performance
  - Temporal overlap
  - Neurons refractory periods (~0.8 ns)
- **Bottom:** Evaluation SNN's performance as a function of the angular difference between track
- Perfomance decrease: Δϕ = 300







Summary

Key Takeaways

- Successfully demonstrated **SNNs for *unsupervised particle tracking***
  - Opening the way to application of NC for particle tracking
- **Achieved:** *high accuracy*, l*ow fake rate*, and clear *neuron-class specialization*
- **Strong potentia**l for real-time, hardware-based implementation at colliders

Future work?

- 3D event encoding, increase SNN complexity…

Backup

Training the SNN

Evolution of the Network Across Four Stages

Stage 1: Simplified Problem with Minimal Background

- Background noise fixed at Nbkg = 100
- GA optimizes 3-class task: μ^- with pT $\in$ {1, 3, 10} GeV
- Delay learning improves selectivity, reduces FP
- Full hyperparameter space explored

Stage 2: Incorporating Antimuon Classification

- Builds on Stage 1's best network
- Adds μ^+ events → expanded classification
- Specialized neurons duplicated & mirrored
- Weight fixed → Fine tuning delays

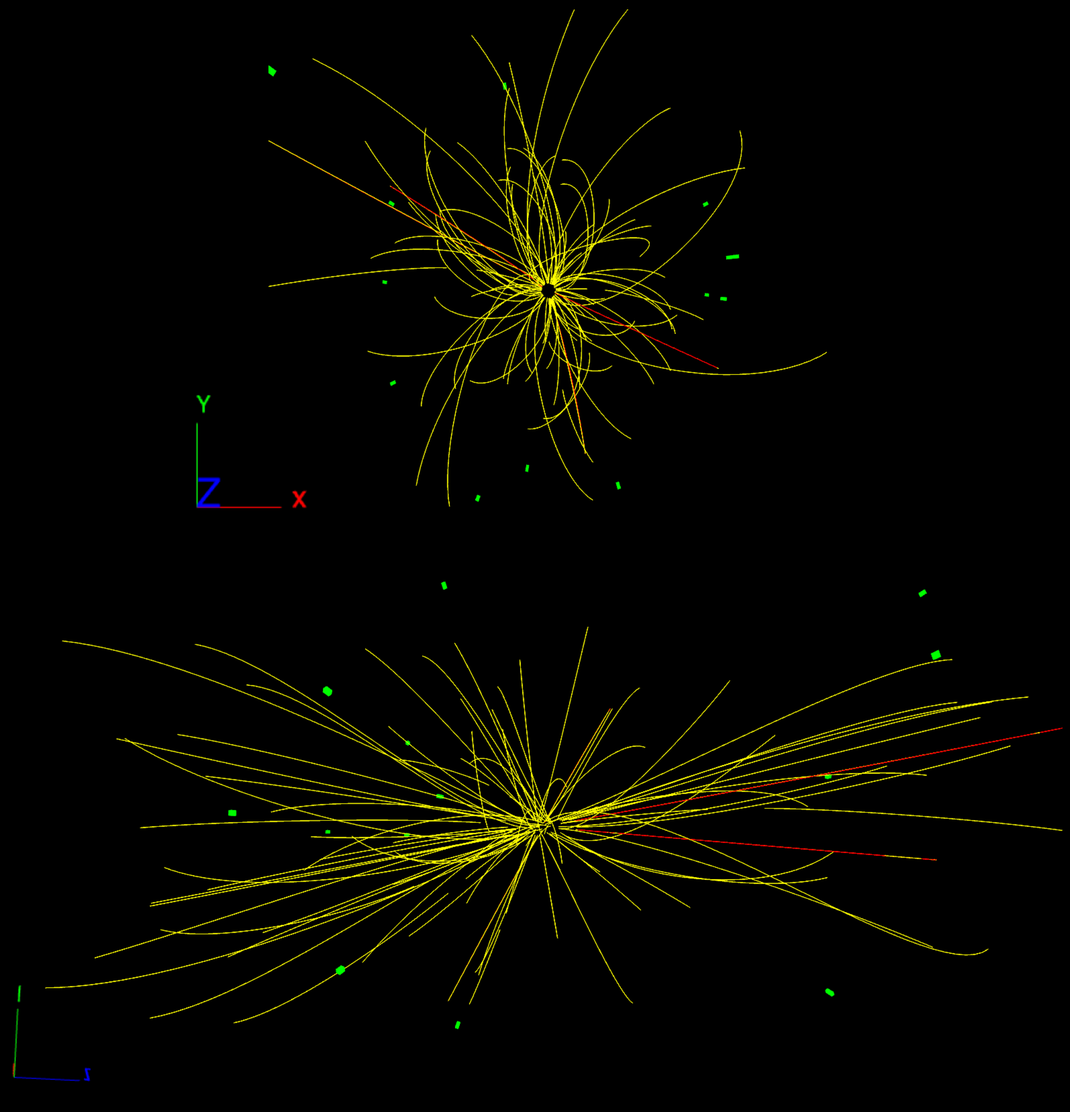
Stage 3: Fine-tuning at High Noise ($\langle N$bkg = 100)

- Starts from Stage 2’s network
- GA explores narrower hyperparameter ranges
- Delay learning rates further reduced
- Focus: incremental accuracy & robustness gains

Encode the 3D

Future work

- 2 stage process:
  - **1st Stage:** use only the projections, identify most of the tracks
  - **2nd Stage:** RE-encode the information of the phi, R (Radial distance in the transv. Plane)



NSGA 2

How does it work?

Main loop:

- **Population Initialization:**
  - Randomly generate SNN configurations (different hyper-par)
- **Evaluation Phase:**
  - Evaluate individuals based on the 3 objectives
- **Non-Dominated Sorting:**
  - Rank solutions using Pareto dominance
- **Crowding Distance:**
  - Select diverse solutions across the Pareto front (variability)
- **Reproduction:**
  - **G**enerate a new population from "Parents" + mutation

NSGA 2

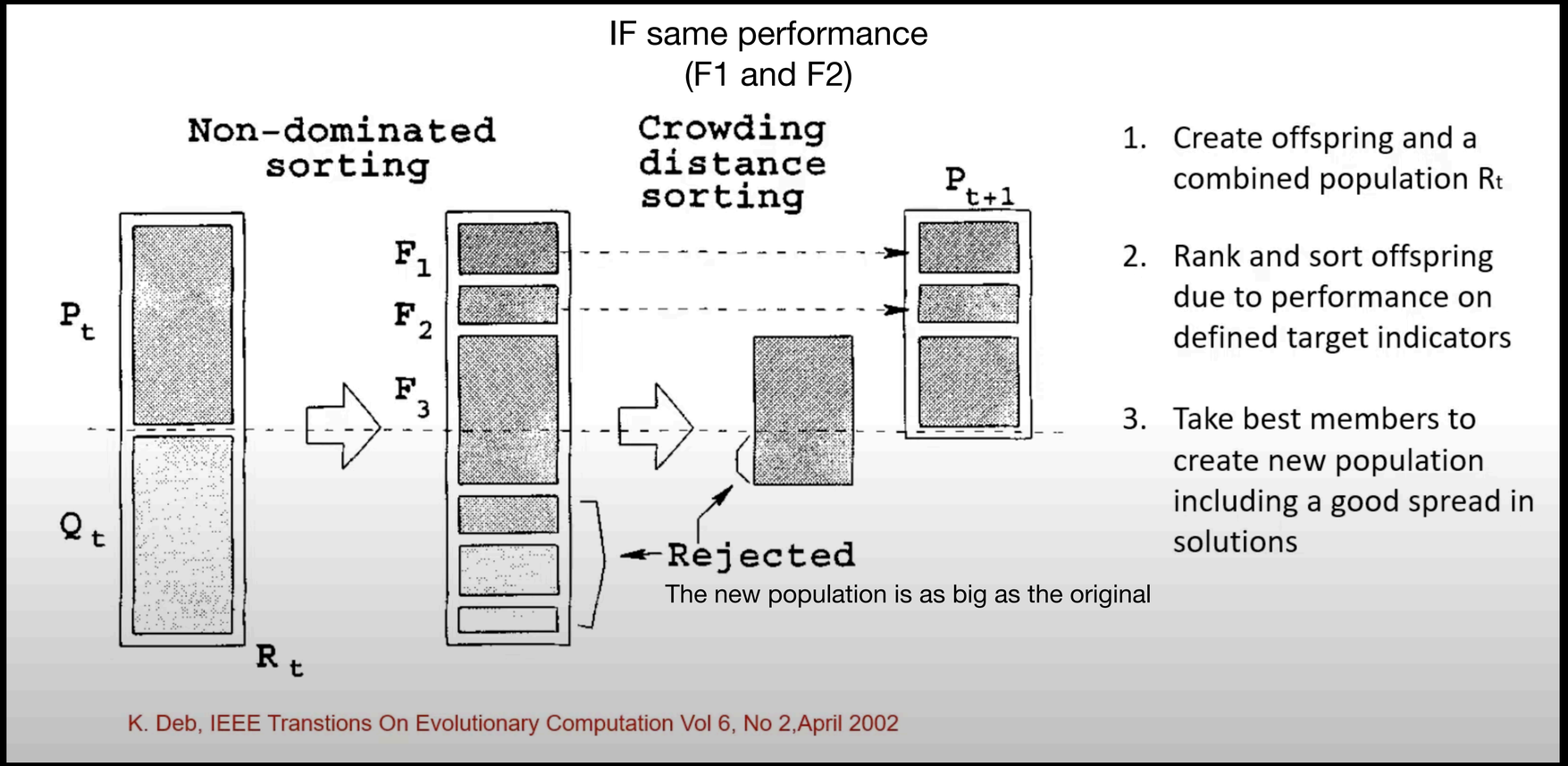
Genetic algorithm

F1 is the leading Pareto front followed by F2 and F3

Original Population

New generated (offspring) Population Q (From genes of Pt)

- Mutations

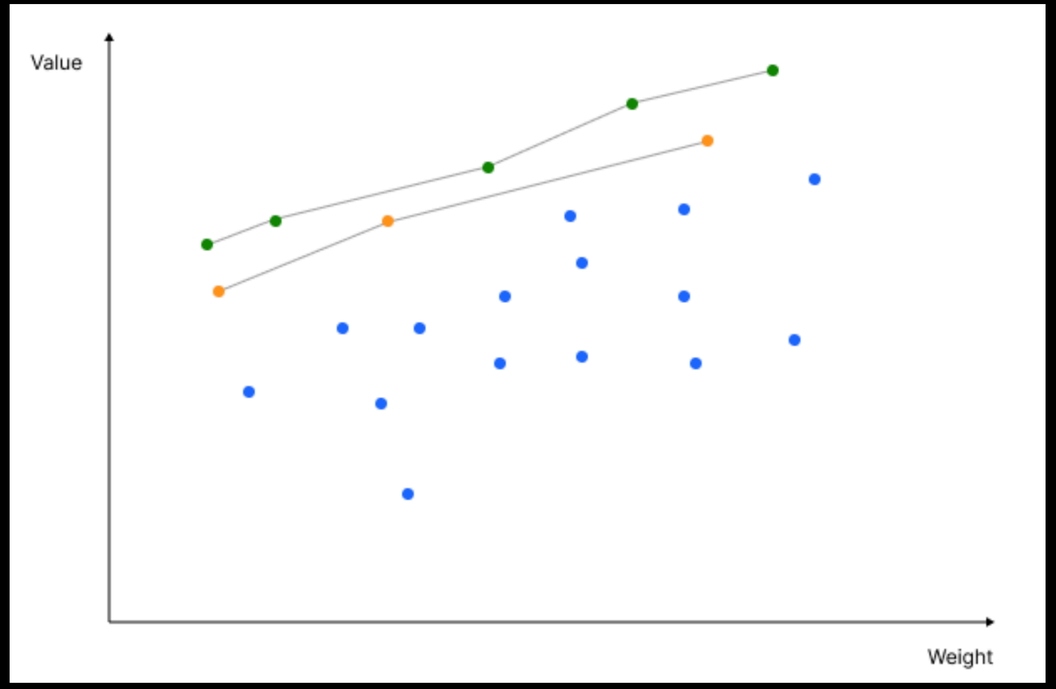


Evaluation of individuals

NSGA 2

Genetic algorithm

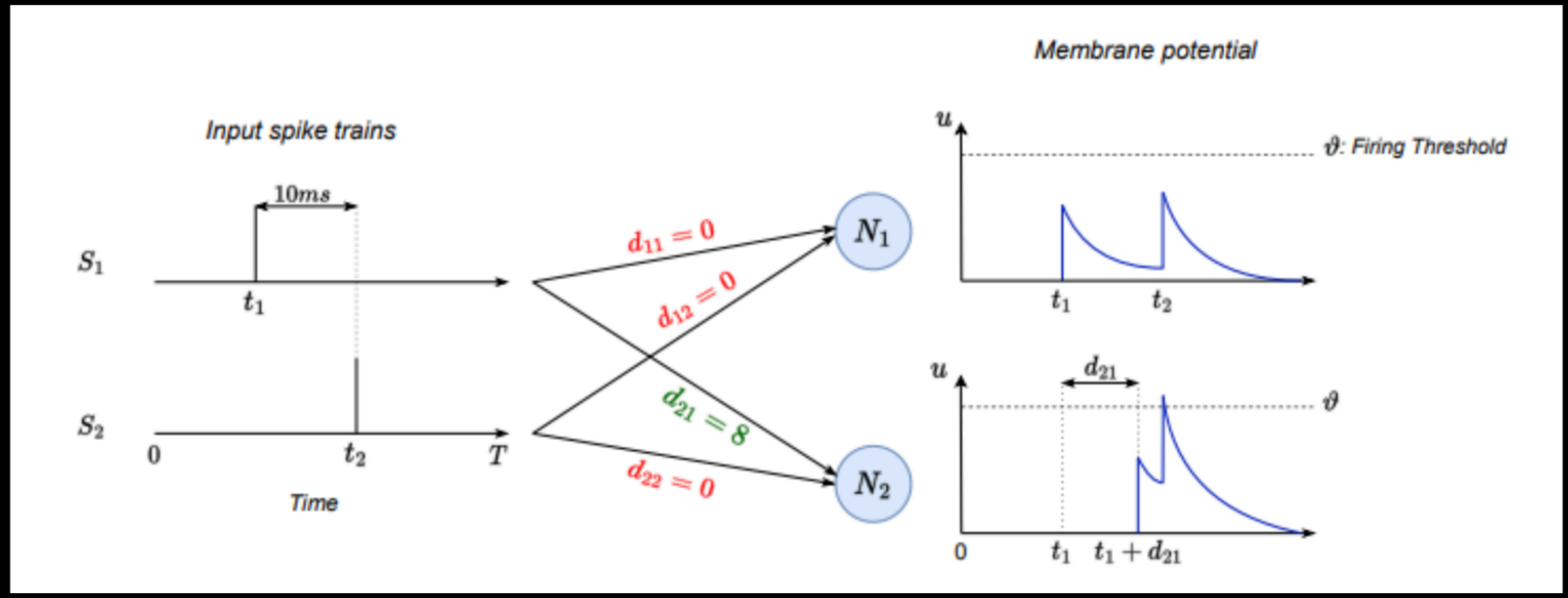
- How does it really work?
- Non dominated sorting:
  - Increase in *weight*, the *value* also increases
  - some individuals have lesser value while being heavier
  - Green points: there is no individual that is both less heavy and more valuable than them
    - Green: all the non-dominated individuals in the graph (Pareto Front 1)



Weights and Decays

Modified Spike-Timing-Dependent Plasticity

- Synaptic delays are **another degree of freedom** that we could exploit
- Delay adaptation to different signals -> improve the specialization

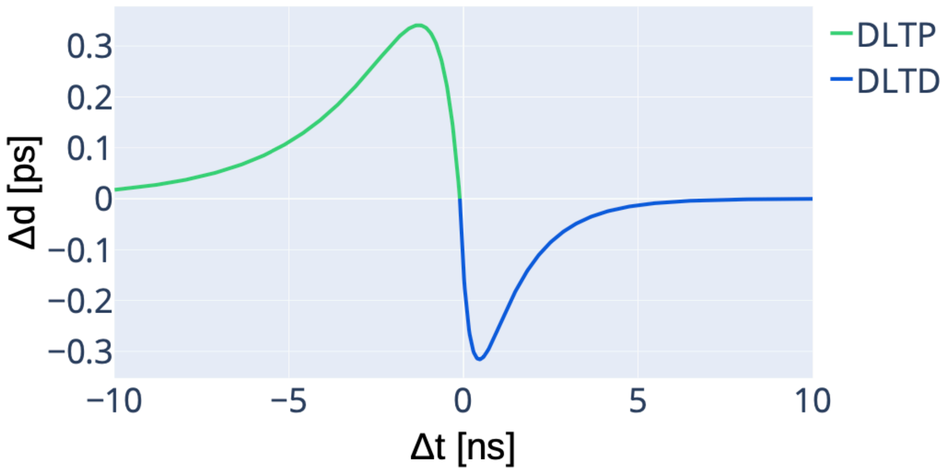


I. Hammouamri et al., “Learning Delays in Spiking Neural Networks using Dilated Convolutions with Learnable Spacings”, arXiv preprint, 2023

Weight and Decay

Modified Spike-Timing-Dependent Plasticity

- Synaptic delays are **another degree of freedom** that we could exploit
- Delay adaptation to different signals -> improve the specialization



Definition 2 (STDP rule for synaptic delays).

$$\Delta d_j = \begin{cases} d_+ \left[\exp\left(\frac{t_j - t_i + t_{max}}{\tau_{d_+}}\right) - \exp\left(\frac{t_j - t_i + t_{max}}{\tau'_{d_+}}\right) \right] & \text{if } t_j \leq t_i - t_{max} \Rightarrow DLTP, \\ -d_- \left[\exp\left(\frac{t_i - t_j - t_{max}}{\tau_{d_-}}\right) - \exp\left(\frac{t_i - t_j - t_{max}}{\tau'_{d_-}}\right) \right] & \text{if } t_j > t_i - t_{max} \Rightarrow DLTD, \\ 0 & \text{if synapse } j \text{ is linking two neurons} \end{cases}$$

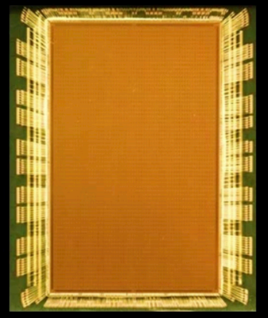
where t_j denotes the pre-synaptic spike arrival time, t_i represents the neuron's activation time, t_{max} indicates the EPSP signal's peak time, d_+ and d_- are the learning rates, and τ_{d_+}, τ'_{d_+}, τ_{d_-}, and τ'_{d_-} are the time constants for potentiation and depression, respectively. The rule updates the delay of synapses linking afferents to neurons.

How much power do they consume?

Energy Efficiency

- **Energy Efficiency:** Consumes minimal power during operation
- **Why?** Sparse activity: Neurons energy use is minimal when not spiking.

- Normal CPUs/GPUs can consume tens to hundreds of watts per TOPS for neural-like workloads.
- Neuromorphic chips like Intel's Loihi or IBM's TrueNorth: **milliwatts per TOPS**
  - ~**1000x or more efficient** on specific neural tasks.
- **BUT**: human brain with (~200 trillion synaptic operations/s), high complexity requires about 20W



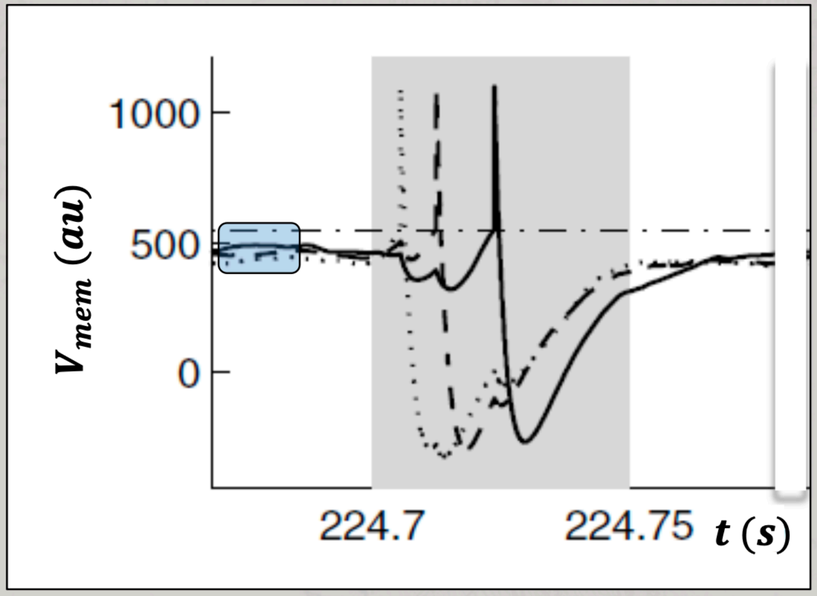
TrueNorth

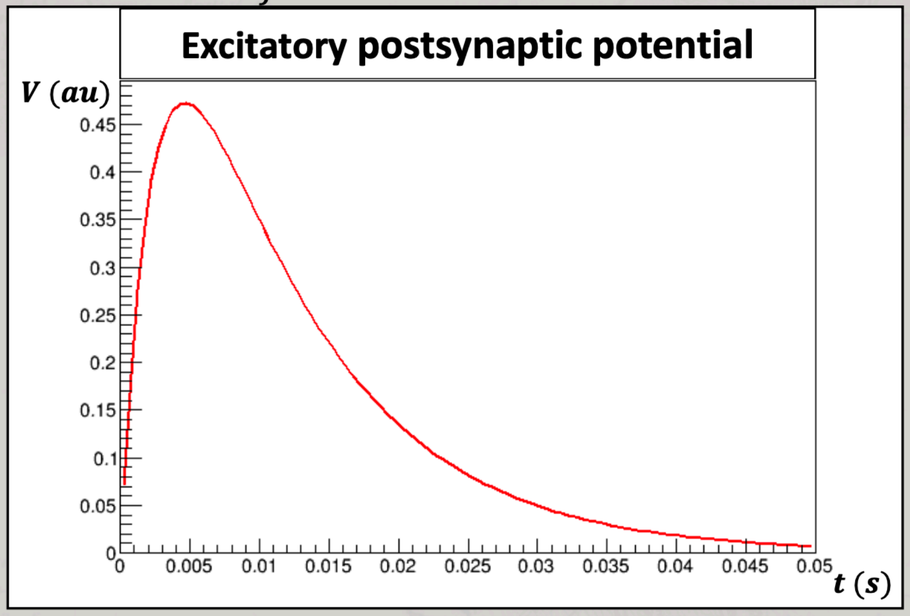
1 M Neurons
256 M Synapses
5.4 B Transistors
Realtime
73 mW

More on Neuron Model

Leaky integrate-and-fire neuron (LIF)

- Input pulse at the synapse → **Excitatory postsynaptic potential** (EPSP)
- $\varepsilon(t - t_j) = K \cdot \left[exp\left(-\frac{t-t_j}{\tau_m}\right) - \exp\left(-\frac{t-t_j}{\tau_s}\right)\right] \cdot \theta(t - t_j)$
- $\boldsymbol{\tau_m}$: Membrane characteristic time
- $\boldsymbol{\tau_s}$: Synapse characteristic time
- **K** multiplicative constant
- t_j pulse arrival time





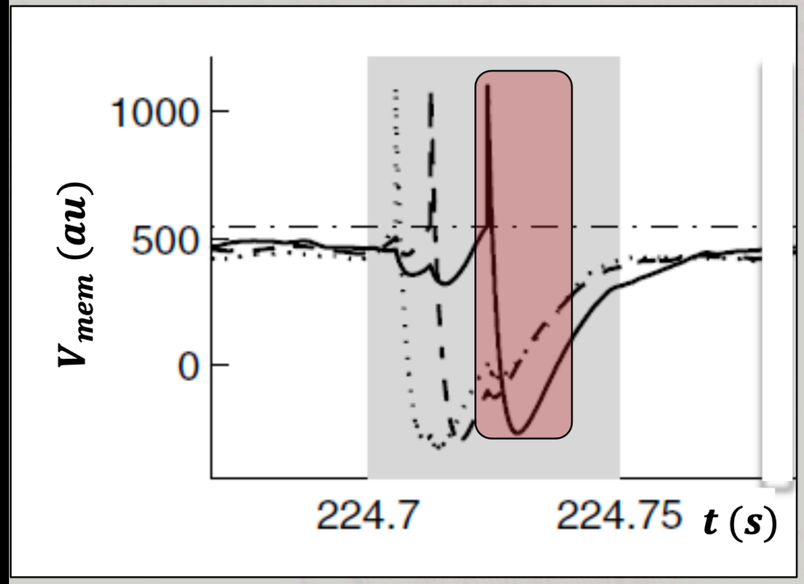
More on Neuron Model

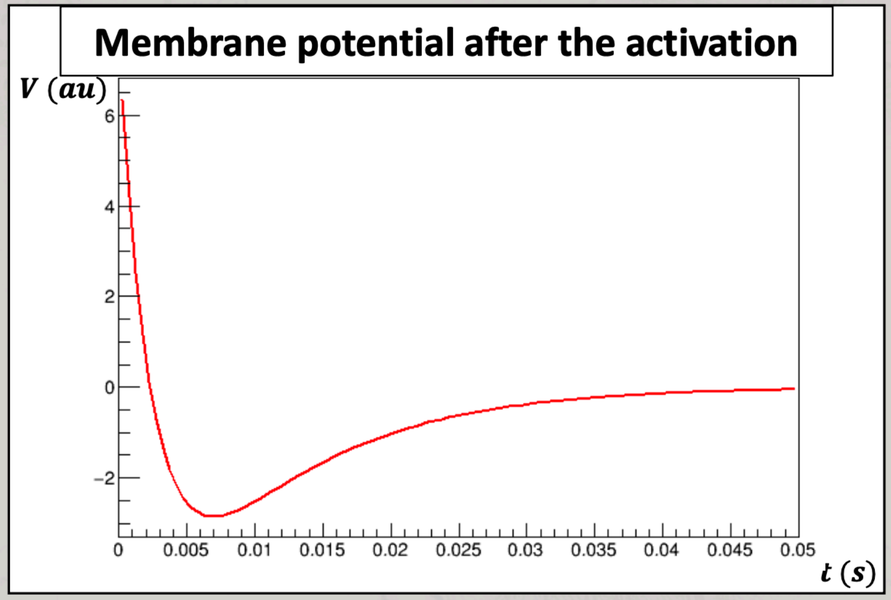
Leaky integrate-and-fire neuron (LIF)

- Potential exceeds threshold **T** → **Neuron activation** → output pulse

- $\eta(t-t_i) = T \cdot \left\{K_1 \cdot exp\left(-\frac{t-t_i}{\tau_m}\right) - K_2 \cdot \left[exp\left(-\frac{t-t_i}{\tau_m}\right) - exp\left(-\frac{t-t_i}{\tau_s}\right)\right]\right\} \cdot \theta(t-t_i)$

- $\boldsymbol{\tau_m}$: Membrane characteristic time
- $\boldsymbol{\tau_s}$: Synapse characteristic time
- $\boldsymbol{K_1, K_2}$ multiplicative shape constants
- t_i activation time



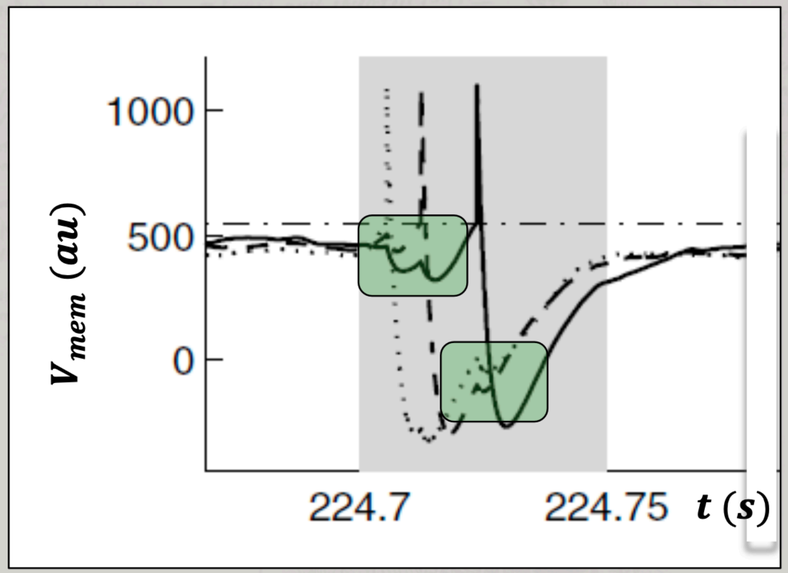


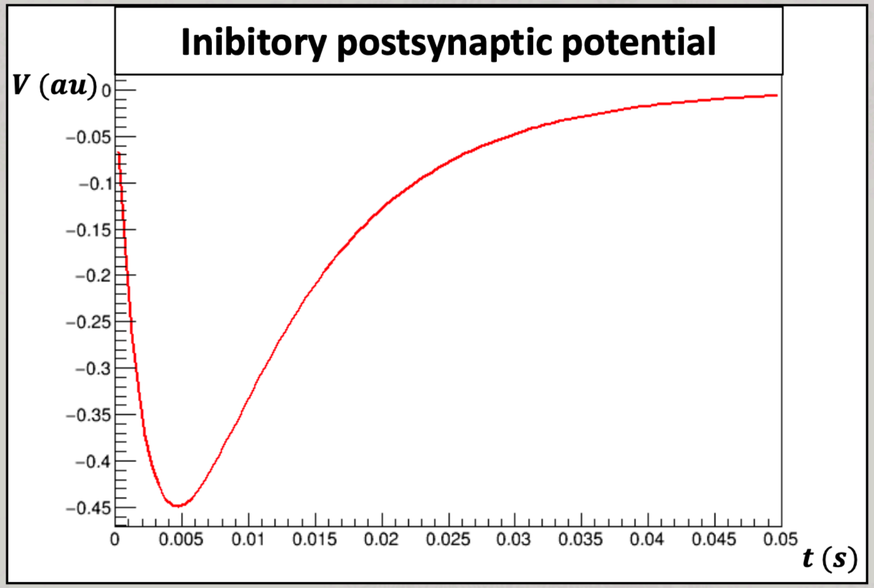
More on Neuron Model

Leaky integrate-and-fire neuron (LIF)

- Neuron activates → It inhibits neurons in its layer
  → **Inhibitory postsynaptic potential** (IPSP)
- $\mu(t - t_k) = -\alpha \cdot T \cdot \varepsilon(t - t_k)$
- $\boldsymbol{\alpha}$ multiplicative intensity constant

Competition among neurons





More on Evaluation Functions

Complete definitions

Definition 3 (Neuron Activation Indicator Function). *This function is used to identify events in which the neuron n has been activated at least once. A neuron is considered "activated at least once" during an event if its membrane potential exceeds the firing threshold T_0 or T_1 within the duration of the event.*

$$1_n(\epsilon) = \begin{cases} 1 & \text{if the neuron } n \text{ has activated at least once during the event } \epsilon \\ 0 & \text{otherwise} \end{cases}$$

Definition 4 (Network Activation Indicator Function).

$$1(\epsilon) = \begin{cases} 1 & \text{if at least one neuron in the network has activated during event } \epsilon \\ 0 & \text{otherwise} \end{cases}$$

Definition 5 (Acceptance per neuron per class). *Measures the fraction of events of class c in which the neuron n has been activated at least once.*

$$A_{n,c} = \frac{\sum_{\epsilon_c} 1_n(\epsilon_c)}{\sum_{\epsilon_c} 1}$$

Definition 6 (Fake rate per neuron). *Measures the fraction of events containing just hits of background in which the neuron n has been activated at least once.*

$$F_n = A_{n,0} = \frac{\sum_{\epsilon_0} 1_n(\epsilon_0)}{\sum_{\epsilon_0} 1}$$

Definition 9 (Selectivity of the network). *Selectivity quantifies the ability of the network to discriminate patterns and is derived from mutual information by comparing the distribution of activations across neurons and particle classes:*

$$S = \sum_{n,c} P_{n,c} \cdot \log_2\left(\frac{P_{n,c} + \delta}{P_c \cdot P_n}\right)$$

with $P_{n,c} = \frac{\sum_{\epsilon_c} 1_n(\epsilon_c)}{N_\epsilon}$, $P_n = \sum_{c=1}^{N_{classes}} P_{n,c}$, $P_c = \sum_{n=1}^{N_{neurons}} P_{n,c}$ *and* $\delta \ll 1$ *to avoid numerical instabilities.*