Spiking Neural Networks: a theoretical framework for Universal Approximation and training

Umberto Biccari
Chair of Computational Mathematics, Universidad de Deusto, Bilbao, Spain
umberto.biccari@deusto.es
cmc.deusto.es

COPI2A meeting
Castro Urdiales, December 1-2 2025

European Commission

European Research Council
Established by the European Commission

Deusto
Universidad de Deusto
Deustuko Unibertsitatea
University of Deusto

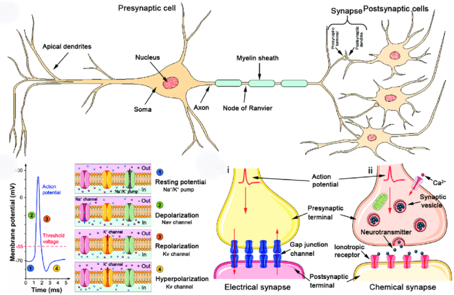
Spiking Neural Networks

Why spiking neurons? A biological perspective

Spikes are the brain's native communication: neurons transmit information via **action potentials**, not continuous activations.

Spiking emerges from **gradual membrane potential evolution**, followed by a **threshold crossing** and **reset** – the basic mechanism behind action potentials.

Neurons fire **only when needed**, producing discrete spikes. SNNs adopt the same event-driven communication principle.



Artificial Neural Networks vs Spiking Neural Networks

Artificial Neural Networks

Continuous activations: neurons output real-valued numbers at every layer.

Synchronous computation: all neurons update at each layer or time step.

Static architecture: matrix multiplications + nonlinearities.

Backpropagation-friendly: smooth activations → differentiability.

Spiking Neural Networks

Event-based outputs: neurons emit spikes (discrete events) when their membrane potential crosses a threshold.

Asynchronous computation: dynamics evolve in continuous time; spikes occur only when needed.

Dynamic architecture: hybrid dynamical system: integration, threshold crossing, reset.

Non-smooth training: requires surrogate gradients or hybrid adjoint formulations.

SNN architecture

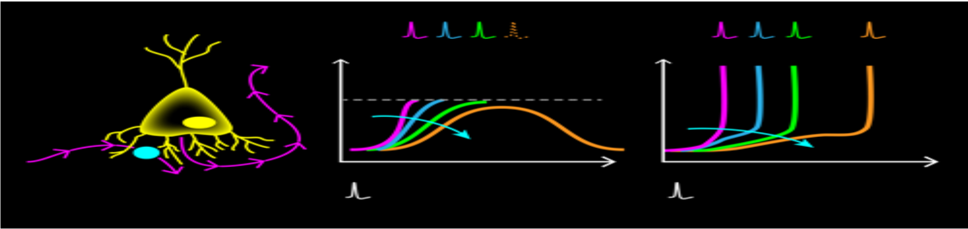
Dataset: $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{0,1\}$ for all $i \in [\![N]\!] := \{1,\ldots,N\}$.

SNN architecture:

- **Input layer** of one neuron with **membrane potential** $v[\boldsymbol{x}_i](t) : \mathbb{R} \to \mathbb{R}$
- $L \geq 1$ **hidden layers** each one of $P \geq 1$ neurons with membrane potentials

$$\boldsymbol{\xi}_1 = \boldsymbol{\xi}_1[v](t) : \mathbb{R} \to \mathbb{R}, \quad \boldsymbol{\xi}_{\ell+1} = \boldsymbol{\xi}_{\ell+1}[\boldsymbol{\xi}_\ell](t) : \mathbb{R} \to \mathbb{R} \quad \text{for all } \ell \in [\![L-1]\!]$$

- **Output layer** of P neurons with potentials $u_p[\xi_{L,p}](t) : \mathbb{R} \to \mathbb{R}$.
- **Readout layer** with one neuron performing classification.



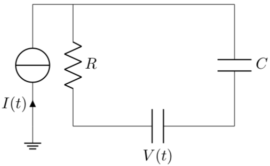
Modeling of the membrane potentials

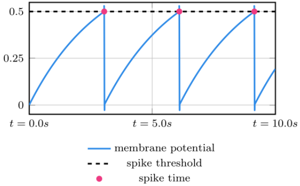
In brain neurons, the potential arises from the flow of ions across the cell membrane, while incoming signals arrive as synaptic currents. When the accumulated depolarization crosses a critical threshold, the neuron emits a spike and reset the potential;

Leaky Integrate-and-Fire (LIF)

Neurons are modeled as **low-pass filter circuits** with one resistor and one capacitor:

- the capacitor mimics the ability of the membrane to store charge;
- the resistor accounts for the natural leakage of ions through open channels;
- synaptic inputs are modeled as external currents charging the circuit.





Hodgkin & Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve*, 1952

Lapicque, *Recherches quantitatives sur l'excitation electrique des nerfs*, 1907

Gerstner & Kistler, *Spiking neuron models: single neurons, populations, plasticity*, 2002

Modeling of the membrane potentials

Input layer

$$\begin{cases} \dot{v}(t) = \dfrac{1}{\tau_v}\Big(- v(t) + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \Big), & t \in (0, T) \setminus \mathbb{T}_0 \\ v(0) = 0 \\ v_k^- = \theta_v, \quad v_k^+ = 0 \end{cases} \tag{In}$$

- $\theta_v > 0$: spike threshold;
- $\tau_v = R_v C_v$, with (R_v, C_v) denoting the neuron's resistor and capacitor;
- $\boldsymbol{a} \in \mathbb{R}^d$ is a trainable parameter modulating the input strength;
- $\mathbb{T}_0 = \Big\{ t_{0,k} \in (0, T) \, : \, k \in \mathbb{N} \text{ and } v(t_{0,k}) = \theta_v \Big\}$ is the sequence of spike times
- $v_k^\pm = \lim_{t \to t_{0,k}^\pm} v(t)$

Modeling of the membrane potentials

Hidden layers

$$\begin{cases} \dot{\xi}_{\ell,p}(t) = \dfrac{1}{\tau_{\ell,p}}\Big(-\xi_{\ell,p}(t) + \omega_{\ell,p} J_\ell(t)\Big), \quad t \in (0,T) \setminus \mathcal{T}_{\ell,p} \\ \xi_{\ell,p}(0) = 0 \\ \xi^-_{\ell,p,k} = \theta_{\ell,p}, \quad \xi^+_{\ell,p,k} = 0 \end{cases} \tag{Hidd}$$

- $\theta_{\ell,p} > 0$: spike threshold
- $\tau_{\ell,p} = R_{\ell,p} C_{\ell,p}$
- $\boldsymbol{\omega}_\ell = (\omega_{\ell,1}, \ldots \omega_{\ell,P}) \subset \mathbb{R}^P$: trainable parameters
- $\mathcal{T}_{\ell,p} = \Big\{ t_{\ell,p,k} \in (0,T) \;:\; k \in \mathbb{N} \text{ and } \xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p} \Big\}$: sequence of spike times
- $\xi^\pm_{\ell,p,k} = \lim_{t \to t^\pm_{\ell,p,k}} \xi_{\ell,p}(t)$
- $J_\ell(t) = \sum_{t^* \in \mathbb{T}_{\ell-1}} \mathcal{G}_{\mu,t^*}(t)$: input current with

$$\mathcal{G}_{\mu,t^*}(t) = \frac{1}{\mu\sqrt{2\pi}} e^{-\frac{(t-t^*)^2}{2\mu^2}} \quad \text{and} \quad \mathbb{T}_{\ell-1} = \bigcup_{p=1}^{P} \mathcal{T}_{\ell-1,p}$$

Modeling of the membrane potentials

The model (Hidd) is a mollification of the hybrid ODE

$$\begin{cases} \dot{\xi}_{\ell,p}(t) = \dfrac{1}{\tau_{\ell,p}} \left(-\xi_{\ell,p}(t) + \omega_{\ell,p} \displaystyle\sum_{t^* \in \mathbb{T}_{\ell-1}} \delta(t - t^*) \right), \quad t \in (0,T) \setminus \mathcal{T}_{\ell,p} \\ \xi_{\ell,p}(0) = 0 \\ \xi^-_{\ell,p,k} = \theta_{\ell,p}, \quad \xi^+_{\ell,p,k} = 0 \end{cases}$$

which models more faithfully the impulsive nature of spike transmission

- due to the Dirac deltas, this equation is not well-posed in standard ODE theory;
- mollification is justified by the weak-$*$ convergence of $\mathcal{G}_{\mu,t^*}(t)$ to $\delta(t-t^*)$;
- one may still write a **formal** by interpreting each delta impulse as producing an instantaneous in the membrane potential. If $\{t^*\}$ are the input spike times

$$\xi_{\ell,p,\delta}(t) = \frac{\omega_{\ell,p}}{\tau_{\ell,p}} \sum_{t^* \in \mathbb{T}_{\ell-1}} e^{-\frac{t-t^*}{\tau_{\ell,p}}},$$

interleaved with threshold-triggered resets.

Modeling of the membrane potentials

Output layer

$$\begin{cases} \dot{u}_p(t) = \frac{1}{\tau_u}\Big(- u_p(t) + w\Phi_p(t)\Big) & t \in (0,T) \\ u_p(0) = 0 \end{cases}, \quad \text{for all } p \in [\![P]\!], \qquad \text{(Out)}$$

- $\Phi_p(t) \coloneqq \sum_{t^* \in \mathbb{T}_{L,p}} \mathcal{G}_{\mu,t^*}(t)$ input current;
- $w \in \mathbb{R}$ a trainable parameter common to each postsynaptic neuron.

Also in this case, (Out) is a smooth approximation of

$$\begin{cases} \dot{u}_p(t) = \frac{1}{\tau_u}\left(-u_p(t) + w \sum_{t^* \in \mathbb{T}_{L,p}} \delta(t - t^*) \right), & t \in (0,T) \\ u_p(0) = 0 \end{cases}$$

Modeling of the membrane potentials

Readout layer

The final stage is a static readout layer, acting on the membrane potentials produced by the output neurons at the final time T

$$\mathcal{R}(\boldsymbol{u}(T)) := \sum_{p=1}^{P} \nu_p \sigma\left(u_p(T) - \theta_u\right),$$

- $\boldsymbol{u} = (u_1, \ldots, u_P)$ denotes the vector of output potentials;
- $\sigma : \mathbb{R} \to \mathbb{R}$ is a Lipschitz non-polynomial activation function;
- $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_P) \in \mathbb{R}^P$ are trainable parameters.

This readout realizes a **Cybenko-type ansatz**: the continuous features $u_p(T)$ generated by the dynamical system are non-linearly combined with trainable weights to produce the final network output. The choice of this structure is crucial, since it ensures the **universal approximation property** of the architecture.

Leshno, Lin, Pinkus & Schocken. *Multilayer feedforward networks with a nonpolynomial activation function can approximate any function*, 1993

Well-posedness

Well-posedness - input layer

Proposition

For all $\boldsymbol{x}_i \in \mathbb{R}^d$, $\boldsymbol{a} \in \mathbb{R}^d$, $0 < \tau_v, \theta_v \in \mathbb{R}$ and $T > 0$ there exists a unique solution $v \in C^\infty\left((0,T)\setminus \mathbb{T}_0\right)$ of the dynamical system (In), where $\mathbb{T}_0 = \{t_{0,k}\} \subset (0,T)$ denotes the sequence of spike times at which $v(t_{0,k}) = \theta_v$. Moreover, $\mathbb{T}_0$ can be characterized as follows:

1. if $\langle \boldsymbol{a}, \boldsymbol{x}_i\rangle \leq \theta_v$, then $\mathbb{T}_0 = \varnothing$;
2. if $\langle \boldsymbol{a}, \boldsymbol{x}_i\rangle > \theta_v$,

$$\mathbb{T}_0 = \begin{cases} \varnothing & \text{if } K = 0 \\ \{t_{0,k}\}_{k=1}^{K} \text{ with } t_{0,k} := k\beta(a) & \text{if } K \geq 1. \end{cases}$$

with

$$K := \lfloor T\beta^{-1} \rfloor \quad \text{and} \quad \beta(\boldsymbol{a}, \boldsymbol{x}_i) := \tau_v \ln\left(\frac{\langle \boldsymbol{a}, \boldsymbol{x}_i\rangle}{\langle \boldsymbol{a}, \boldsymbol{x}_i\rangle - \theta_v}\right)$$

Well-posedness - hidden layers

Proposition

For all $\ell \in [\![L]\!]$, $p \in [\![P]\!]$, $0 < \tau_{\ell,p}, \theta_{\ell,p} \in \mathbb{R}$ and $T > 0$ there exists a unique solution $\xi_{\ell,p} \in C^\infty\big((0,T) \setminus \mathcal{T}_{\ell,p}\big)$ of the dynamical system (Hidd), where $\mathcal{T}_{\ell,p} = \{t_{\ell,p,k}\} \subset (0,T)$ denotes the sequence of spike times at which $\xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p}$. Moreover, $\mathcal{T}_{\ell,p}$ can be characterized as follows:

1. if $\mathbb{T}_{\ell-1} = \cup_{p=1}^{P} \mathcal{T}_{\ell-1,p} = \varnothing$, then $\mathcal{T}_{\ell,p} = \varnothing$;
2. if $\mathbb{T}_{\ell-1} \neq \varnothing$, then every spike time $t_{\ell,p,k} \in \mathcal{T}_{\ell,p}$ satisfies the necessary condition

$$\omega_\ell J_\ell(t_{\ell,p,k}) = \theta_{\ell,p}.$$

Universal approximation

Universal approximation

Theorem

Let ρ be a Borel probability measure with compact support $\Omega \Subset \mathbb{R}^d$, and let $f \in C_0(\Omega) \cap L^2(\rho)$. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a Lipschitz, non-polynomial activation function. Then, for every $\varepsilon > 0$, there exists $L, P \in \mathbb{N}^*$, $\mu > 0$, and a Spiking Neural Network $\mathrm{SNN} : \mathbb{R}^d \to \mathbb{R}$ such that

$$\|\mathrm{SNN} - f\|_{L^2(\rho)} < \varepsilon,$$

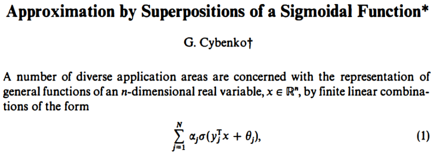
provided

$$\mu \leq \frac{\varepsilon}{C},$$

where $C = C(\sigma, T, w, \tau_u, \boldsymbol{\nu}) > 0$ depends only on the regularity of σ, the time interval T, the parameters w and τ_u, and the readout parameters $\boldsymbol{\nu} = \{\nu_p\}_{p=1}^{P}$.

Universal approximation - proof intuition

Universality comes from the readout layer, which reproduces a **Cybenko-type ansatz** by forming nonlinear combinations of the final output potentials.



Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

A number of diverse application areas are concerned with the representation of general functions of an n-dimensional real variable, $x \in \mathbb{R}^n$, by finite linear combinations of the form

$$\sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + \theta_j), \qquad (1)$$

The internal spiking dynamics transform each input into a structured collection of spike times whose integrated effect at the output layer can be precisely shaped.

By adjusting the parameters of the SNN, these dynamics can generate arbitrarily rich families of exponential-like responses, providing the readout with the features it needs to approximate any continuous target function.

In this, the hidden layers provide an indispensable **encoding mechanism**. The output layer does not receive the input ***x*** directly: it only integrates spike trains. Without hidden layers, there is no mechanism to transform the input into spike patterns that the output neurons can process.

Learning framework

Learning framework

Training functional

$$G(\Upsilon, \boldsymbol{\nu}) = \frac{1}{N} \sum_{i=1}^{N} \text{loss}\Big(\mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i), y_i\Big) + \gamma\Big(\|\Upsilon\|^2 + \|\nu\|^2\Big),$$

- $\Upsilon := \Big(\boldsymbol{a}, (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_L), w\Big) \in \mathbb{R}^d \times \mathbb{R}^{PL} \times \mathbb{R}$
- $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_P) \in \mathbb{R}^P$

Empirical risk minimization

$$(\Upsilon^*, \boldsymbol{\nu}^*) = \arg\min G(\Upsilon, \boldsymbol{\nu})$$

$$G(\Upsilon, \boldsymbol{\nu}) = \frac{1}{N} \sum_{i=1}^{N} \text{loss}\Big(\mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i), y_i\Big) + \gamma\Big(\|\Upsilon\|^2 + \|\nu\|^2\Big)$$

Learning framework

Minimization via gradient-based algorithms $\longrightarrow$ compute $\nabla_{\Upsilon} G$ and $\nabla_{\boldsymbol{\nu}} G$

Easy part: $\nabla_{\nu_p} G = 2\gamma\nu_p + \dfrac{1}{N}\sum_{i=1}^{N} \partial_{z_i}\text{loss}(z_i, y_i)\sigma(u_p(T) - \theta_u)$ for all $p \in [\![P]\!]$

Hard part: for $\nabla_{\Upsilon} G$, the computation is more delicate due to the reset mechanisms which introduces non-differentiability points and jumps in the state trajectories.

- we will apply a **surrogate gradient** approach, based on the mollification of the discontinuous spike events by replacing the hard thresholding in the LIF model with differentiable surrogate functions;
- once the dynamics are regularized, the computation of $\nabla_{\Upsilon} G$ can be rigorously formulated through an optimal control framework.

$$\begin{cases} \dot{X}(t) = \mathcal{F}(X(t), \Upsilon), & t \in (0, T) \setminus \mathbb{T} \\ X(0) = 0 \\ X^+ = \mathcal{J}(X^-) \end{cases} \qquad \mathbb{T} = \bigcup_{\ell=0}^{L} \mathbb{T}_\ell$$

- $X(t) = \left(v(t), \{\boldsymbol{\xi}_\ell(t)\}_{\ell=1}^{L}, \boldsymbol{u}(t)\right)^\top$
- $\mathcal{F}(X(t), \Upsilon) = \left(f_v, \{f_{\ell,1}, \ldots, f_{\ell,P}\}_{p=1}^{P}, \{f_{u_p}\}_{p=1}^{P}\right)^\top,$

Input layer	$f_v := \tau_v^{-1}\big(- v + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \big)$	for all $i \in [\![N]\!]$
Hidden layers	$f_{\xi_{\ell,p}} := \tau_{\ell,p}^{-1}\big(- \xi_{\ell,p} + \omega_{\ell,p} J_\ell \big)$	for all $\ell \in [\![L]\!]$ and $p \in [\![P]\!]$
Output layer	$f_{u_p} := \tau_u^{-1}\big(- u_p + w \Phi_p \big)$	for all $p \in [\![P]\!]$

- $X^+ = \mathcal{J}(X^-)$: **jump conditions**

Learning framework - computation of $\nabla_\Upsilon G$

Duality

$$\nabla_\Upsilon G = 2\gamma\Upsilon - \int_0^T \lambda(t)^\top \partial_\Upsilon \mathcal{F}(X(t), \Upsilon)\, dt.$$

The adjoint states $\lambda(t) = \Big(\lambda_v(t), \{\lambda_{\boldsymbol{\xi}_\ell}(t)\}_{\ell=1}^L, \lambda_{\boldsymbol{u}(t)}\Big)$ formally satisfy the backward dynamics

$$\begin{cases} -\dot{\lambda}(t) = \Big(\partial_{X(t)}\mathcal{F}(X(t), \Upsilon)\Big)^\top \lambda(t), \quad t \in (0, T) \setminus \mathbb{T} \\ \lambda(T) = \partial_{X(T)}\left(\dfrac{1}{N}\displaystyle\sum_{i=1}^N \Psi_i\right) \\ \text{Adjoint jump condition} \end{cases} \qquad \Psi_i = \text{loss}\Big(\mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i), y_i\Big)$$

The **adjoint jump condition** captures how the discontinuous reset of the forward dynamics at spike times affects the backward propagation of sensitivities.

Intuitively, whenever the membrane potential is reset in the forward system, the adjoint variables must also be updated to account for the sudden change in the state trajectory. This ensures that the contribution of each spike event to the overall objective functional is correctly transmitted backward in time.

Learning framework - computation of $\nabla_\Upsilon G$

Mollifier

$$H_\varsigma(s) := \frac{1}{2}\left(1 + \tanh\left(\frac{\varsigma s}{2}\right)\right) : \mathbb{R} \to (0,1)$$

$\varsigma > 0$

Discharge map

$$D_\varsigma(s;\theta) := (1 - H_\varsigma(s-\theta))s$$

As $\varsigma \to +\infty$

$$H_\varsigma(s) \to H(s) = \begin{cases} 1 & \text{if } s > 0 \\ \frac{1}{2} & \text{if } s = 0 \\ 0 & \text{if } s < 0 \end{cases} \quad \text{and} \quad D_\varsigma(s;\theta) \to D(s;\theta) = \begin{cases} s & \text{if } s > \theta \\ \frac{\theta}{2} & \text{if } s = \theta \\ 0 & \text{if } s < \theta \end{cases}$$

$$\begin{cases} \dot{v}(t) = \frac{1}{\tau_V}\Big(-v(t) + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle\Big), & (0,T) \setminus \mathbb{T}_0 \\ v(0) = 0, & \\ v(t^{*+}) = \mathcal{D}_\varsigma\left(v(t^{*-}); \theta_v\right), & t^* \in \mathbb{T}_0 \end{cases} \qquad \begin{cases} \dot{\xi}_{\ell,p}(t) = \frac{1}{\tau_{\ell,p}}\Big(-\xi_{\ell,p}(t) + \omega_{\ell,p} J_\ell(t)\Big), & (0,T) \setminus \mathcal{T}_{\ell,p} \\ \xi_{\ell,p}(0) = 0, & \\ \xi_{\ell,p}(t^{*+}) = \mathcal{D}_\varsigma\left(\xi_{\ell,p}(t^{*-}); \theta_{\ell,p}\right), & t^* \in \mathcal{T}_{\ell,p} \end{cases}$$

Learning framework - computation of $\nabla_\Upsilon G$

Proposition

Assume $\dot{v}(t_0^*) \neq 0$ and $\dot{\xi}_{\ell,p}(t_{\ell,p}^*) \neq 0$ for all $t_0^* \in \mathbb{T}_0$ and $t_{\ell,p}^* \in \mathcal{T}_{\ell,p}$.

Let $\{H_\zeta\}_{\zeta>0}$ be a family of smooth reset mollifiers $H_\zeta : \mathbb{R} \to (0,1)$ and $\{D_\zeta(z;\theta)\}_{\zeta>0}$ be the associated discharge maps.

For all $T > 0$, the solutions v_ζ and $\xi_{\zeta,\ell,p}$ to the reset–mollified input and hidden dynamics satisfy

$$\lim_{\zeta\to+\infty} \sup_{t\in(0,T)} |v_\zeta(t) - v(t)| = 0 \quad \text{and} \quad \lim_{\zeta\to+\infty} \sup_{t\in(0,T)} |\xi_{\zeta,\ell,p}(t) - \xi_{\ell,p}(t)| = 0.$$

Proposition

Under the assumptions of the previous proposition, let $\{t_k\}$ denote the spike times of the hybrid input or hidden systems. Then, for each t_k^* there exists a spike time $t_{\zeta,k}$ of the ζ-mollified system such that $t_{\zeta,k} \to t_k$ as $\zeta \to +\infty$. Moreover, the entire spike train of the mollified system converges to that of the hybrid system.

Simulation experiments

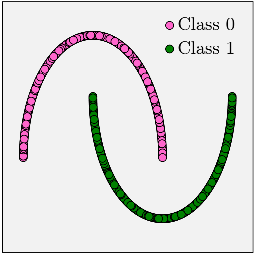
Dataset

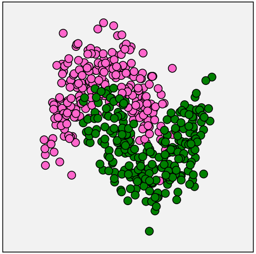
We employ the `make_moons` dataset from the `scikit-learn` library. The data consist of two interleaving half-circles embedded in the plane, which form a nonlinear decision boundary that cannot be captured by linear classifiers. Points are generated by sampling from these two half-moon manifolds and assigning binary labels according to membership.

The dataset can be enriched with Gaussian noise, thereby introducing variability and overlap between the classes. In this spirit, we considered two variants:

- 0% **noise:** the classes are perfectly separable
- 20% **noise:** a significant fraction of the samples are perturbed, yielding overlapping distributions.

In both scenarios, we have split the dataset among 70% **training data**, 5% **validation data**, and 25% **testa data**.





Model

Model

Single hidden layer SNN ($L = 1$) with $P = 8$ neurons. Taking into account the input dimension $d = 2$, this amounts at a total number of only 19 trainable parameters.

Hyper-parameters:

- **Time horizon:** $T = 60s$
- **Input layer:** $\tau_v = 8$ and $\theta_v = 0.8$
- **Hidden layers:** $\tau_{\ell,p} = 6$ and $\theta_{\ell,p} = 0.25$ for all $\ell \in [\![L]\!]$ and $p \in [\![P]\!]$
- **Output layer:** $\tau_u = 10$ and $\theta_u = 0.3$
- **Gaussian mollification:** $\mu = 0.2$

ζ is initialized at a low value $\zeta_0 = 3$ and gradually increased to $\zeta_1 = 10$ over the course of training epochs, according to the update rule

$$\zeta(e) = \zeta_0 \left(\frac{\zeta_1}{\zeta_0} \right)^{\frac{e}{E-1}},$$

where e is the epoch index and E the total number of epochs. This choice yields a progressive sharpening of the surrogate gradient while maintaining stability in the early stages.

Training results

Accuracy: measures the overall proportion of correct predictions

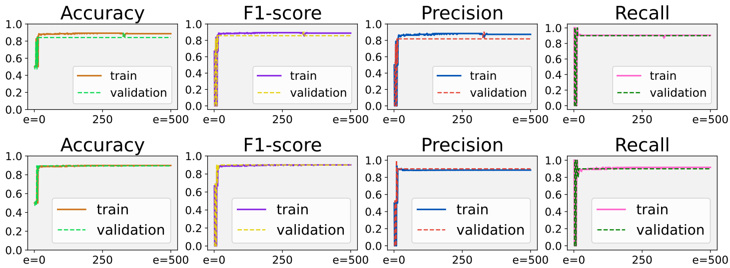
Precision: assesses the reliability of positive predictions

Recall: measures the ability to correctly identify all positive samples

F1-score: harmonic mean of precision and recall, providing a single measure that balances the trade-off between false positives and false negatives

0% noise

20% noise



	Accuracy	Precision	Recall	F1-score
0% noise	0.8889	0.9286	0.8387	0.8814
20% noise	0.8413	0.8889	0.7742	0.8276

Conclusions

Open problems

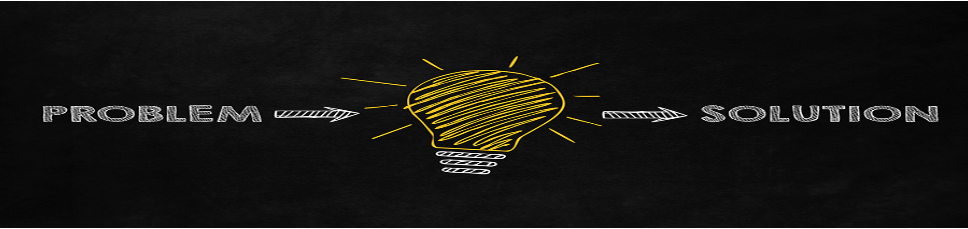
Depth-width tradeoffs in SNNs. While our universality result is constructive, it does not quantify the minimal number of neurons per layer or the necessary depth for approximation within a given error tolerance.

Quantitative approximation rates. Our universal approximation Theorem guarantees density in the space of continuous functions but does not provide explicit approximation rates in terms of the number of neurons, spikes, or layers.

Training under hybrid dynamics. The introduction of mollifiers in the LIF dynamics ensures well-posedness and differentiability, but the resulting surrogate models are only approximations of true spiking dynamics.

Generalization and capacity bounds. Beyond approximation, little is known about the generalization properties of SNNs.

Dynamics beyond LIF models. Our analysis focused on LIF neurons, but biologically realistic neurons exhibit richer dynamics (e.g., adaptive thresholds, conductance-based models, or stochastic firing).



THANKS FOR YOUR ATTENTION!!

The **CoDeFeL** project (ERC-2022-ADG) has received funding from the European Union's Horizon ERC Grants programme under grant agreement No. 101096251. The views and opinions expressed are solely those of the author(s) and do not necessarily reflect those of the European Research Council Executive Agency (ERCEA), the European Union or the granting authority who cannot be held responsible for them.

Additional funding

- PID2023-146872OB-I00-**DyCMaMod** (MINECO)

Paper

- U. Biccari, *Spiking Neural Networks: a theoretical framework for Universal Approximation and training*, 2025 (LINK)

European Research Council

Established by the European Commission

European Commission

GOBIERNO DE ESPAÑA

MINISTERIO DE ECONOMÍA, INDUSTRIA Y COMPETITIVIDAD