Temporal Neural Networks: A Dynamical Systems Approach to Stable and Robust Neural Computation

Edward Chalk
Independent Researcher
edward@fleetingswallow.com
https://github.com/pcoz

January 2025

Abstract

We present Temporal Neural Networks (TNNs), a biologically-inspired architecture where each neuron is modeled as a continuous-time dynamical system rather than an instantaneous function. Unlike classical neural networks that compute $y = f(x)$ instantaneously, TNN neurons evolve according to $dV/dt = (1/\tau)(-V+f(Wx+b))$, introducing temporal dynamics and internal state. We demonstrate a three-phase pipeline: (1) classical training, (2) symbolic regression via PPF to discover interpretable temporal dynamics, and (3) conversion to temporal form. On the UCI Human Activity Recognition benchmark with proper subject-based splits, TNNs match classical accuracy (95.1% vs 95.3%) while exhibiting dramatically improved temporal stability (75–91% fewer prediction flips under noise) and superior robustness to missing data (+8.4% accuracy at 40% dropout). These properties—more stable decisions over time, fewer false alarms, and graceful degradation—are directly relevant to clinical and industrial deployment where sensor unreliability and alarm fatigue are critical concerns.

1 Introduction

Classical neural networks model neurons as instantaneous functions: given input x, compute output $y = f(Wx + b)$ with no temporal dynamics. This abstraction, while computationally convenient, diverges fundamentally from biological neural computation where neurons exhibit membrane potential dynamics, time constants, and temporal filtering [7].

We propose **Temporal Neural Networks (TNNs)**, where each neuron is a continuous-time dynamical system:

$$\frac{dV}{dt} = \frac{1}{\tau}\left(-V + f(Wx + b)\right) \tag{1}$$

where V is the neuron's membrane potential (state), τ is the time constant, and the neuron *evolves toward* its target activation rather than jumping instantaneously.

This seemingly simple change has profound implications: the network now *exists in time*, maintains *internal state*, and exhibits *temporal inertia*—properties that provide natural robustness to noise and sensor dropout.

1.1 Contributions

1. A three-phase pipeline for converting classical networks to temporal form: training, symbolic form discovery via PPF, and temporal conversion

2. Demonstration that TNNs match classical accuracy while providing 75–91% fewer prediction flips under noise

3. Evidence of superior robustness: +8.4% accuracy retention at 40% feature dropout

4. Analysis of clinical relevance: alarm stability, sensor robustness, graceful degradation

2 Related Work

2.1 Neural Ordinary Differential Equations

Neural ODEs [1] treat network depth as a continuous variable, replacing discrete layers with differential equation solvers. While powerful, they face challenges including high computational cost and sensitivity to adversarial inputs. Closed-form continuous-time networks [2] address computational cost through analytical solutions, achieving 1–5 orders of magnitude speedup.

Our approach differs by operating at the *neuron level* rather than network depth, using simple leaky integration without ODE solvers, and focusing on inference-time temporal dynamics.

2.2 Spiking Neural Networks

SNNs achieve robustness through temporal processing [3], with research showing they can surpass traditional ANNs by leveraging temporal dynamics. The geometry of SNN robustness [4] demonstrates that networks become robust when voltages are confined to lower-dimensional subspaces.

Our TNN approach captures similar benefits while remaining compatible with standard backpropagation and providing smoother dynamics suitable for regression tasks.

2.3 Symbolic Regression for Neural Dynamics

Recent work demonstrates that symbolic regression can discover interpretable governing equations from neural network dynamics [5, 6]. We integrate this approach through PPF (Partial Form Finder) to discover per-neuron temporal dynamics.

2.3.1 PPF: Partial Form Finder

PPF (Partial Form Finder) is a symbolic regression tool developed by Chalk[1] specifically designed for discovering mathematical forms from time series data. Unlike general-purpose symbolic regression tools, PPF is optimized for:

- **Partial matching**: Discovering forms that explain portions of the data, not requiring a single global equation
- **Temporal patterns**: Specialized primitives for oscillations, exponential decay, and dynamical systems
- **Interpretable output**: Producing human-readable equations rather than black-box approximations

PPF uses genetic programming to evolve candidate expressions, evaluating fitness based on R^2 correlation with the target time series. The search space includes:

[1] https://github.com/pcoz/timeseries-formula-finder

- Trigonometric functions: $\sin(\omega t + \phi)$, $\cos(\omega t)$
- Exponential forms: $e^{-t/\tau}$, $e^{\alpha t}$
- Rational functions: $(at + b)/(ct + d)$
- Compositions: damped oscillations, modulated signals

For TNN form discovery, PPF analyzes neuron activation trajectories and returns symbolic expressions characterizing the temporal dynamics, from which we extract parameters such as natural frequency ω, damping coefficient ζ, and time constant τ.

3 Method

3.1 Phase 1: Classical Training

We begin with a standard feedforward network trained via backpropagation:

$$h^{(l)} = \sigma(W^{(l)} h^{(l-1)} + b^{(l)}) \tag{2}$$

This phase establishes the network's learned representations without temporal dynamics.

3.2 Phase 2: Form Discovery via PPF

We record continuous activations as the trained network processes temporal data, then apply symbolic regression to discover mathematical forms governing neuron dynamics. PPF can identify patterns such as:

- Damped oscillations: $Ae^{-t/\tau}\sin(\omega t + \phi)$
- Exponential decay: $Ae^{-t/\tau} + B$
- Rational functions: $(at + b)/(ct + d)$

Discovered forms provide interpretable, task-appropriate dynamics rather than uniform arbitrary parameters.

3.3 Phase 3: Temporal Conversion

Each neuron is converted to temporal form using Equation 1. The time constant τ can be:

- Uniform across the network
- Layer-specific
- Per-neuron (from PPF discovery)
- Learnable during fine-tuning

3.4 Inference

At inference time, the network processes inputs through multiple "settle steps":

Reset all neuron states $V \leftarrow 0$
for $t = 1$ to T_{settle} **do**
  for each layer l **do**
    target $\leftarrow \sigma(W^{(l)} h^{(l-1)} + b^{(l)})$
    $V^{(l)} \leftarrow V^{(l)} + \frac{dt}{\tau}(\text{target} - V^{(l)})$
  end for
end for
return $\arg\max(V^{(L)})$

4 Experiments

4.1 Dataset

We use the UCI Human Activity Recognition dataset with **proper subject-based splits**:

- Training: 7,352 samples from 21 subjects
- Testing: 2,947 samples from 9 subjects
- **Zero subject overlap**—no data leakage
- 6 activities: Walking, Walking Upstairs/Downstairs, Sitting, Standing, Laying

4.2 Models

- **Classical**: $[561, 128, 64, 6]$ with tanh activation
- **Temporal**: Same architecture with leaky integration ($\tau = 8.0$)

4.3 Results

4.3.1 Baseline Accuracy

Model	Accuracy	Macro F1
Classical	95.3%	0.952
Temporal	95.1%	0.951

Table 1: Baseline performance on clean data. TNN matches classical accuracy.

4.3.2 Stability Under Noise

We measure **prediction flip rate**—how often the model changes its prediction across consecutive evaluations under noise.

Noise σ	Classical Flips	TNN Flips	Reduction
0.0	0.0	0.9	—
0.2	1.1	0.9	16%
0.3	1.8	0.9	49%
0.5	3.7	0.9	**75%**
0.7	6.3	1.0	**84%**
1.0	11.0	1.0	**91%**

Table 2: Prediction flip rates under Gaussian noise. TNN shows 75–91% fewer flips.

Noise σ	Classical Acc	TNN Acc	Δ
0.0	98.3%	98.3%	0%
0.3	96.3%	99.0%	+2.7%
0.5	93.0%	99.0%	**+6.0%**
0.7	92.7%	99.0%	+6.3%
1.0	83.3%	97.7%	**+14.4%**

Table 3: Accuracy under noise. TNN is not just more stable—it is more accurate.

4.3.3 Robustness to Missing Data

5 Discussion

5.1 Why Stability Matters

A 75–91% reduction in prediction flips is not marginal—it is a **qualitative behavioral difference**. The classical network oscillates under noise; the TNN maintains coherent predictions.

This directly addresses **alarm fatigue** in clinical settings, where flickering predictions cause:

- False alarms overwhelming clinicians
- Reduced trust in automated systems
- Potential patient safety issues

5.2 Why This Is Not "Just Smoothing"

Post-hoc temporal smoothing reduces flips but *also reduces accuracy.* Our TNN:

- Reduces flips **and** improves accuracy under noise
- Handles missing data through temporal integration
- Has dynamics built into computation, not applied after

This combination cannot be achieved by post-hoc smoothing alone.

Dropout %	Classical Acc	TNN Acc	Δ
0%	97.0%	96.8%	-0.2%
20%	94.2%	96.4%	+2.2%
30%	91.6%	95.4%	+3.8%
40%	86.0%	94.4%	**+8.4%**
50%	81.2%	89.0%	+7.8%
60%	76.0%	82.8%	+6.8%

Table 4: Accuracy with feature dropout. TNN degrades 33% more gracefully.

5.3 Biological Plausibility

The TNN equation (Eq. 1) is the **Leaky Integrate-and-Fire model** used throughout computational neuroscience [7]. It captures key features of biological neurons: integration of inputs, leak toward resting potential, and temporal filtering.

Real neural circuits trade instantaneous responsiveness for temporal stability—exactly what we observe in TNNs.

6 Conclusion

Temporal Neural Networks demonstrate that modeling neurons as dynamical systems provides substantial practical benefits: matching classical accuracy while delivering dramatically improved stability (75–91% fewer prediction flips) and robustness (+8.4% accuracy at 40% dropout).

These properties—more stable decisions over time, fewer false alarms, and graceful degradation—are directly relevant to clinical and industrial deployment.

Classical neural networks are *accurate but brittle*. Temporal neural networks are *accurate and well-behaved in time*. This is not a tweak; it is a different computational ontology.

Code Availability

Code is available at: https://github.com/pcoz/temporal-neural-networks

References

[1] Chen, R.T.Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural Ordinary Differential Equations. In *NeurIPS*, 2018.

[2] Hasani, R., Lechner, M., Amini, A., et al. Closed-form continuous-time neural networks. *Nature Machine Intelligence*, 4:992–1003, 2022.

[3] Wang, W., et al. Neuromorphic computing paradigms enhance robustness through spiking neural networks. *Nature Communications*, 16:65197, 2025.

[4] Rullán Buxó, C.E. and Bhatt, P. The geometry of robustness in spiking neural networks. *eLife*, 11:e73276, 2022.

[5] Yang, Z., et al. Learning interpretable network dynamics via universal neural symbolic regression. *Nature Communications*, 16:61575, 2025.

[6] Cranmer, M., et al. Discovering Symbolic Models from Deep Learning with Inductive Biases. In *NeurIPS*, 2020.

[7] Gerstner, W. and Kistler, W.M. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[8] Fang, W., et al. Incorporating Learnable Membrane Time Constant to Enhance Learning of Spiking Neural Networks. In *ICLR*, 2021.

[9] Kar, K., et al. Recurrent neural network dynamical systems for biological vision. *PNAS*, 2025.

[10] Yamazaki, K., et al. Spiking Neural Networks and Their Applications: A Review. *Brain Sciences*, 12(7):863, 2022.