Advancing Spiking Neural Networks for Sequential Modeling with Central Pattern Generators

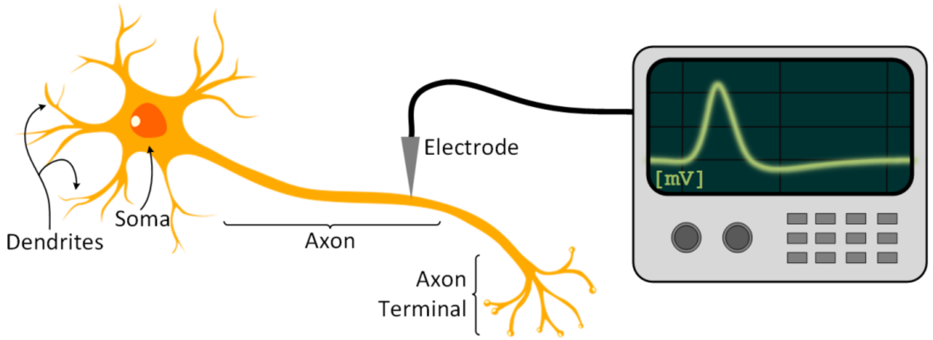
Changze Lv[1], Dongqi Han[2], Yansen Wang[2], Xiaoqing Zheng[1], Xuanjing Huang[1], Dongsheng Li[2]

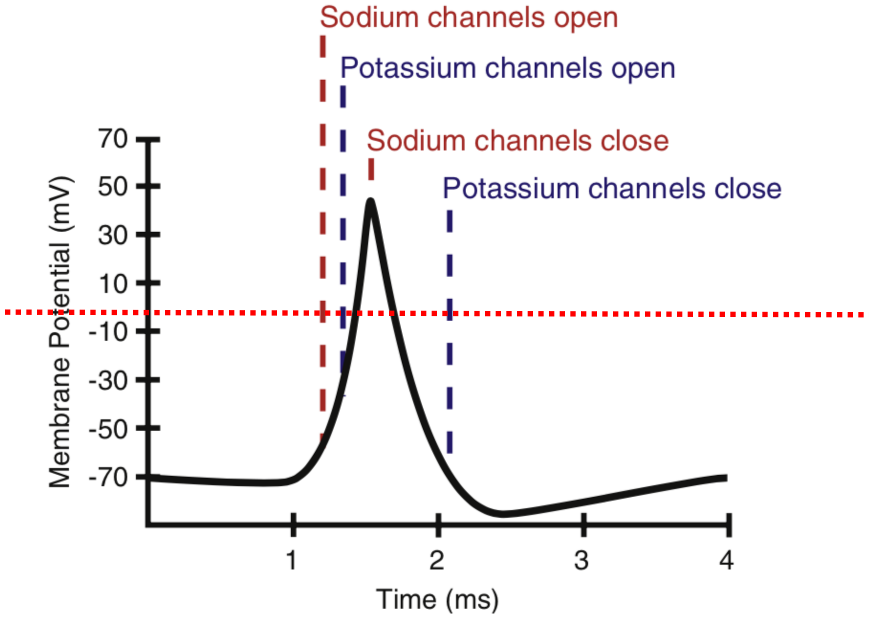
[1]Fudan University, [2]Microsoft Research Asia

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

Spike Neuron





Threshold Potential

1

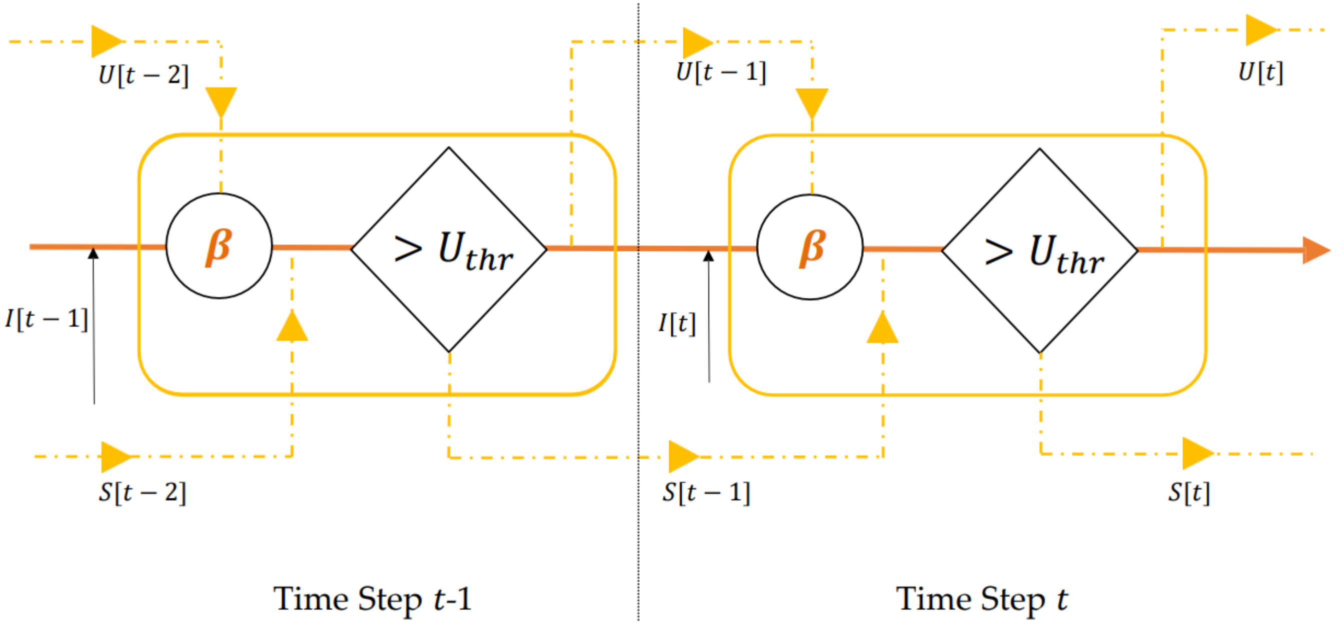
0

Information in Spiking Neurons

- Floating-point Numbers ✖
- Binary Value ✔

"Training Spiking Neural Networks Using Lessons From Deep Learning."
Eshraghian, Jason Kamran et al. 2021

Spiking Neural Networks

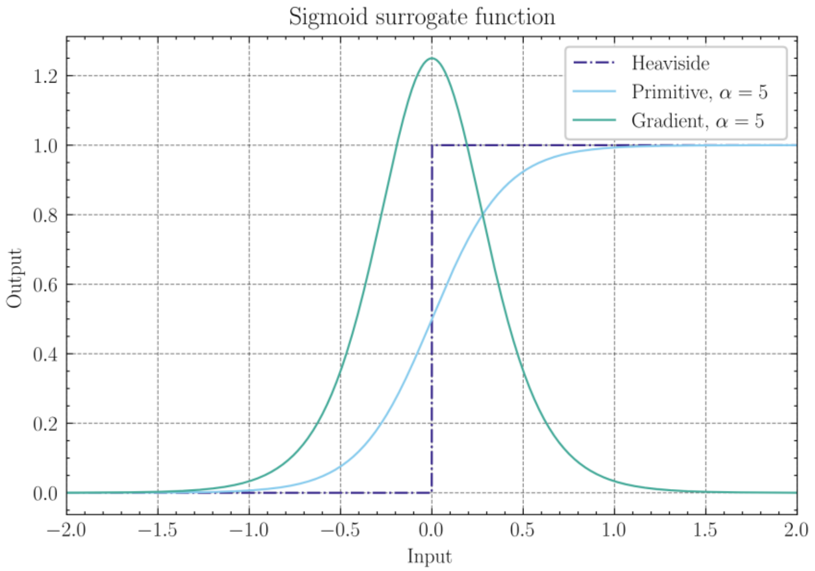


$$U(t) = H(t - \Delta t) + I(t), \quad I(t) = f(\mathbf{x}; \theta),$$

$$H(t) = V_{reset} S(t) + (1 - S(t))\, \beta U(t),$$

$$S(t) = \begin{cases} 1, & \text{if } U(t) \geq U_{\text{thr}} \\ 0, & \text{if } U(t) < U_{\text{thr}} \end{cases},$$

$$S(t) \approx \frac{1}{\pi} \arctan\left(\frac{\pi}{2} \alpha U(t)\right) + \frac{1}{2}$$

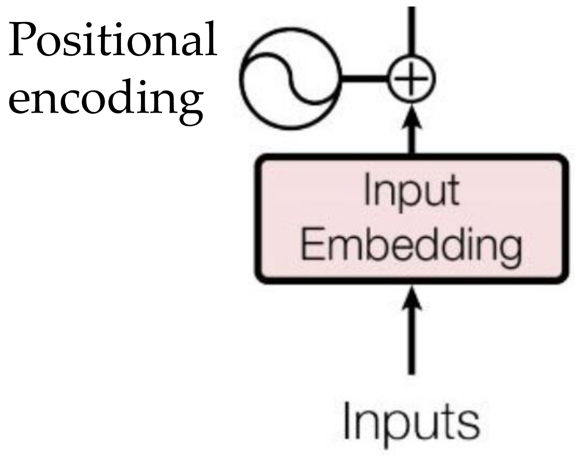


https://spikingjelly.readthedocs.io/

SNN version of Transformer

- In Transformer models, positional encoding:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right),$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right)$$



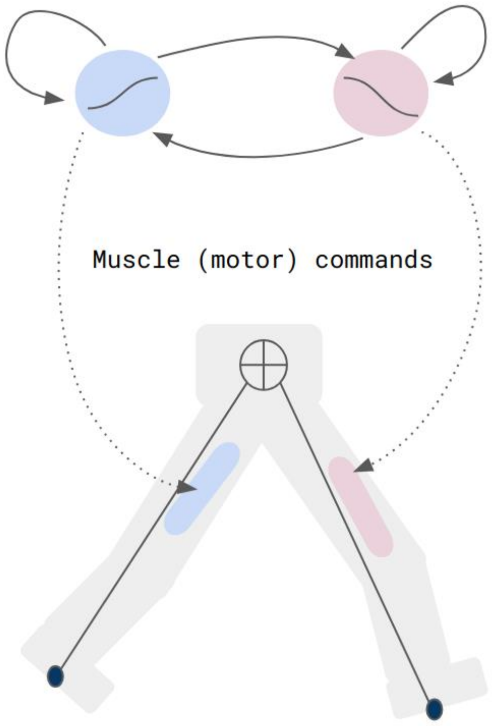
- However, in the SNN version of Transformer (Spikformer), there has not been a good mechanism for Positional Encoding (PE).
  - Positional encoding neurons in Transformer outputs float numbers.
  - The output of SNN neurons is 1 or 0

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

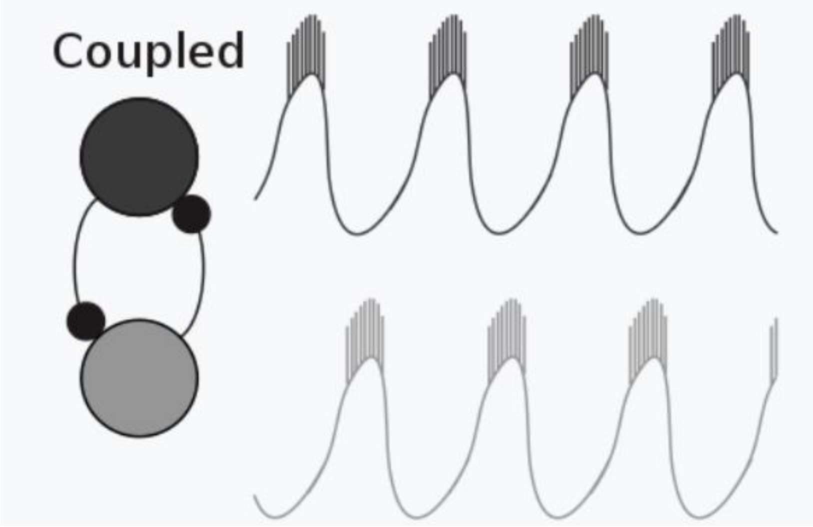
Central Pattern Generators

Central Pattern Generators (CPG) is a group of neurons capable of producing rhythmic patterned outputs without requiring rhythmic inputs.



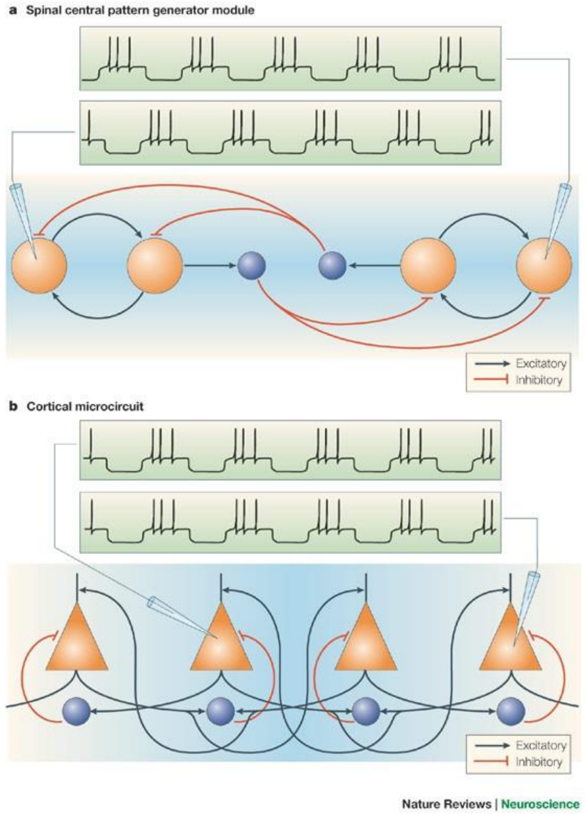
These neural circuits are responsible for generating the rhythmic signals that control vital activities such as locomotion, respiration, and chewing

Simple Model



Marder, Eve; Bucher, Dirk (2001-11-27). "Central pattern generators and the control of rhythmic movements". Current Biology. 11 (23): R986–R996

Comprehensive model



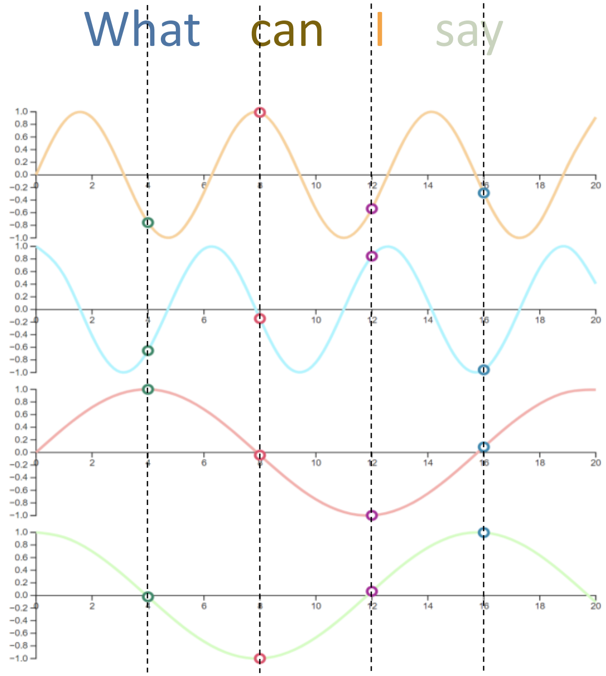
Yuste, R., MacLean, J., Smith, J. et al. The cortex as a central pattern generator. Nat Rev Neurosci 6, 477–483 (2005). https://doi.org/10.1038/nrn1686

Positional Encoding in Transformers

- Transformer plays a crucial role in the latest AI models (for example, GPT).
- Positional Encoding (PE) is an important technique within the Transformer architecture.

Sequential Data

Positional Encoding



What	can	I	say	
-0.757	0.989	-0.537	-0.288	i=0
-0.654	-0.146	0.844	-0.958	i=1
1.000	-0.043	-0.998	0.086	i=2
-0.022	-0.999	0.065	0.996	i=3

PE of the first token

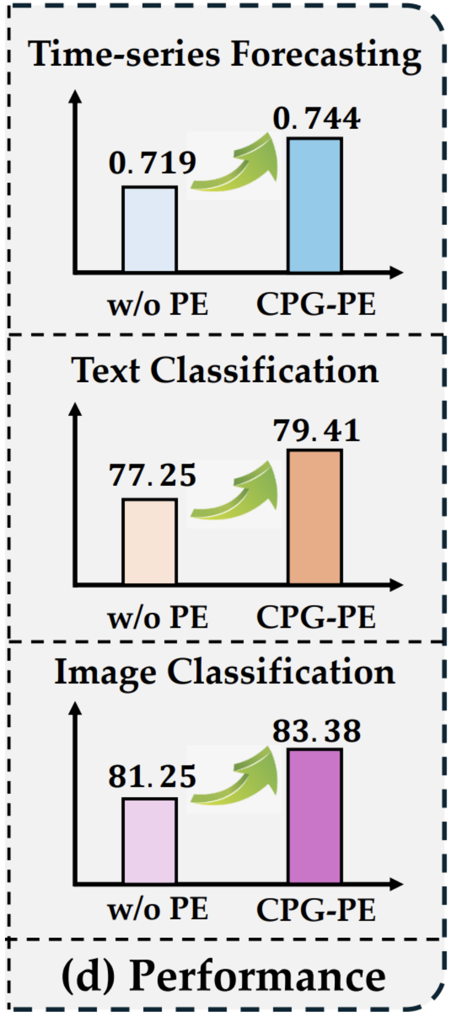
The encoding for each position in this sequence is different, making it easier for the model to process their sequential relationships.

We find that the CPGs can be used as a PE for SNNs!

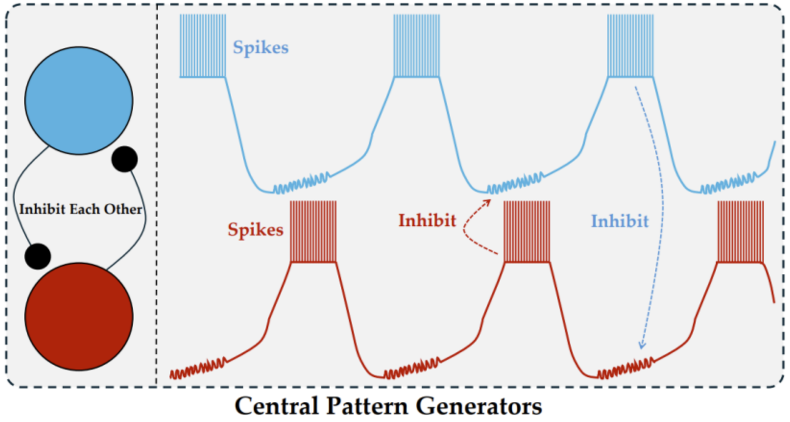
CPG-PE

- Current SNN version of the Transformer model:
  - (1) Non-uniqueness of Each Position in Spike-form;
  - (2) Non-spike Output
- Our proposed CPG-PE:
  - (1) Both brain-inspired and hardware-friendly
  - (2) Uniqueness of Each Position in Spike-form

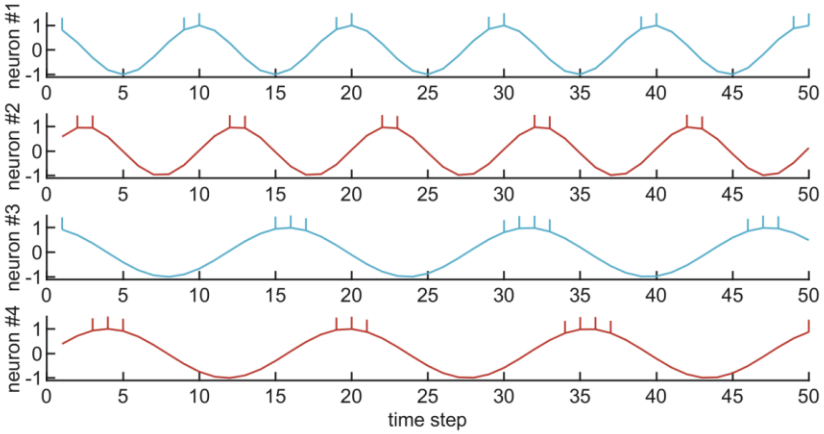
Experiment Results



CPG Illustration



PE in SNNs



- This also inspires us to consider a new role for CPGs in neuroscience—not just as rhythm generators.

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

Relationship between PE and CPGs

The general form of CPGs: $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) + \mathbf{G}(\mathbf{x}, \mathbf{y}), \quad \dot{\mathbf{y}} = \mathbf{H}(\mathbf{y}) + \mathbf{K}(\mathbf{x}, \mathbf{y})$

Consider one of the simplest CPGs with the following assumptions:

1. The CPG is a coupled nonlinear oscillator with 2 neurons whose states are represented as $\mathbf{x}(t)$ and $\mathbf{y}(t)$.
2. Both neurons are autonomic neurons and will gain membrane voltage with constant speed, i.e., $\mathbf{F}(\mathbf{x}) = b > 0, \mathbf{H}(\mathbf{y}) = d > 0$.
3. Neuron represented by $\mathbf{x}$ will inhibits $\mathbf{y}$ while $\mathbf{y}$ excites $\mathbf{x}$. And the influence is proportional to the other neuron's state. Formally, $\mathbf{G}(\mathbf{x}, \mathbf{y}) = a\mathbf{y}, \mathbf{K}(\mathbf{x}, \mathbf{y}) = -c\mathbf{x}$ where $a > 0, c > 0$.

$$\dot{\mathbf{x}}(t) = a\mathbf{y}(t) + b, \quad \dot{\mathbf{y}}(t) = -c\mathbf{x}(t) + d$$

The PE in Transformers is a particular solution of the membrane potential variations in a specific type of CPG.

General Solution:

$$\mathbf{x}(t) = k_1 \cos(\sqrt{ac}\, t) + k_2 \sqrt{\frac{a}{c}} \sin(\sqrt{ac}\, t) + \frac{d}{c}$$

$$\mathbf{y}(t) = -k_1 \sqrt{\frac{c}{a}} \sin(\sqrt{ac}\, t) + k_2 \cos(\sqrt{ac}\, t) - \frac{b}{a}$$

re-parameterize

$$\mathbf{x}(t') = \sqrt{k_1^2 + \frac{a}{c} k_2^2} \sin(\sqrt{ac}\, t') + \frac{d}{c} = A_1 \sin(w_1 t') + b_1,$$

$$\mathbf{y}(t') = \sqrt{\frac{c}{a} k_1^2 + k_2^2} \cos(\sqrt{ac}\, t') - \frac{b}{a} = A_2 \cos(w_2 t') + b_2$$

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

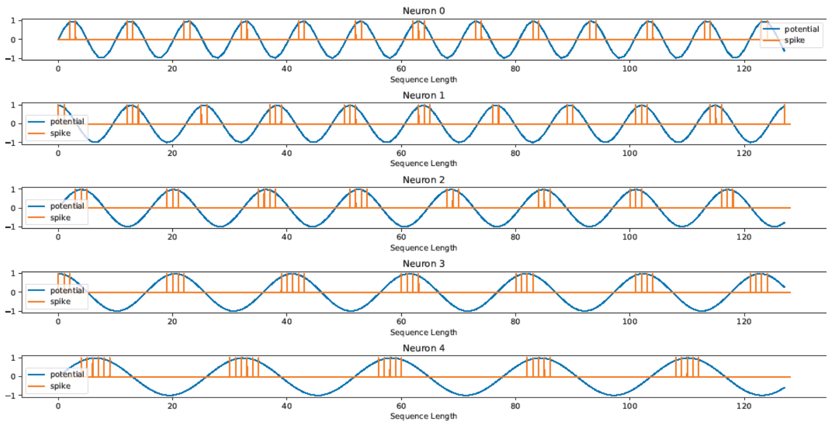
Implementations

Consider a system with N pairs of CPG neurons, resulting in a total of $2N$ cells. Then for $i = 1, 2, \ldots, N$, the equations governing the CPG-PE are as follows:

$$\text{CPG-PE}^{2i-1}(t) = H\left(\cos\left(\eta \frac{t}{\tau^{\frac{i}{N}}}\right) - v^{\text{thres}}\right), \tag{11}$$

$$\text{CPG-PE}^{2i}(t) = H\left(\sin\left(\eta \frac{t}{\tau^{\frac{i}{N}}}\right) - v^{\text{thres}}\right), \tag{12}$$

where η is a constant to control the period, τ represents the base period, and v^{thres} denotes the membrane potential threshold.



Simply concatenate with SNN neurons

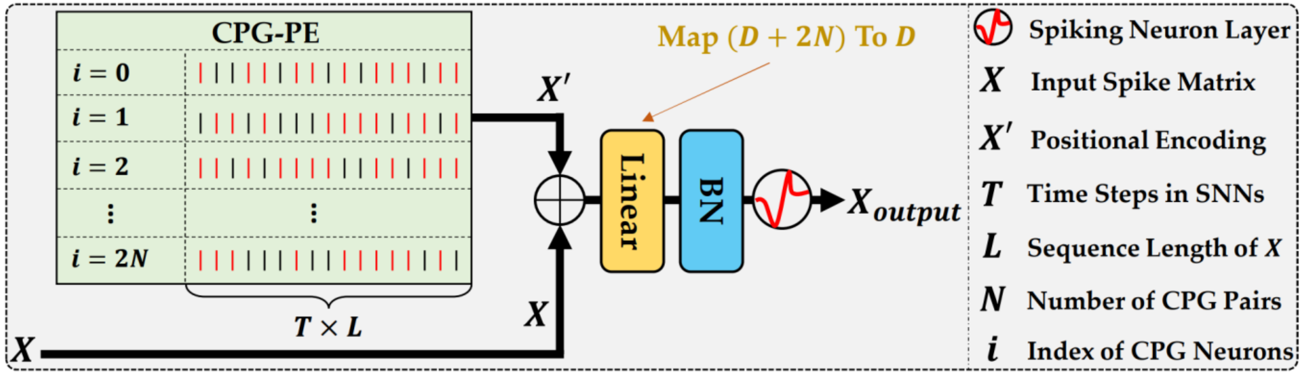


Illustration of applying CPG-PE to SNNs

We also prove that CPG-PE is hardware-friendly because:

- CPG-PE can be integrated into a linear layer;
- CPG-PE can be simply implemented with 2 LIF neurons.

Refer to Appendix B and C

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

Forecasting Tasks

Experimental results of time-series forecasting on
4 benchmarks with different prediction lengths.

Model	SNN	Spike PE	Metric	Metr-la				Pems-bay				Solar				Electricity				Avg.
				6	24	48	96	6	24	48	96	6	24	48	96	6	24	48	96	
TCN (ANN)	✗	–	R^2↑	.820	.601	.455	.330	.881	.749	.695	.689	.958	.871	.737	.661	.975	.973	.968	.962	.770
			RSE↓	.446	.665	.778	.851	.373	.541	.583	.587	.210	.359	.513	.583	.282	.287	.319	.345	.483
SpikeTCN w/o PE [32]	✓	–	R^2↑	.783	**.603**	**.468**	.326	.811	.729	.662	.633	.937	.840	.708	.650	.970	**.963**	.958	.953	.750
			RSE↓	.491	.665	**.769**	.865	.469	**.541**	.625	.635	.259	.401	.541	**.596**	.333	**.342**	**.368**	.389	.518
SpikeTCN w/ CPG-PE	✓	✓	R^2↑	**.802**	**.603**	.467	**.337**	**.839**	**.737**	**.684**	**.656**	**.951**	**.861**	**.729**	**.651**	**.974**	.960	**.959**	**.956**	**.760**
			RSE↓	**.469**	**.664**	.770	**.859**	**.433**	.555	**.604**	**.632**	**.222**	**.373**	**.521**	.606	**.278**	.380	.374	**.370**	**.506**
RNN (ANN)	✗	✗	R^2↑	.844	.600	.442	.307	.870	.775	.690	.683	.959	.830	.810	.718	.978	.972	.971	.964	.776
			RSE↓	.414	.668	.781	.897	.390	.511	.578	.609	.208	.413	.438	.549	.273	.295	.299	.316	.477
SpikeRNN w/o-PE [32]	✓	–	R^2↑	**.846**	**.622**	.433	.283	.872	.745	.685	.654	.923	.820	**.812**	.714	**.977**	**.972**	.962	**.960**	.768
			RSE↓	**.412**	.648	.794	.935	.387	.528	.588	.634	.278	.425	**.435**	.586	.267	.296	.346	.481	.503
SpikeRNN w/ CPG-PE	✓	✓	R^2↑	.844	.621	**.438**	**.306**	**.874**	**.763**	**.688**	**.667**	**.934**	**.833**	.811	**.724**	**.977**	**.972**	**.966**	.958	**.773**
			RSE↓	.416	**.645**	**.782**	**.878**	**.380**	**.523**	**.579**	**.621**	**.264**	**.419**	**.435**	**.544**	**.265**	**.294**	**.315**	**.366**	**.482**
Transformer (ANN)	✗	✗	R^2↑	.727	.554	.413	.284	.785	.734	.688	.673	.953	.858	.759	.718	.978	.975	.972	.964	.752
			RSE↓	.551	.704	.808	.895	.502	.558	.610	.618	.223	.377	.504	.545	.260	.277	.347	.425	.512
Spikformer w/o PE	✓	–	R^2↑	.697	.491	.383	.242	.768	.684	.678	.663	.903	.819	.715	.656	.956	.955	.953	.943	.719
			RSE↓	.581	.753	.828	.917	.521	.607	.613	.627	.319	.439	.548	.602	.371	.375	.386	.450	.559
Spikformer w/ RPE [4]	✓	✓	R^2↑	.713	.527	.399	.267	.773	.697	.686	.667	.929	.828	.744	.674	.959	.955	.955	.954	.733
			RSE↓	.565	.725	.818	.903	.514	.594	.606	.621	.272	.426	.519	.586	.373	.371	.379	.382	.541
Spikformer w/ Float-PE	✓	✗	R^2↑	.699	.502	.409	.255	.762	.704	.687	.666	.934	.834	.752	.699	.970	.967	.960	.957	.734
			RSE↓	.578	.744	.810	.912	.527	.588	.605	.623	.264	.418	.512	.563	.307	.322	.356	.362	.531
Spikformer w/ CPG-PE	✓	✓	R^2↑	**.726**	.526	**.419**	**.287**	**.780**	.712	**.690**	.666	**.937**	.833	**.757**	.707	**.972**	.970	.966	.960	**.744**
			RSE↓	**.553**	.720	**.806**	**.890**	.508	.580	**.602**	.622	**.257**	.420	**.506**	.555	**.299**	.310	.314	**.355**	**.519**
Spikformer w/ CPG-Full	✓	✓	R^2↑	.719	**.530**	.417	.286	.779	**.714**	.689	**.668**	.936	**.835**	**.757**	**.709**	.971	**.971**	**.968**	**.962**	**.744**
			RSE↓	.560	**.719**	.807	.893	**.507**	**.577**	.605	**.620**	.260	**.417**	.508	**.548**	.304	**.308**	**.311**	.439	.523

Performance 0.719 -> 0.744 by just adding the CPG-PE (marginal computation cost increase)

Classification Tasks

Text Classification

Model	SNN	Spike PE	Param (M)	English Dataset				Chinese Dataset		Avg.
				MR	SST-2	Subj	SST-5	ChnSenti	Waimai	
Fine-tuned BERT [39]	✗	✗	109.8	**87.63**±0.18	**92.31**±0.17	**95.90**±0.16	**50.41**±0.13	**89.48**±0.16	**90.27**±0.13	**84.33**
Spikformer w/o PE [29]	✓	–	109.8	75.87±0.35	81.71±0.31	91.60±0.30	41.84±0.39	85.62±0.25	86.87±0.28	77.25
Spikformer w/ Random-PE	✓	✓	110.4	75.90±0.42	81.64±0.31	91.40±0.35	41.86±0.41	85.63±0.29	86.90±0.30	77.23
Spikformer w/ Float-PE	✓	✗	109.8	79.67±0.36	82.18±0.34	92.20±0.31	42.58±0.41	85.71±0.26	88.34±0.32	78.44
Spikformer w/ CPG-PE [Ours]	✓	✓	110.4	**82.42**±0.42	**82.90**±0.33	**92.50**±0.25	**43.62**±0.36	**86.54**±0.26	**88.49**±0.29	**79.41**

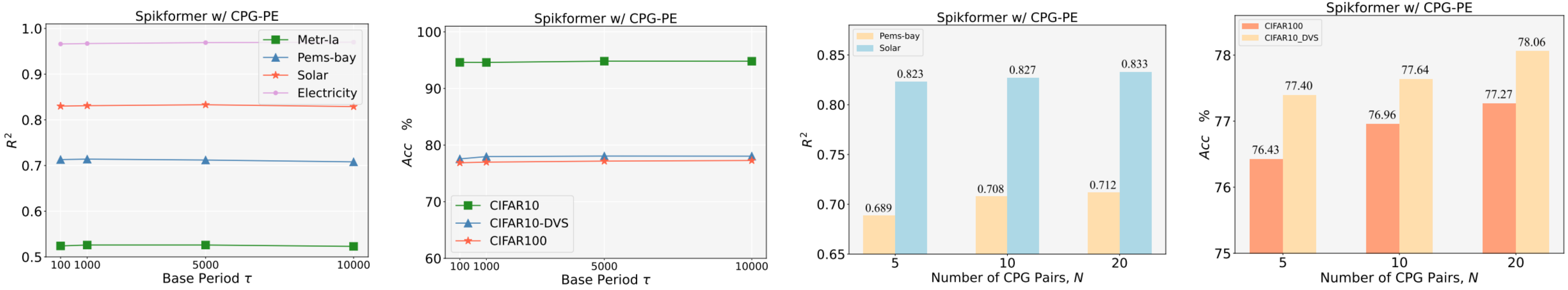
Image Classification

Model	SNN	Spike PE	CIFAR10		CIFAR10-DVS		CIFAR100		Avg.
			Param (M)	Accuracy	Param (M)	Accuracy	Param (M)	Accuracy	
Vision-Transformer [23]	✗	✗	9.32	**96.73**	–	–	9.36	**81.02**	–
Spikformer w/o PE	✓	–	8.00	93.77	1.99	76.40	8.04	73.59	81.25
Spikformer w/ Random-PE	✓	✓	8.17	93.85	2.06	76.44	8.20	73.54	81.27
Spikformer w/ Float-PE	✓	✗	8.00	94.42	1.99	77.60	8.04	74.73	82.25
Spikformer w/ RPE [4]	✓	✓	9.33	94.64*	2.57	77.95*	9.37	76.78*	83.12
Spikformer w/ CPG-PE [Ours]	✓	✓	8.17	**94.82**	2.06	**78.06**	8.20	**77.27**	**83.38**

Contents

- **Background**
  - Spiking Neural Networks
  - Central Pattern Generators
- **Methods**
  - Relationship between PE and CPGs
  - Implementations
- **Experiments**
  - Main Results
  - Analysis

Sweeping CPG properties



Positional Encoding Analysis

An ideal PE method for SNNs:
(1) Uniqueness of each position;
(2) Ability to discern positional information;
(3) Compatibility with neuromorphic hardware;
(4) Formulation in spike-form.

	(1)	(2)	(3)	(4)
CPG-PE	√	√	√	√
PE in Spikformer	×	×	√	×

Repetition rate 12.19%

THANKS!

Contact us:
czlv24@m.fudan.edu.cn
dongqihan@ microsoft.com
yansenwang@microsoft.com