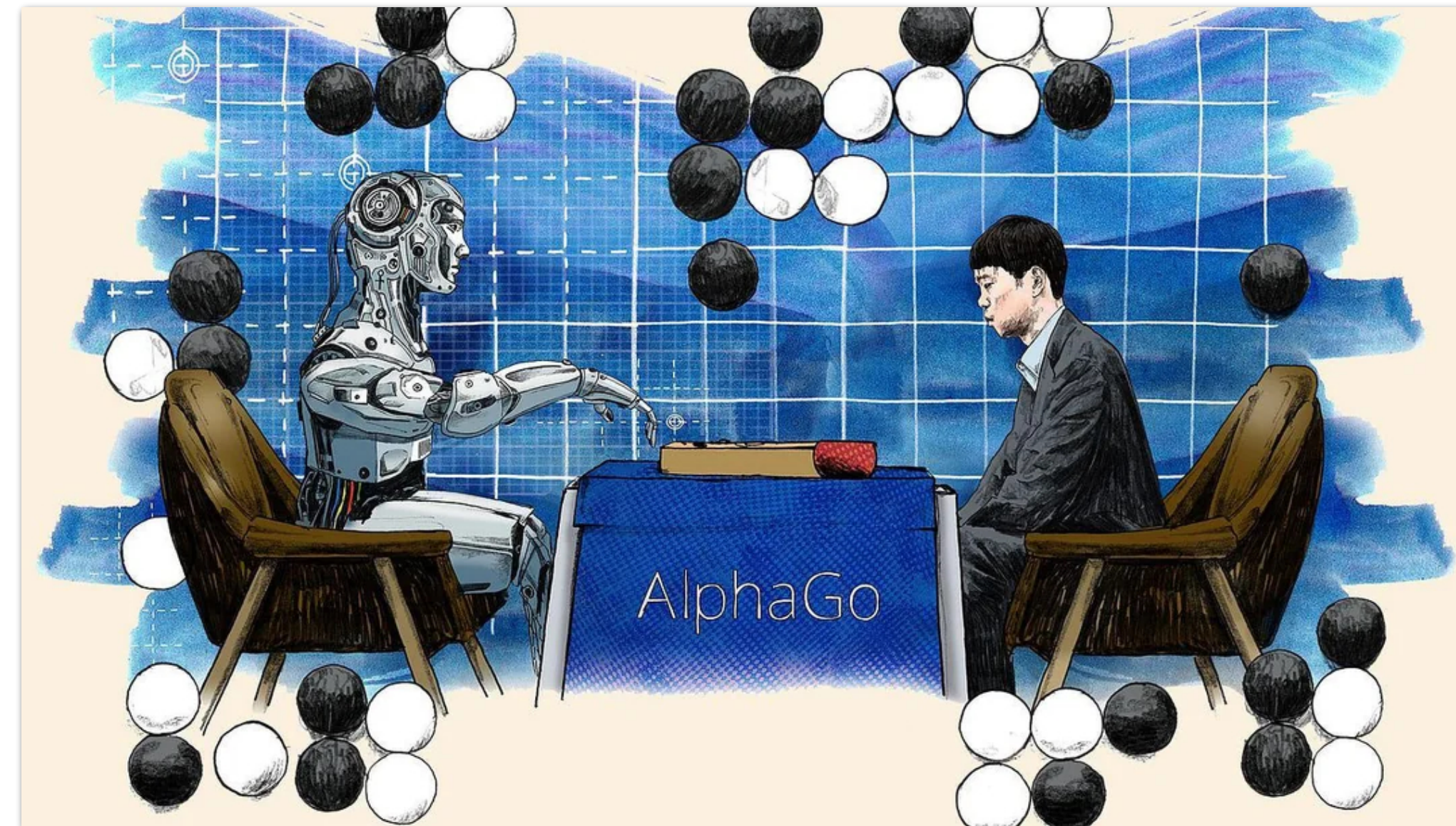


A mixed-feedback recurrent neural network to create spikes

F. De Geeter, D. Ernst & G. Drion

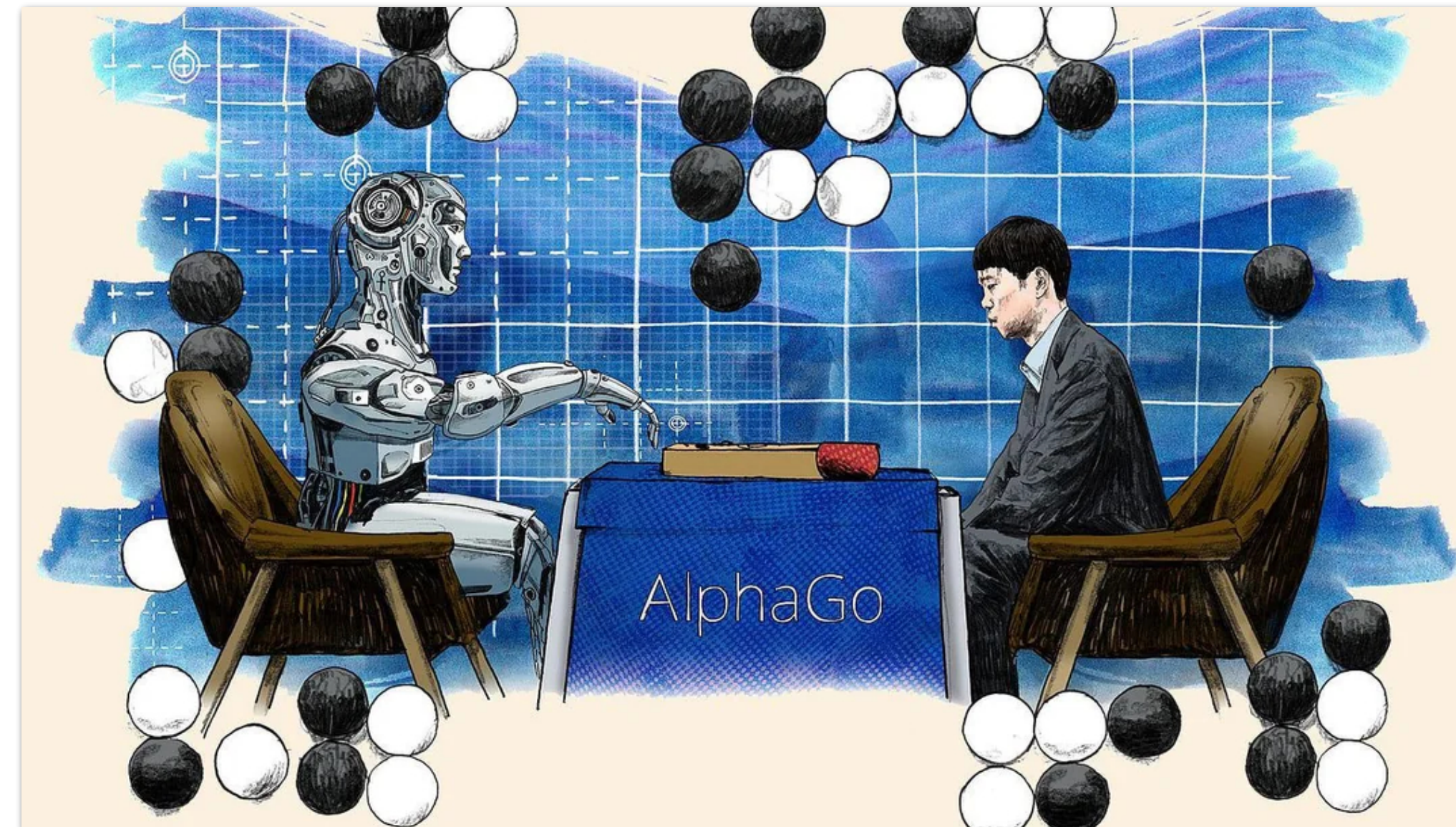


Current AI's are performant



Current AI's are performant But not energy-efficient

AlphaGo:
 ± 1 MegaWatts

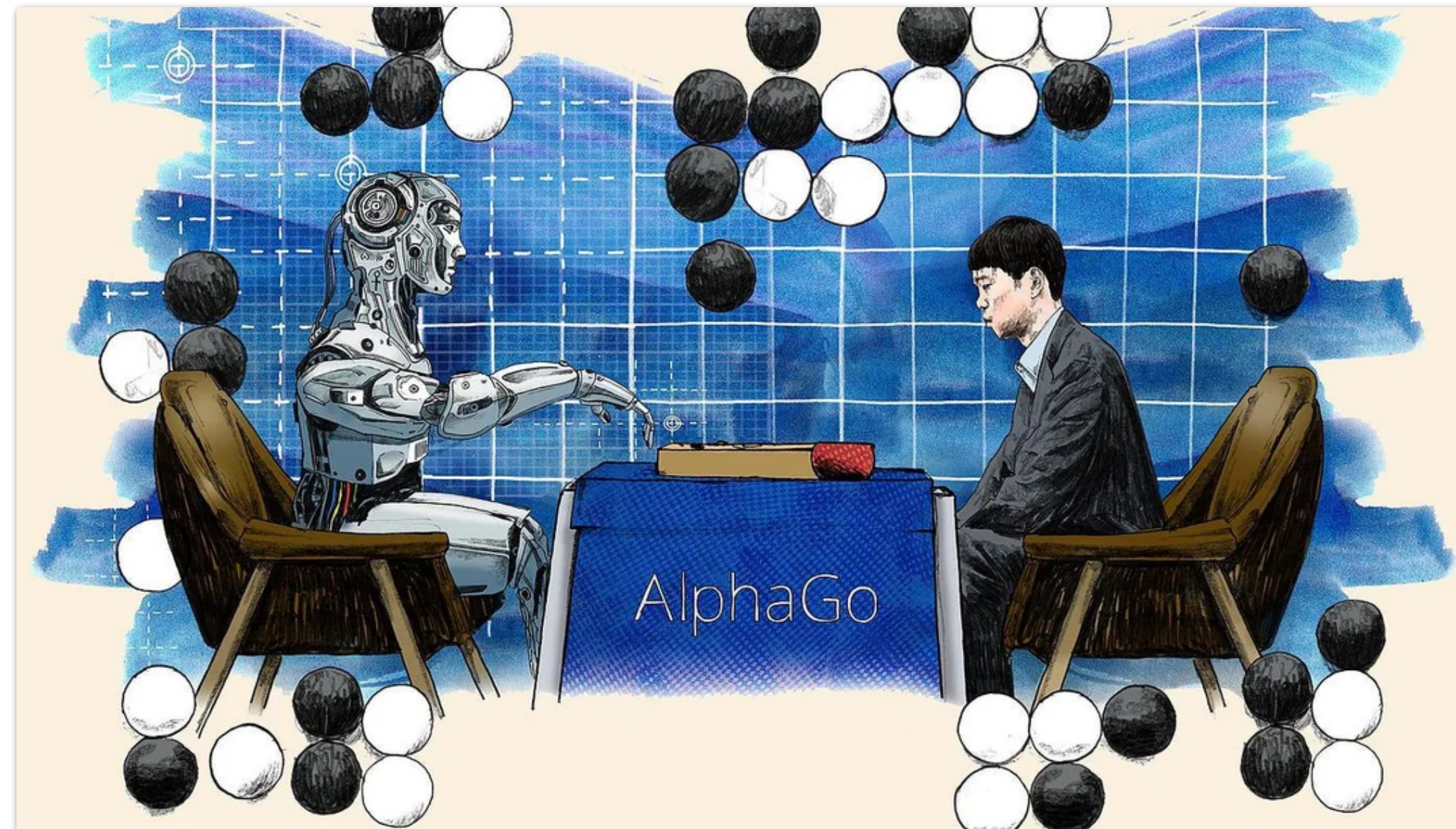


L. Sedol's brain:
 ± 20 Watts

Current AI's are performant But not energy-efficient

AlphaGo:

± 1 MegaWatts

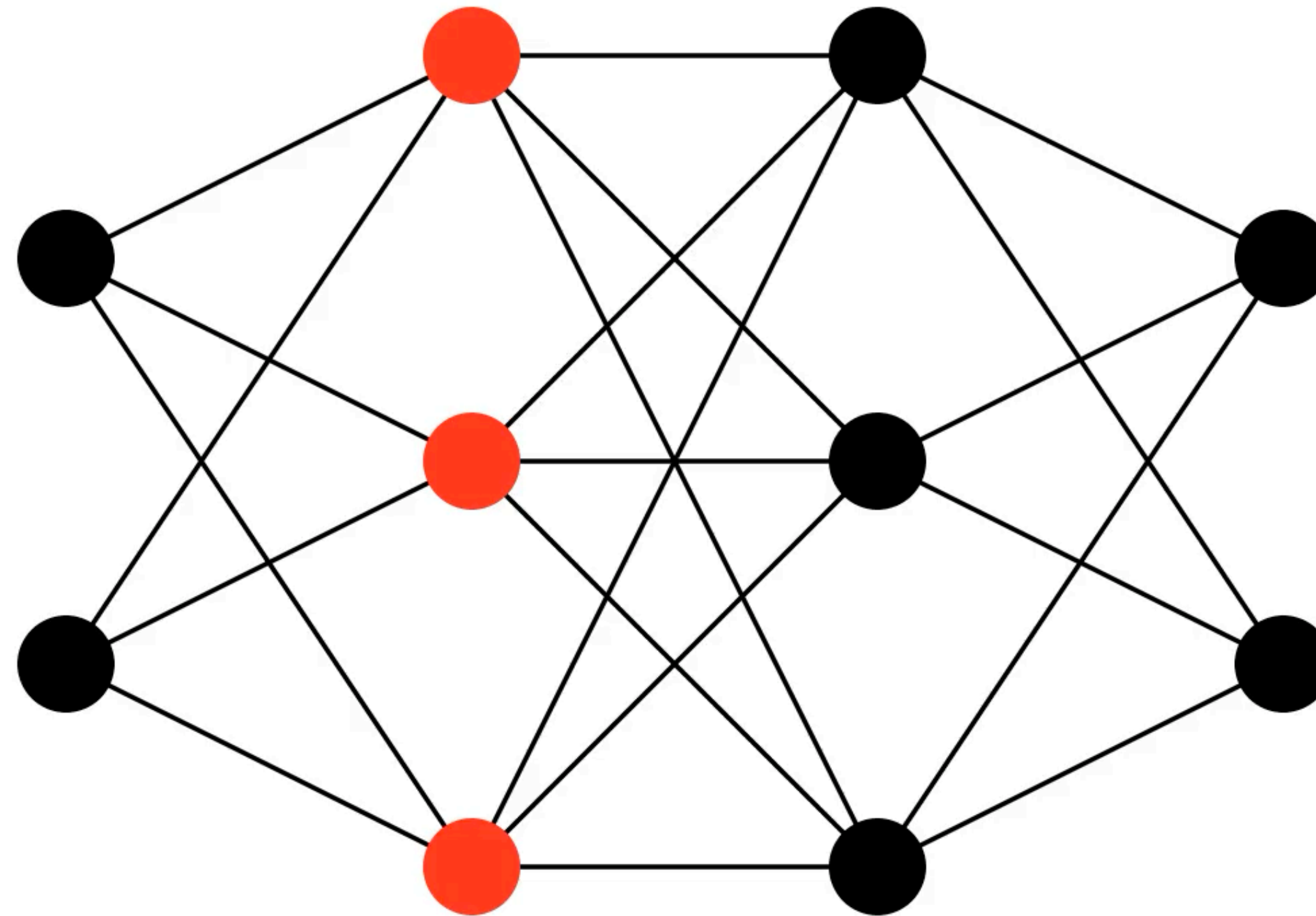


L. Sedol's brain:

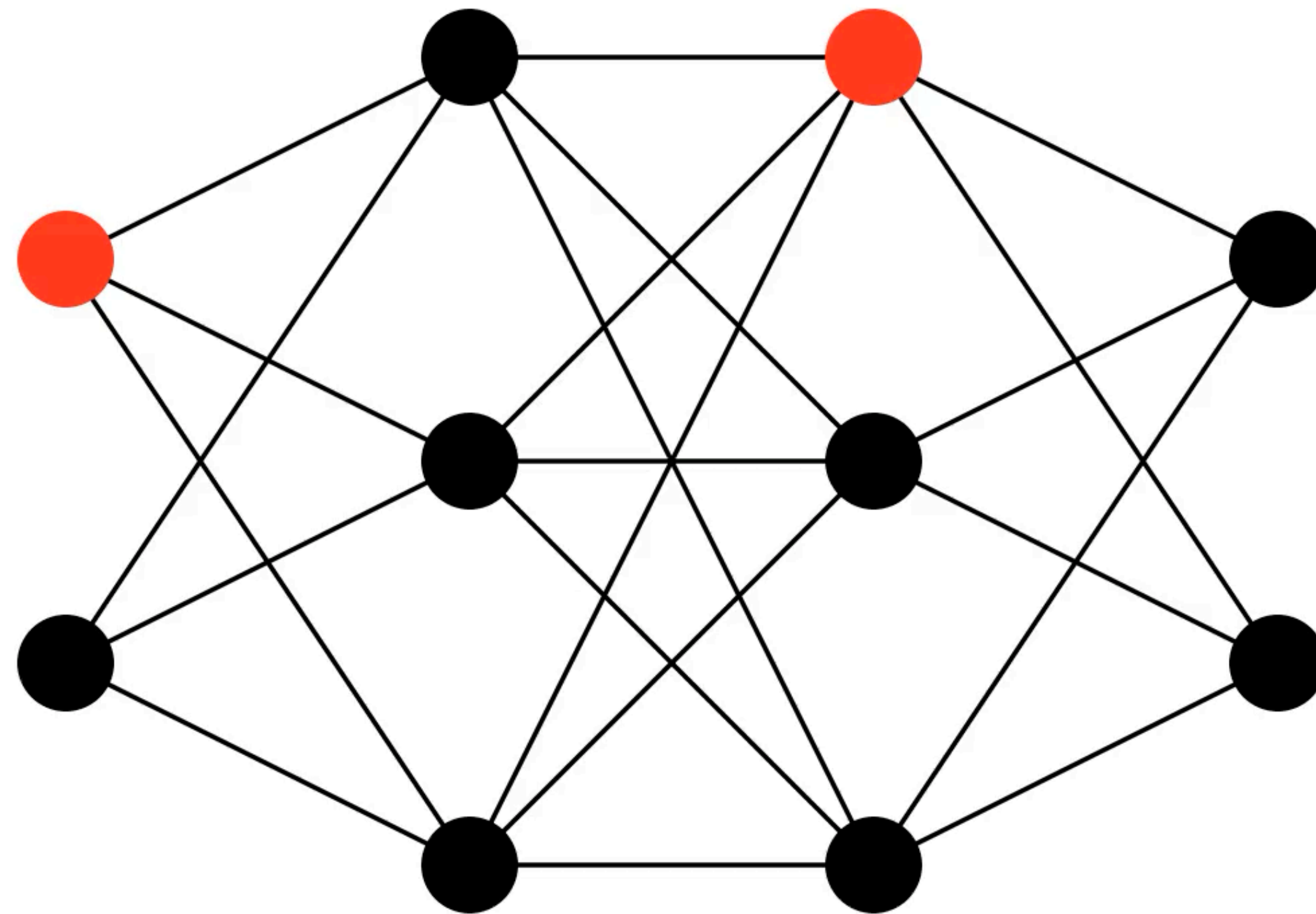
± 20 Watts

Need more energy-efficiency in some applications (e.g. embedded systems)

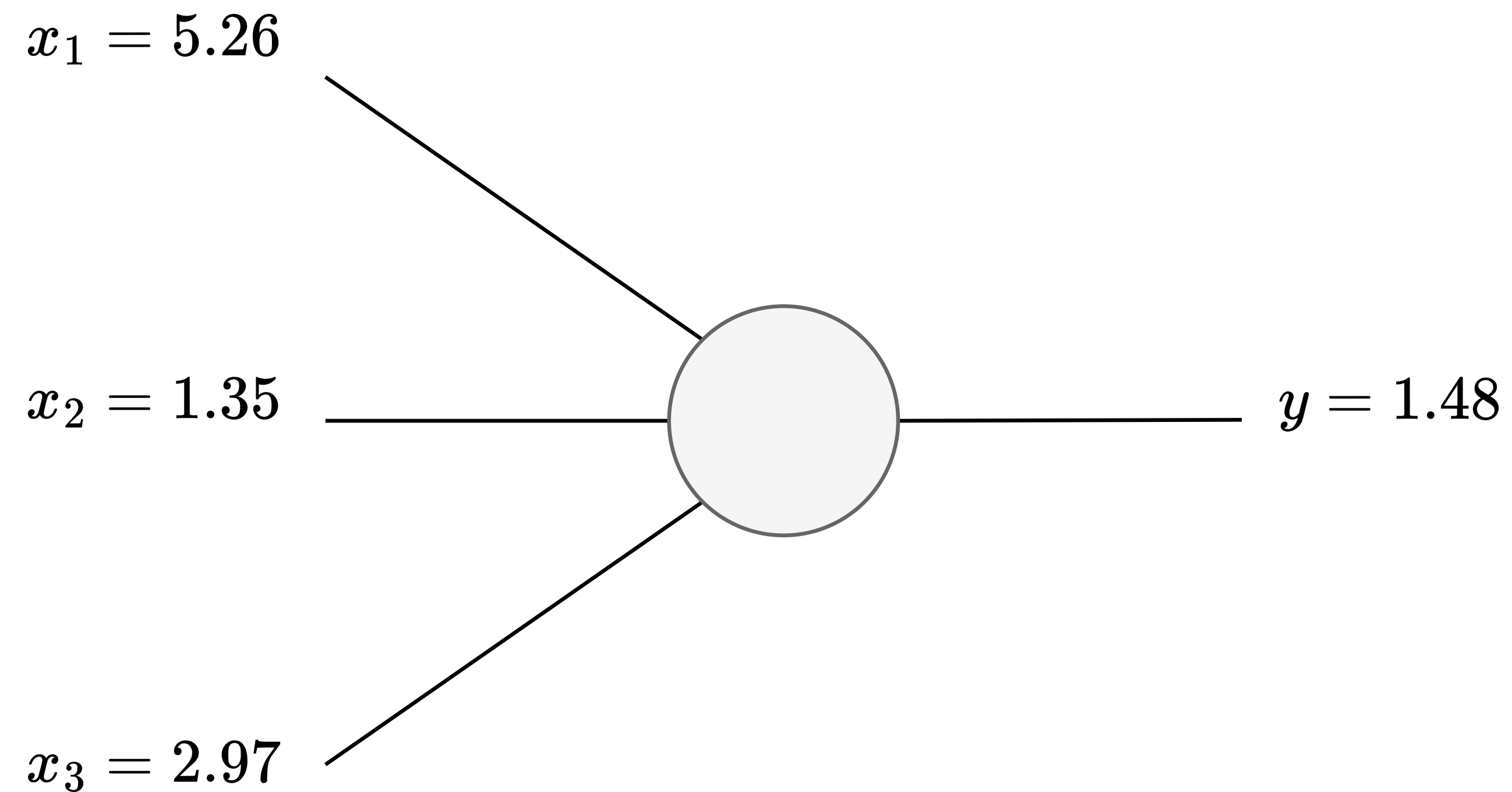
Artificial neurons are clock-driven,



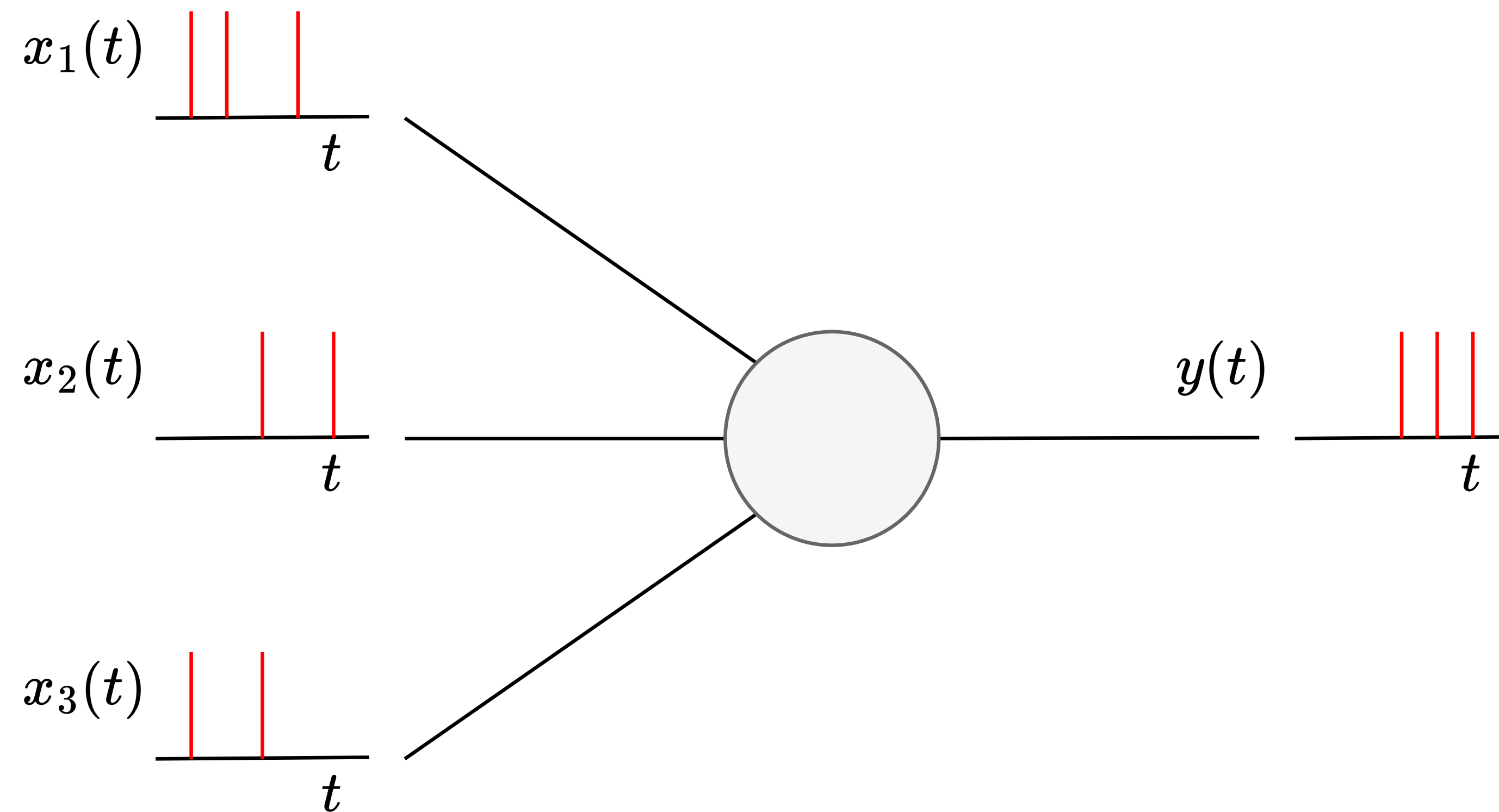
Artificial neurons are clock-driven,
Biological neurons are event-driven



Artificial neurons output numbers,



Artificial neurons output numbers, Biological neurons output events



Good results have been obtained
with spiking neural networks

Good results have been obtained with spiking neural networks

- Training
 - Hard to train
 - But more and more progress

Good results have been obtained with spiking neural networks

- Training
 - Hard to train
 - But more and more progress
- Performance
 - Comparable to ANNs on simple benchmarks
 - Less performant for more complex ones

Good results have been obtained with spiking neural networks

- Training
 - Hard to train
 - But more and more progress
- Performance
 - Comparable to ANNs on simple benchmarks
 - Less performant for more complex ones
- Energy-efficiency
 - Less than 1 Watt for Loihi 2
 - For ± 1 million neurons and ± 120 million synapses

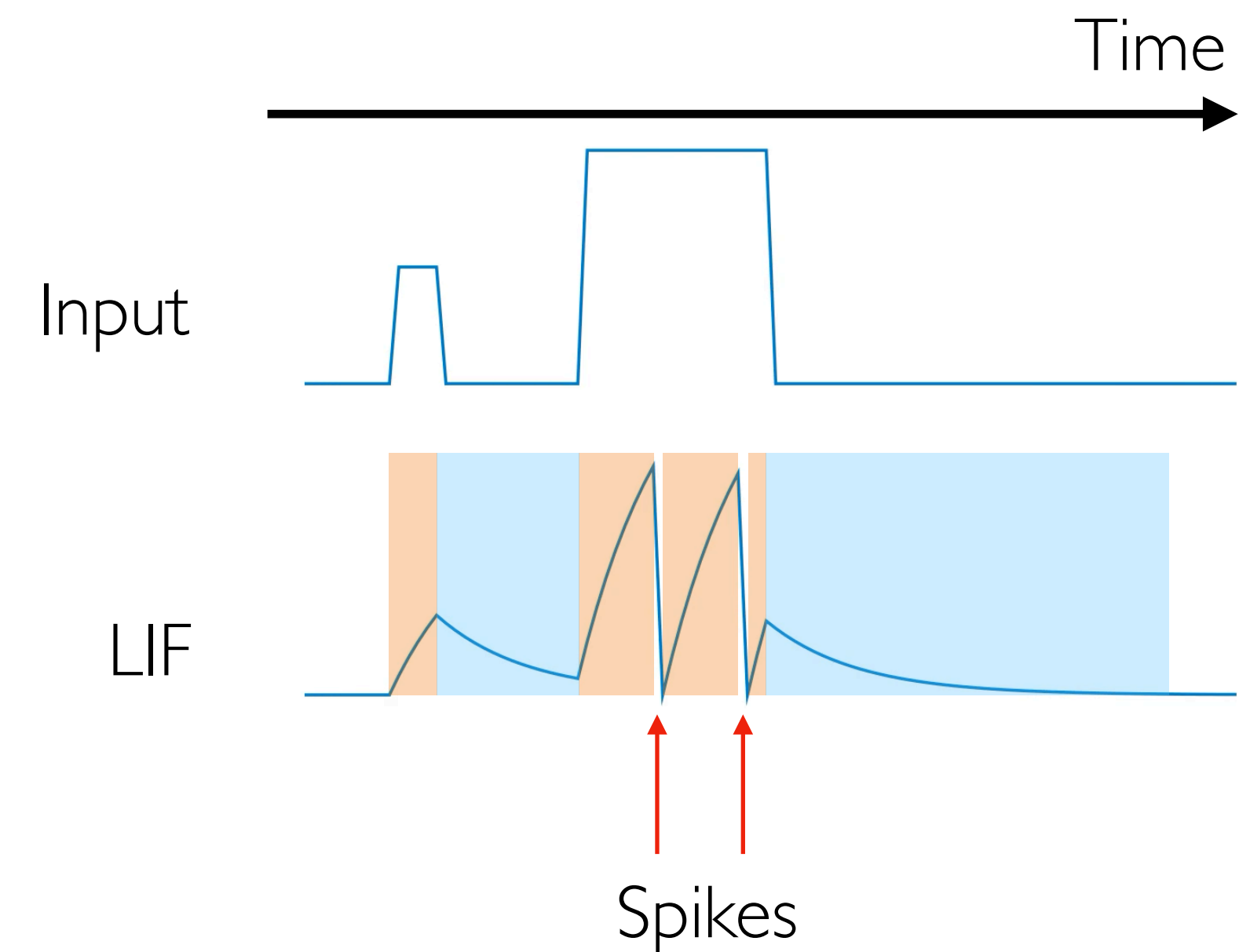


LIF neurons are used in SNNs

- Leaky Integrate-and-Fire
 - Simple leaky integrator with a reset
 - Spikes manually generated
 - Sub-threshold dynamic is purely linear

LIF neurons are used in SNNs

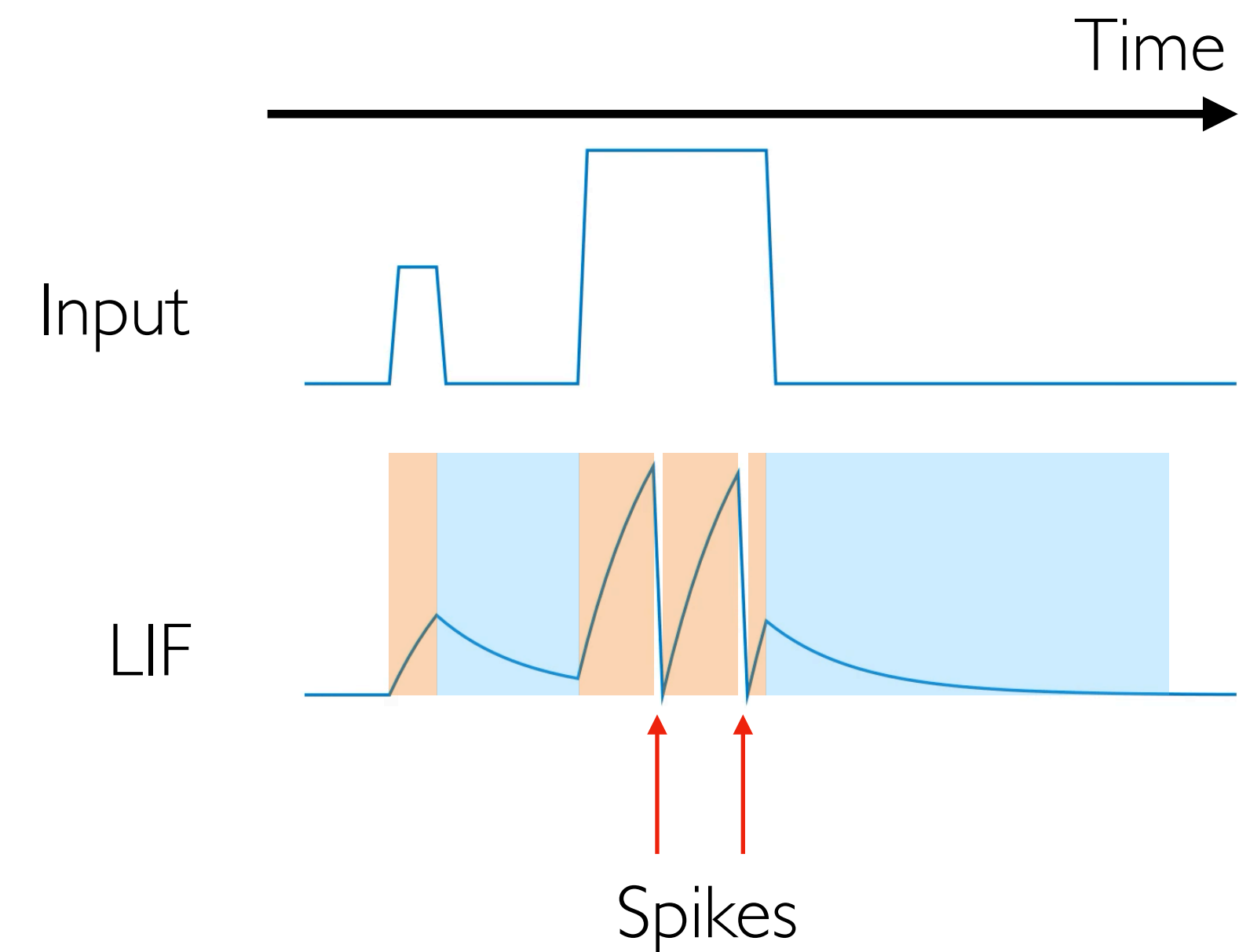
- **Leaky Integrate-and-Fire**
 - Simple leaky integrator with a reset
 - Spikes manually generated
 - Sub-threshold dynamic is purely linear



LIF neurons are used in SNNs

But they do not capture spikes dynamic

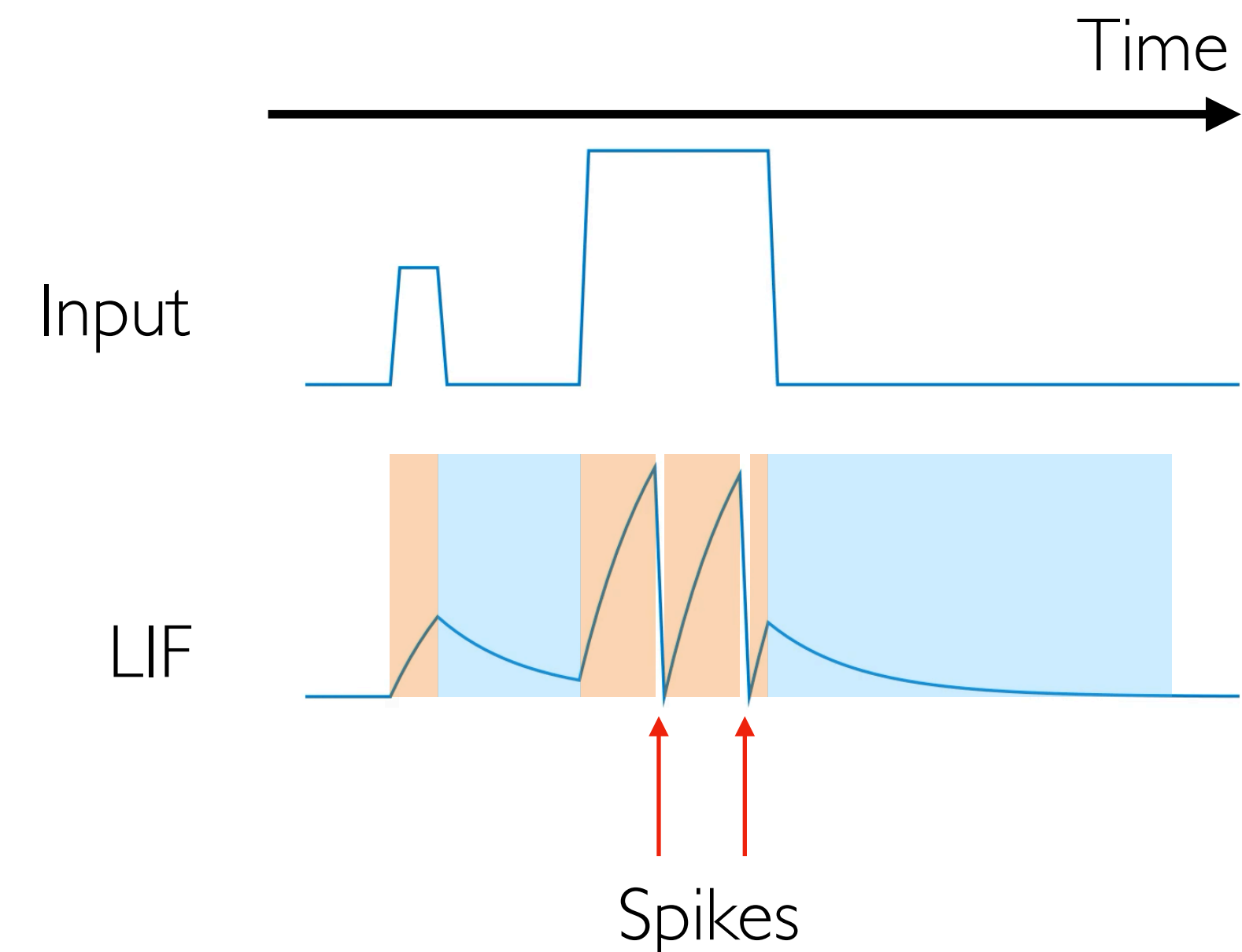
- **Leaky Integrate-and-Fire**
 - Simple leaky integrator with a reset
 - Spikes manually generated
 - Sub-threshold dynamic is purely linear



LIF neurons are used in SNNs

But they do not capture spikes dynamic

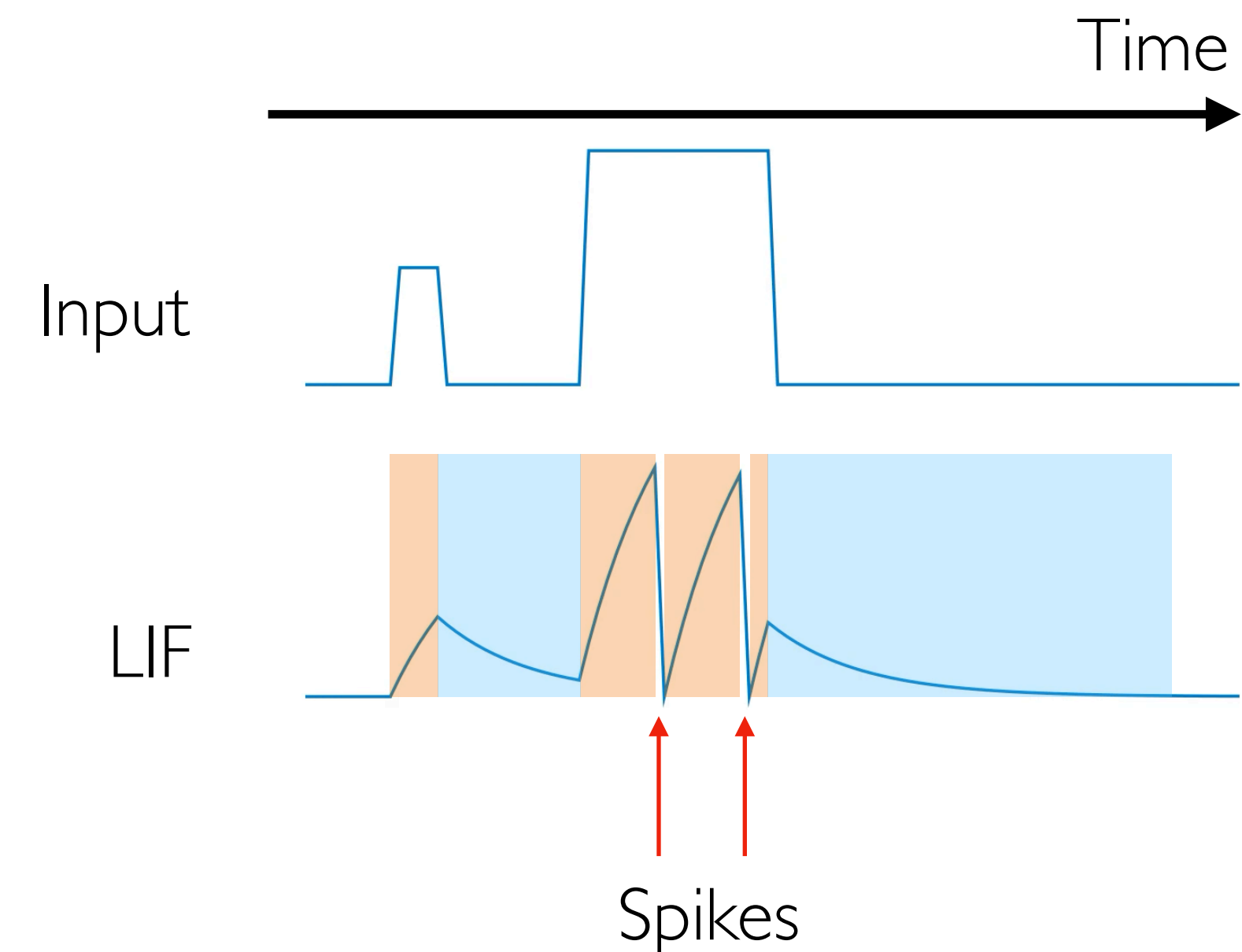
- **Leaky Integrate-and-Fire**
 - Simple leaky integrator with a reset
 - Spikes manually generated
 - Sub-threshold dynamic is purely linear
- Better models to simulate spikes
 - Too complex for machine learning



LIF neurons are used in SNNs

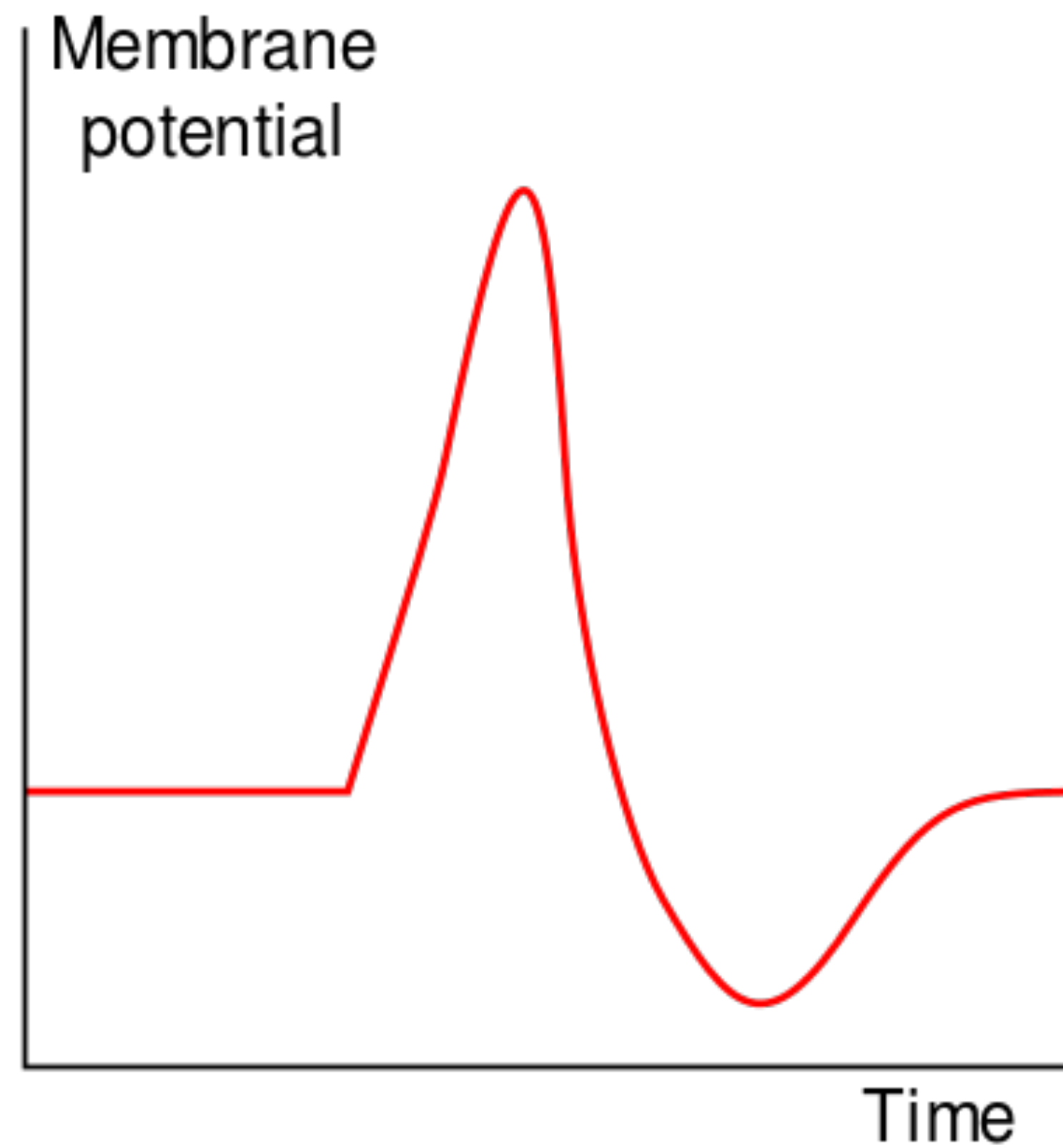
But they do not capture spikes dynamic

- **Leaky Integrate-and-Fire**
 - Simple leaky integrator with a reset
 - Spikes manually generated
 - Sub-threshold dynamic is purely linear
- Better models to simulate spikes
 - Too complex for machine learning

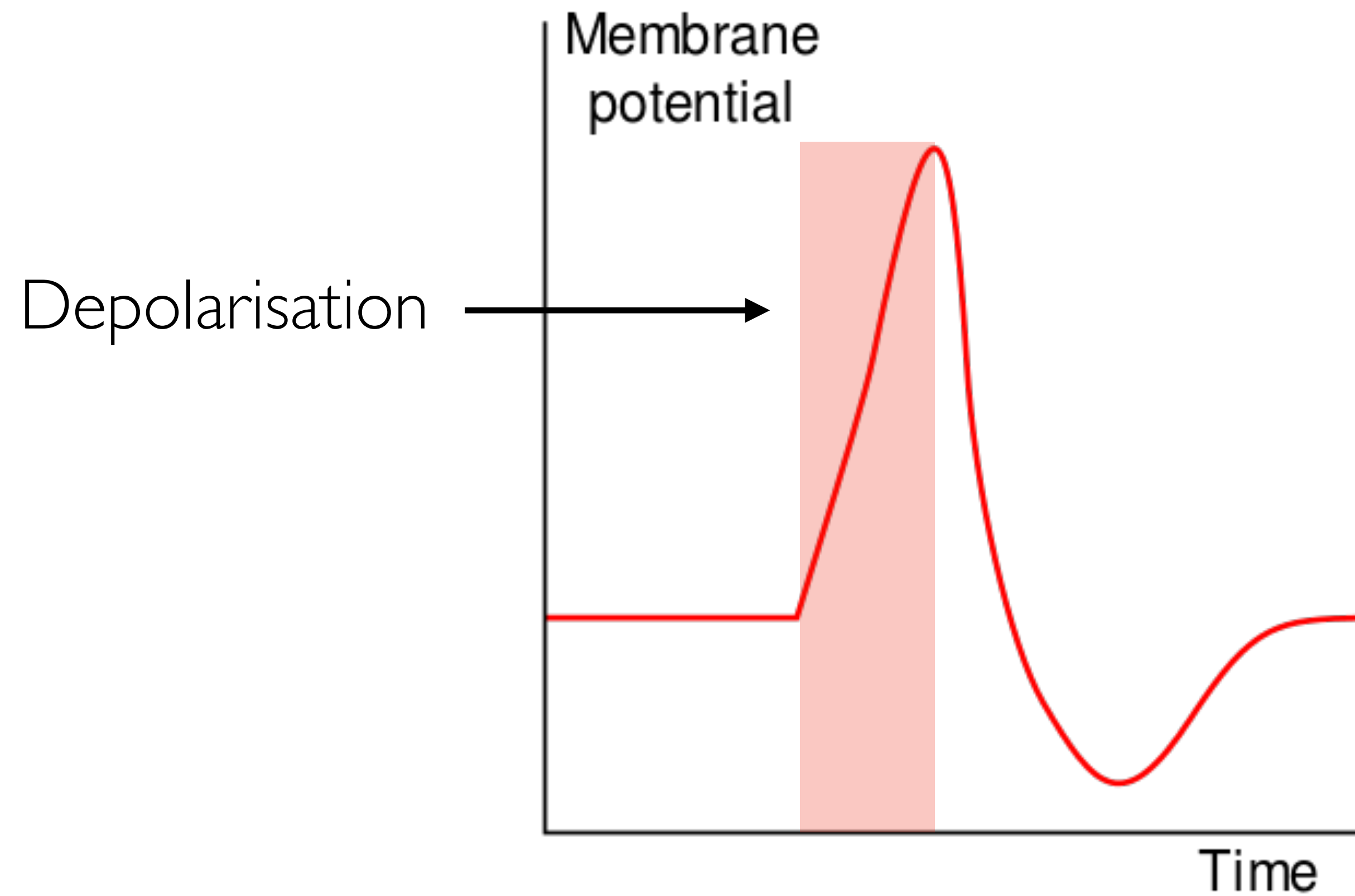


Can we find a trade-off ?

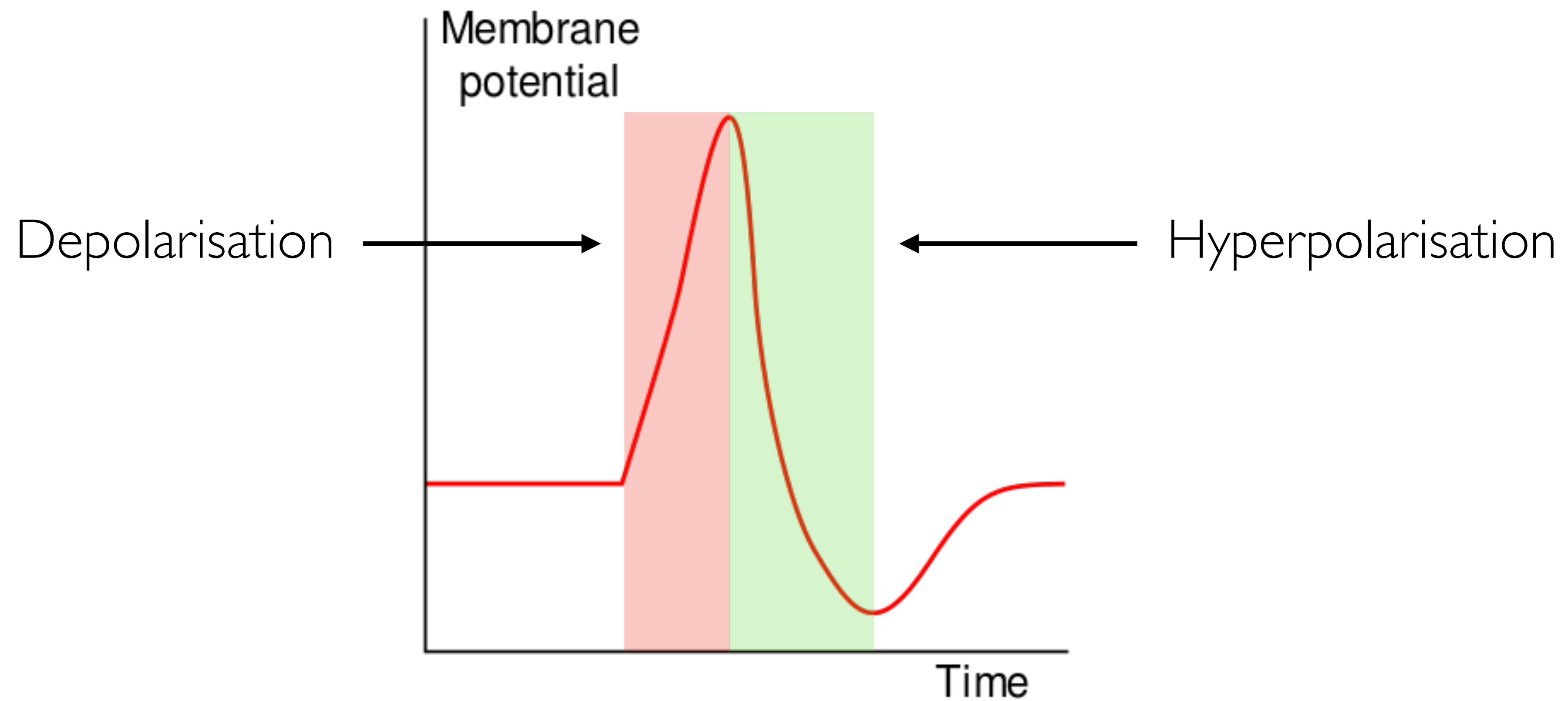
Spikes can be generated by a mixed-feedback system



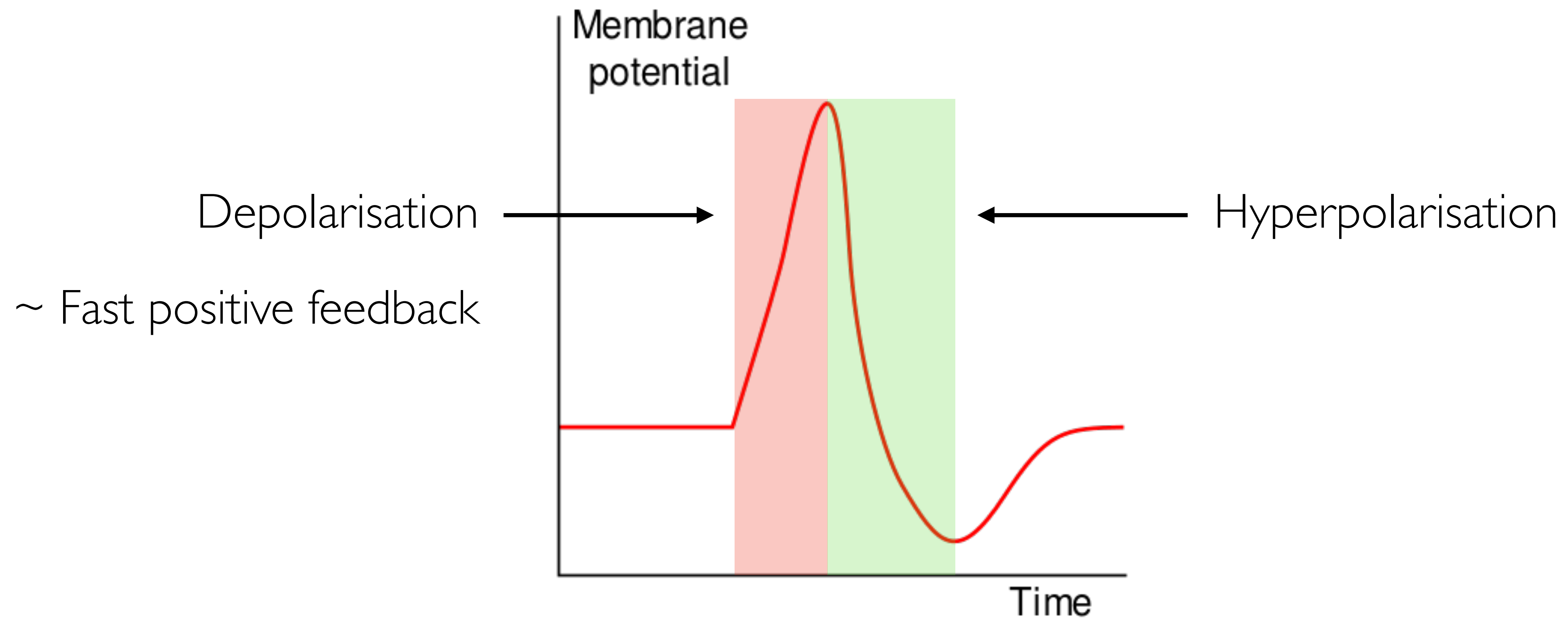
Spikes can be generated by a mixed-feedback system



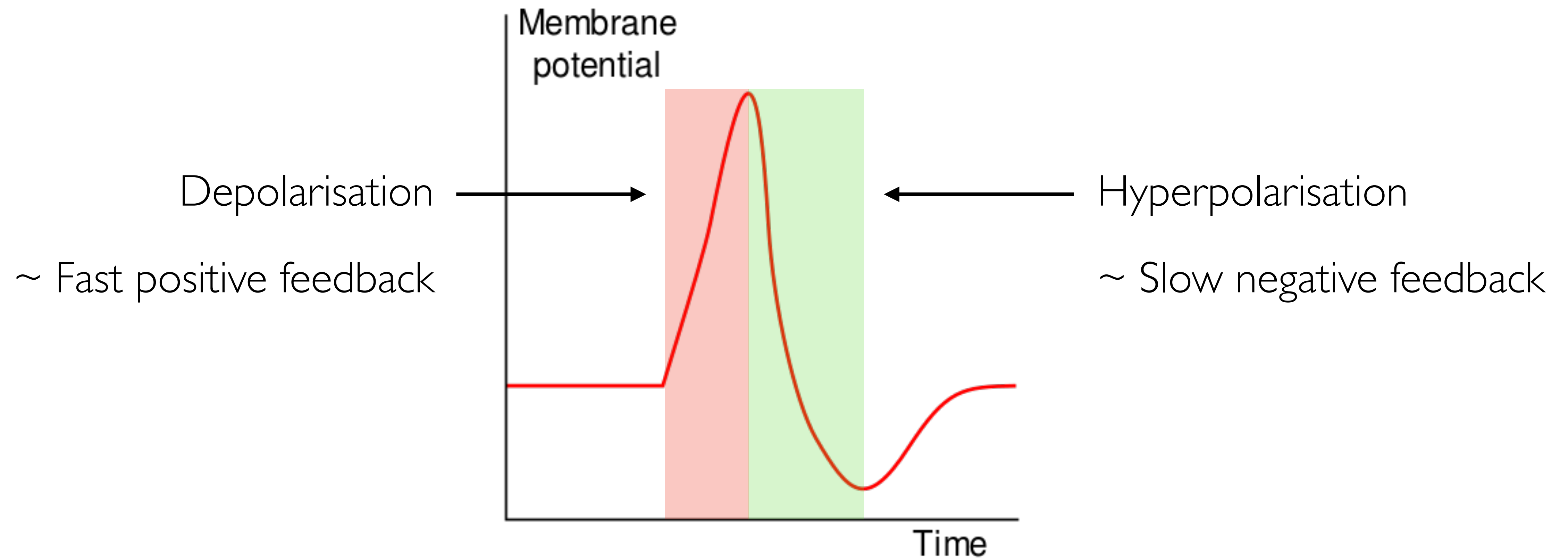
Spikes can be generated by a mixed-feedback system



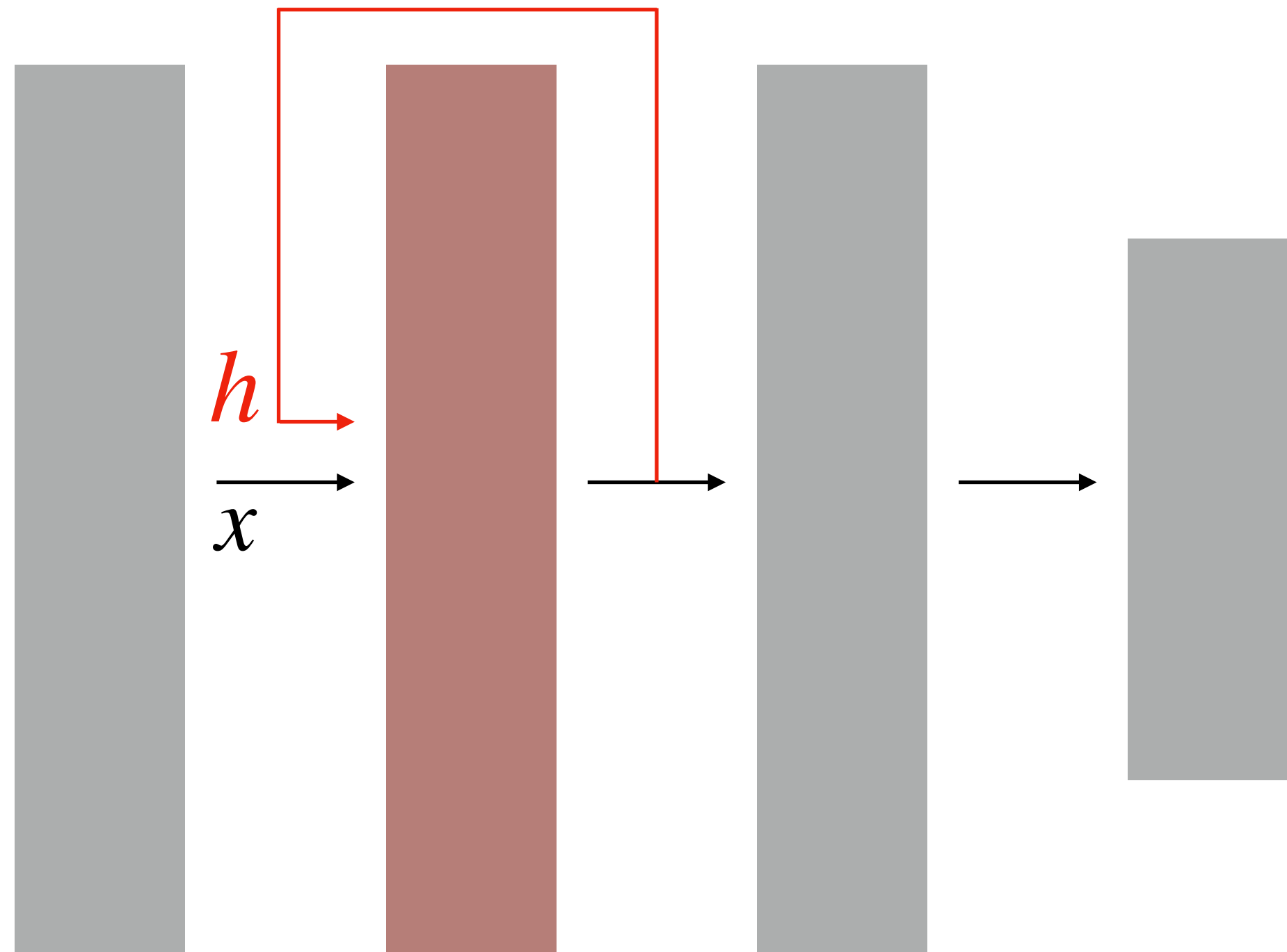
Spikes can be generated by a mixed-feedback system



Spikes can be generated by a mixed-feedback system

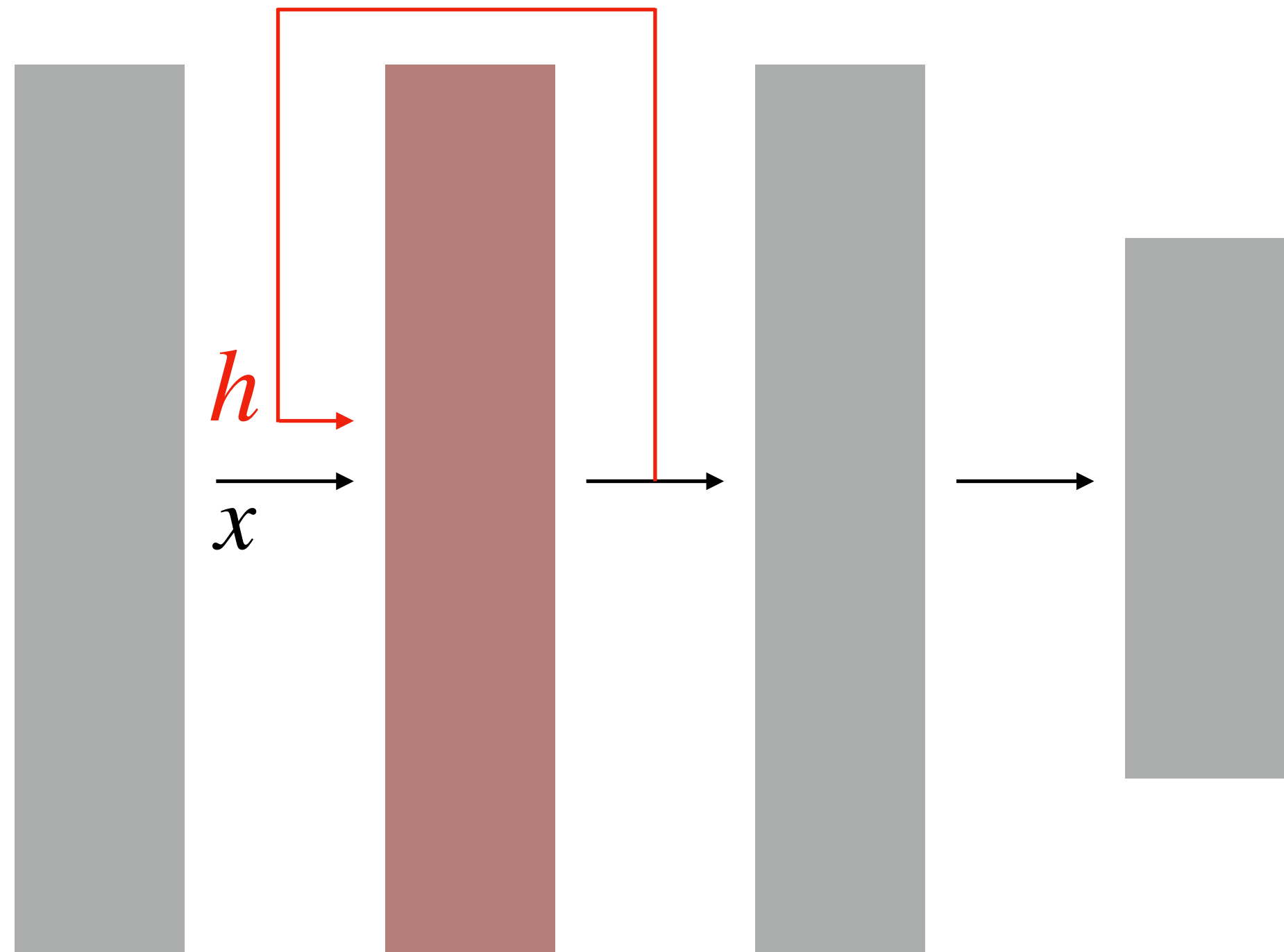


Recurrent neural networks

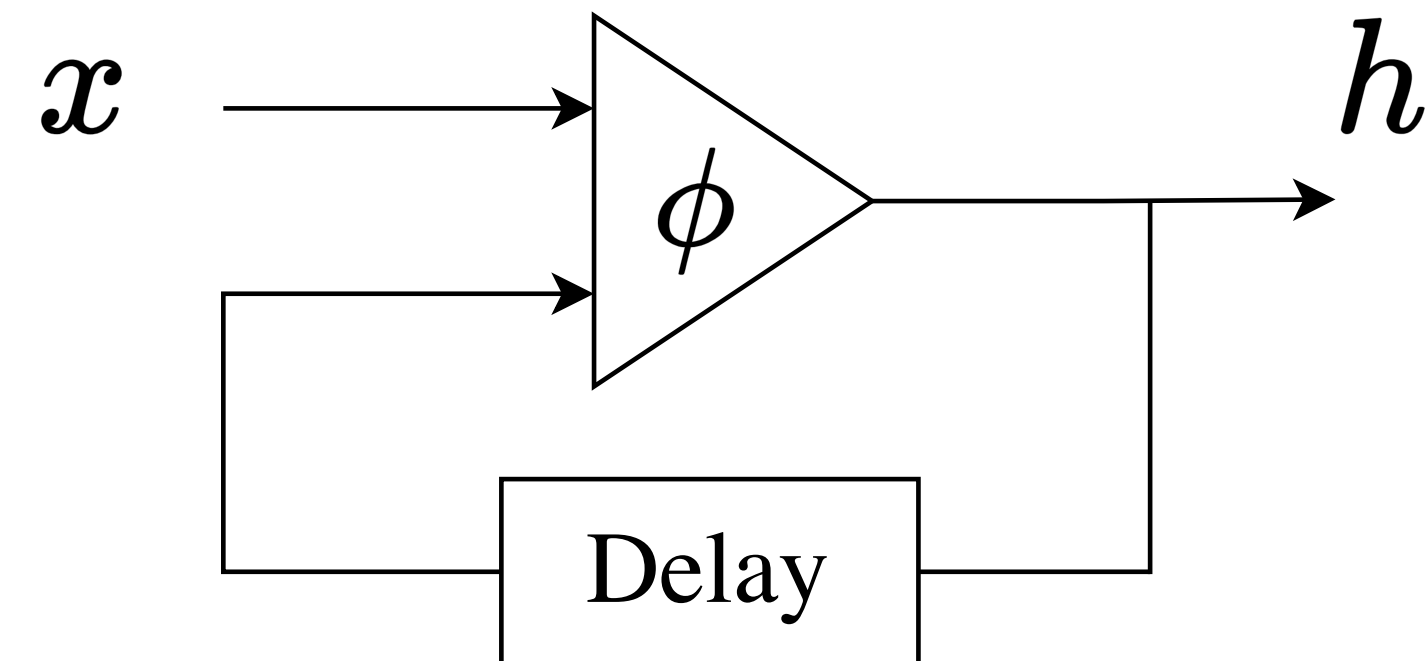


$$h[t] = \phi(x[t], h[t - 1])$$

Recurrent neural networks can be seen as dynamical systems



$$h[t] = \phi(x[t], h[t - 1])$$



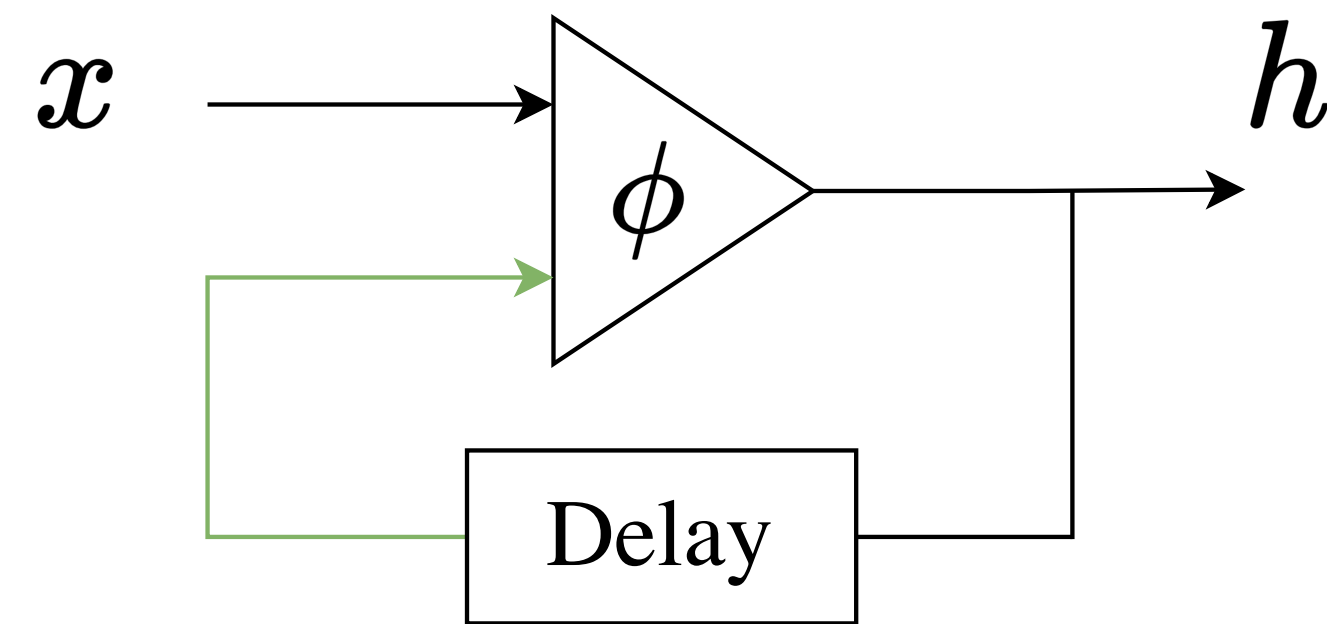
Small feedback loop

Controlling feedbacks leads to a spiking RNN

Controlling feedbacks leads to a spiking RNN

GRU

$$r \in [0,1]$$



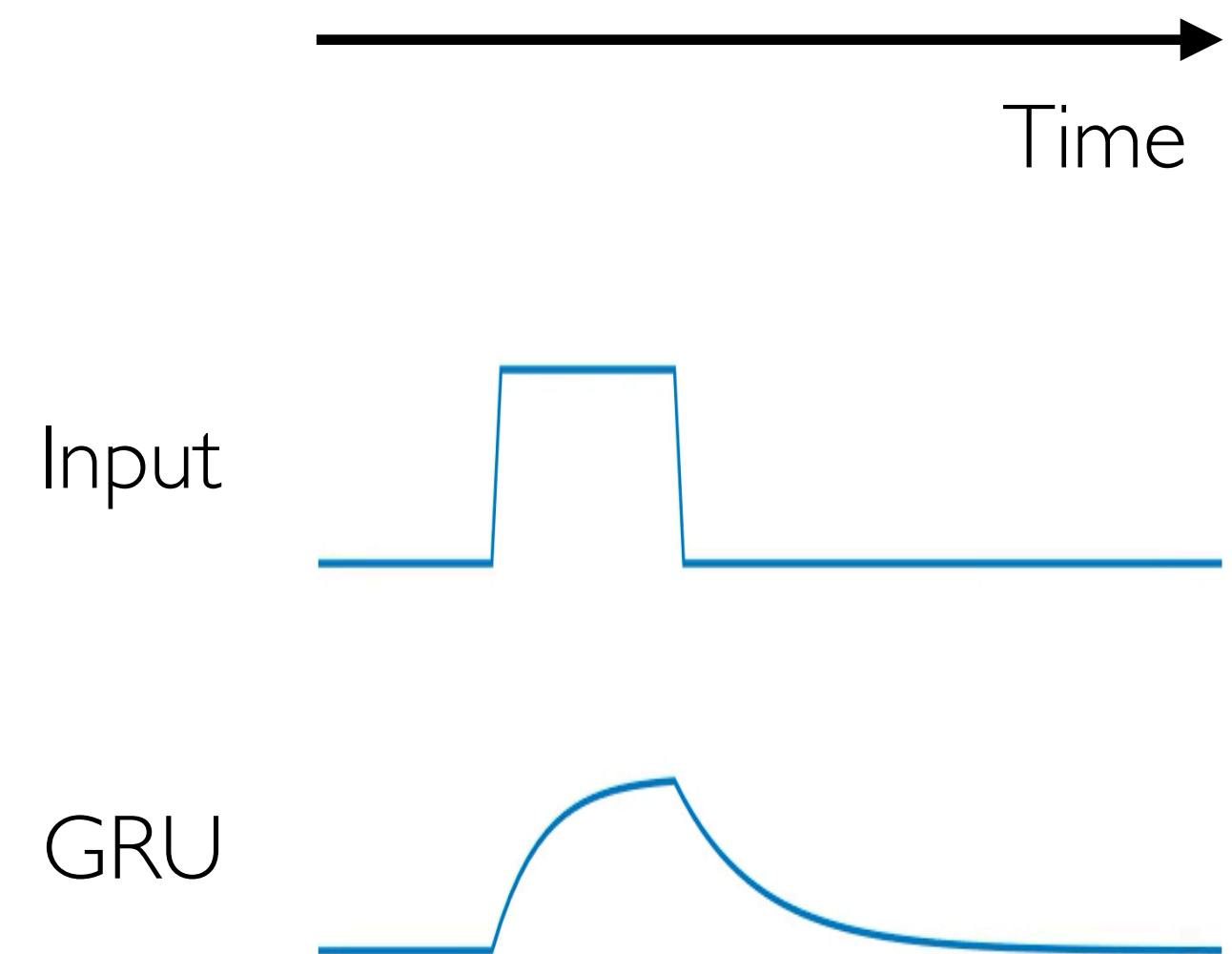
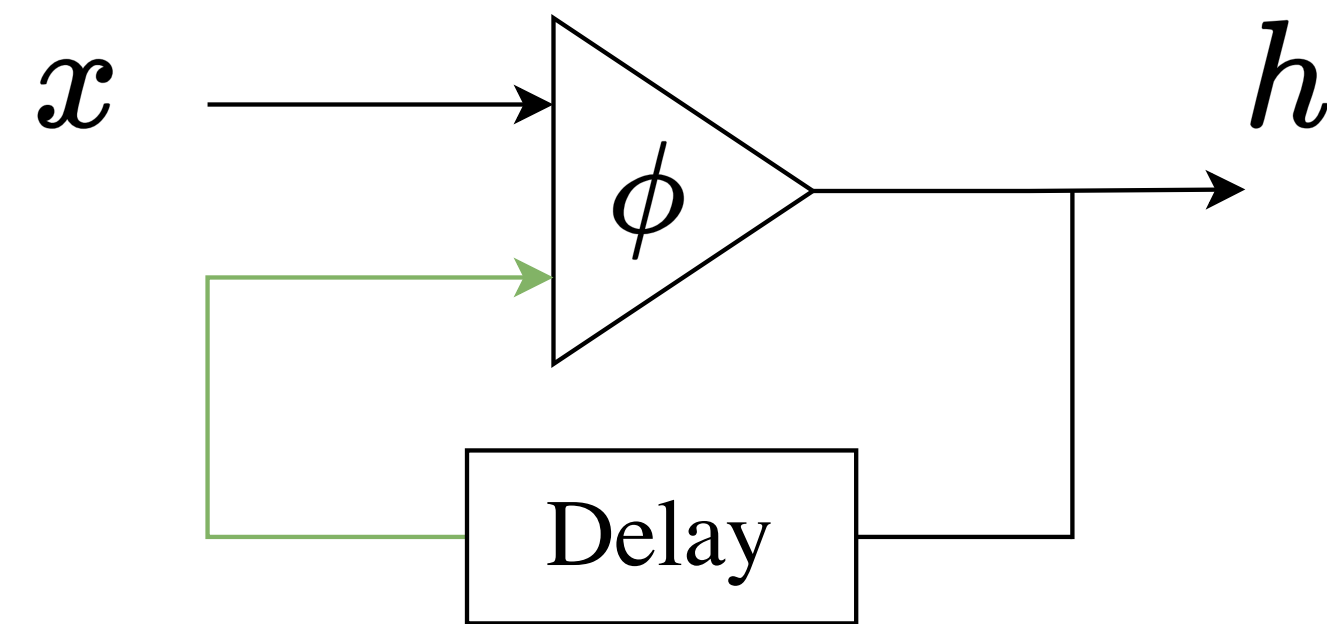
$$\hat{h}[t] = \tanh (x + r \cdot h[t - 1])$$

$$h[t] = z \cdot h[t - 1] + (1 - z) \cdot \hat{h}[t - 1]$$

Controlling feedbacks leads to a spiking RNN

GRU

$$r \in [0,1]$$



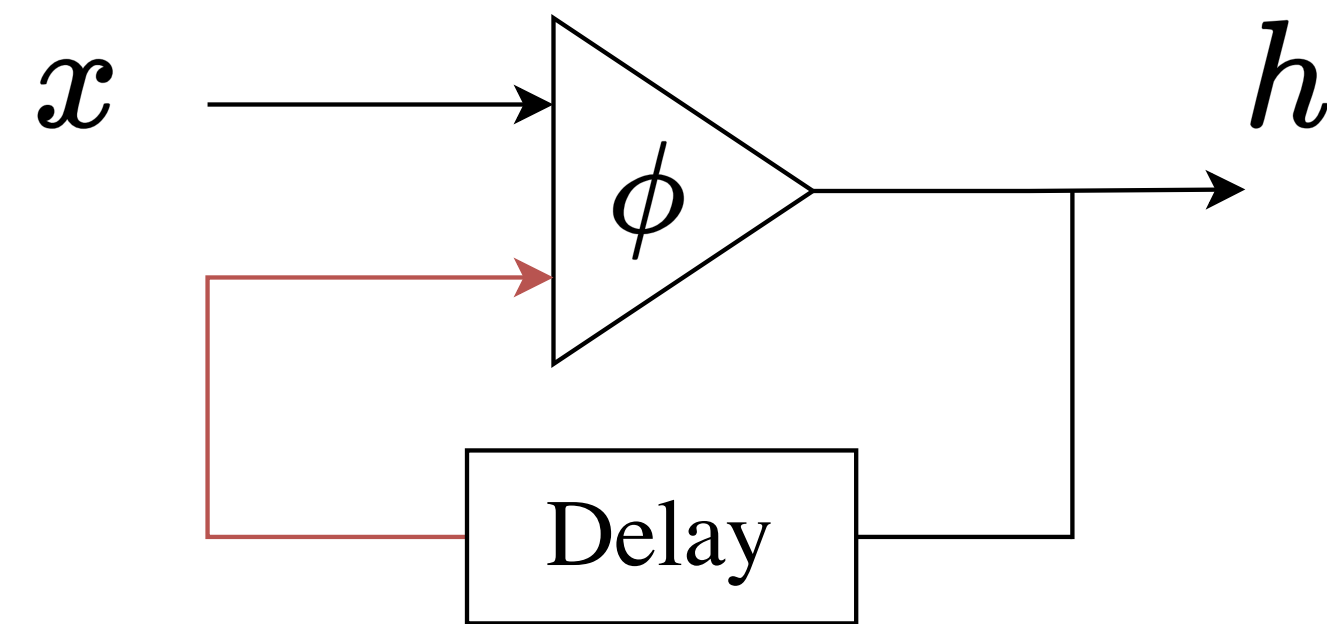
$$\hat{h}[t] = \tanh (x + r \cdot h[t - 1])$$

$$h[t] = z \cdot h[t - 1] + (1 - z) \cdot \hat{h}[t - 1]$$

Controlling feedbacks leads to a spiking RNN

BRC

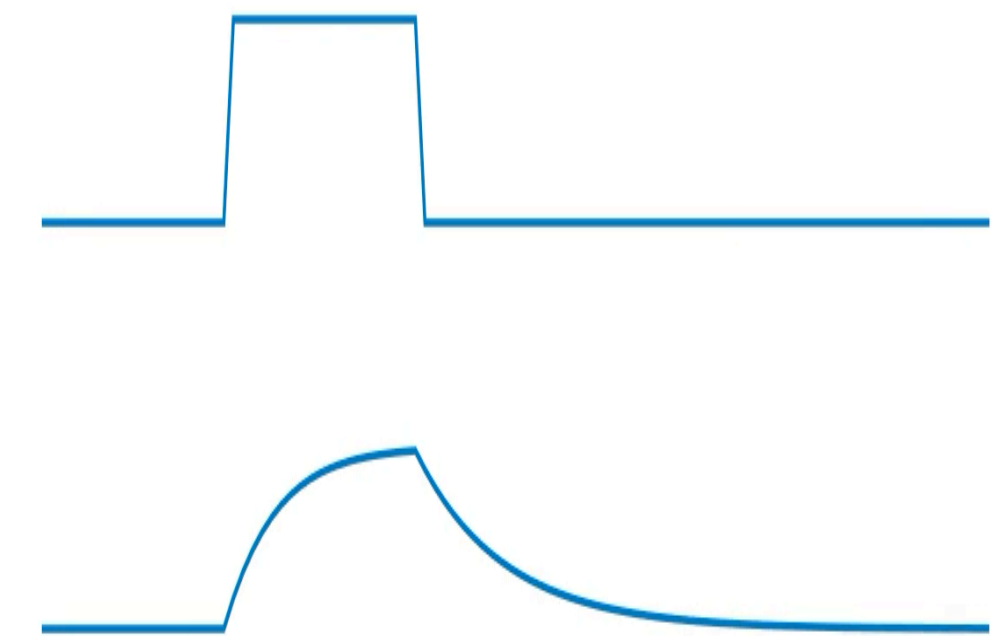
$$r > 1$$



Input

Time

GRU



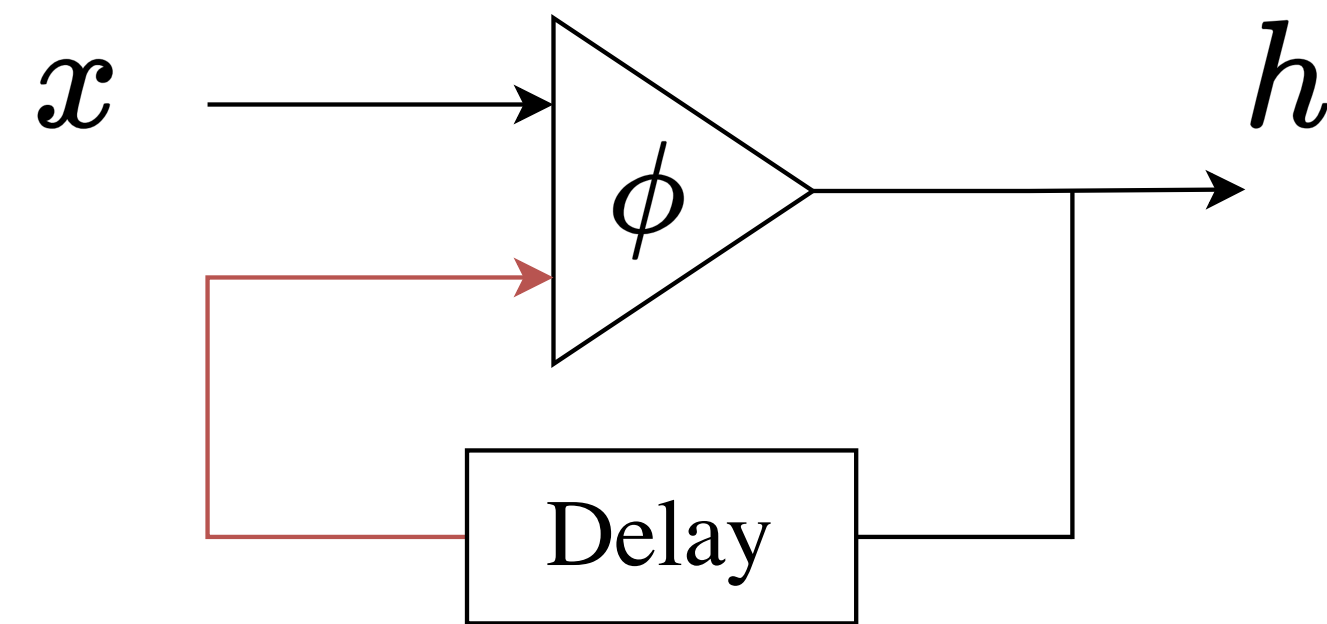
$$\hat{h}[t] = \tanh(x + r \cdot h[t-1])$$

$$h[t] = z \cdot h[t-1] + (1 - z) \cdot \hat{h}[t-1]$$

Controlling feedbacks leads to a spiking RNN

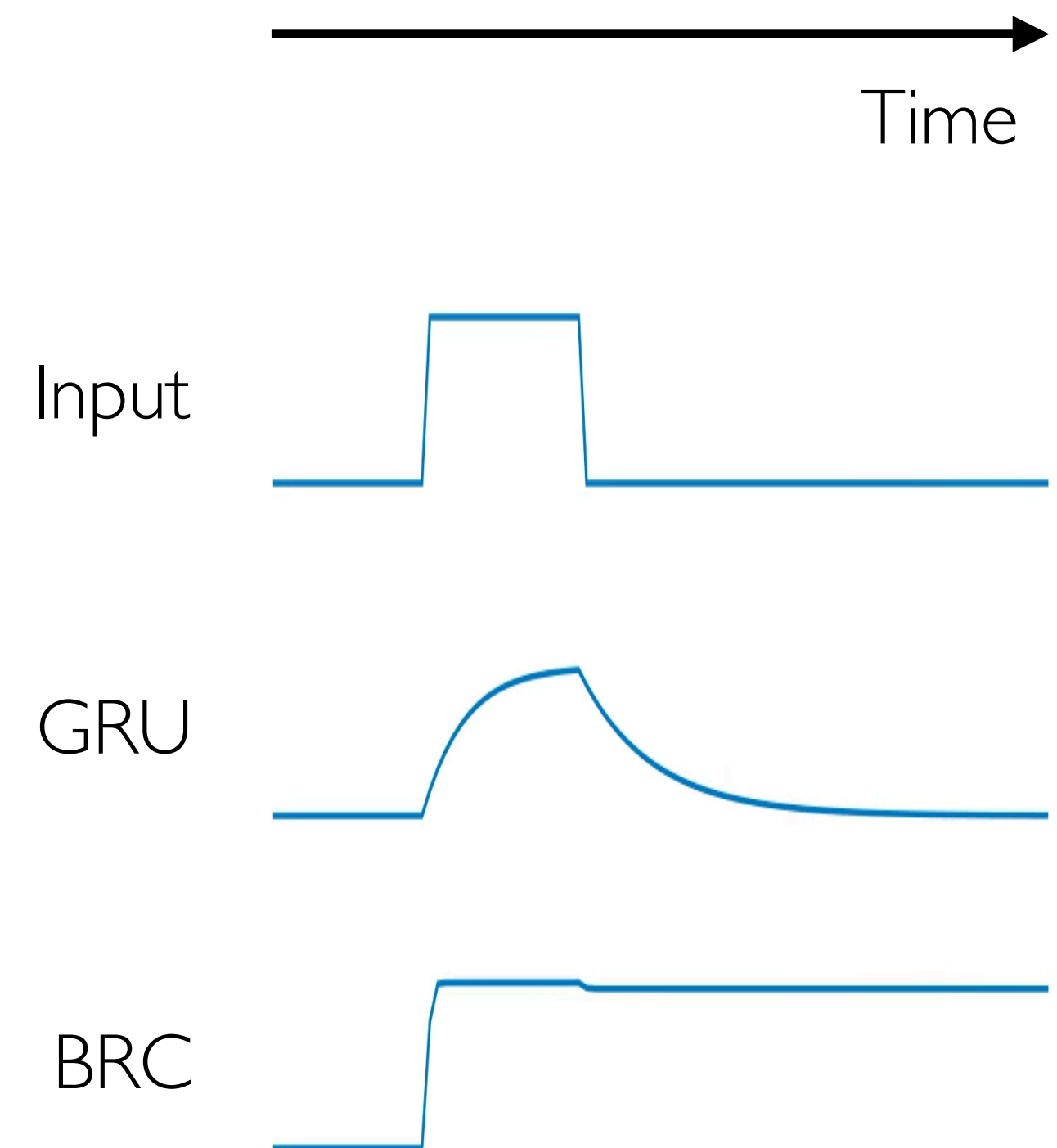
BRC

$$r > 1$$



$$\hat{h}[t] = \tanh(x + r \cdot h[t-1])$$

$$h[t] = z \cdot h[t-1] + (1 - z) \cdot \hat{h}[t-1]$$

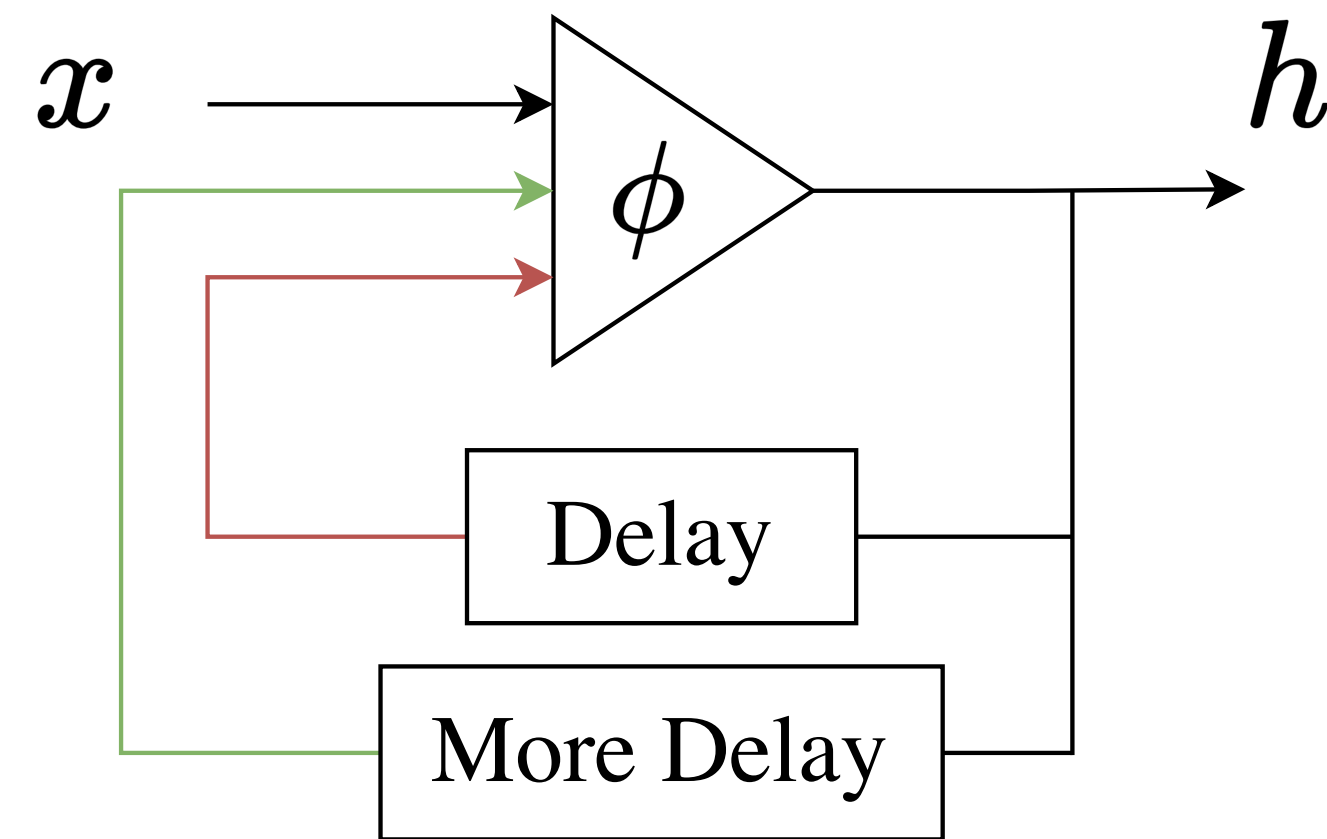


Controlling feedbacks leads to a spiking RNN

SRC

$$r > 1$$

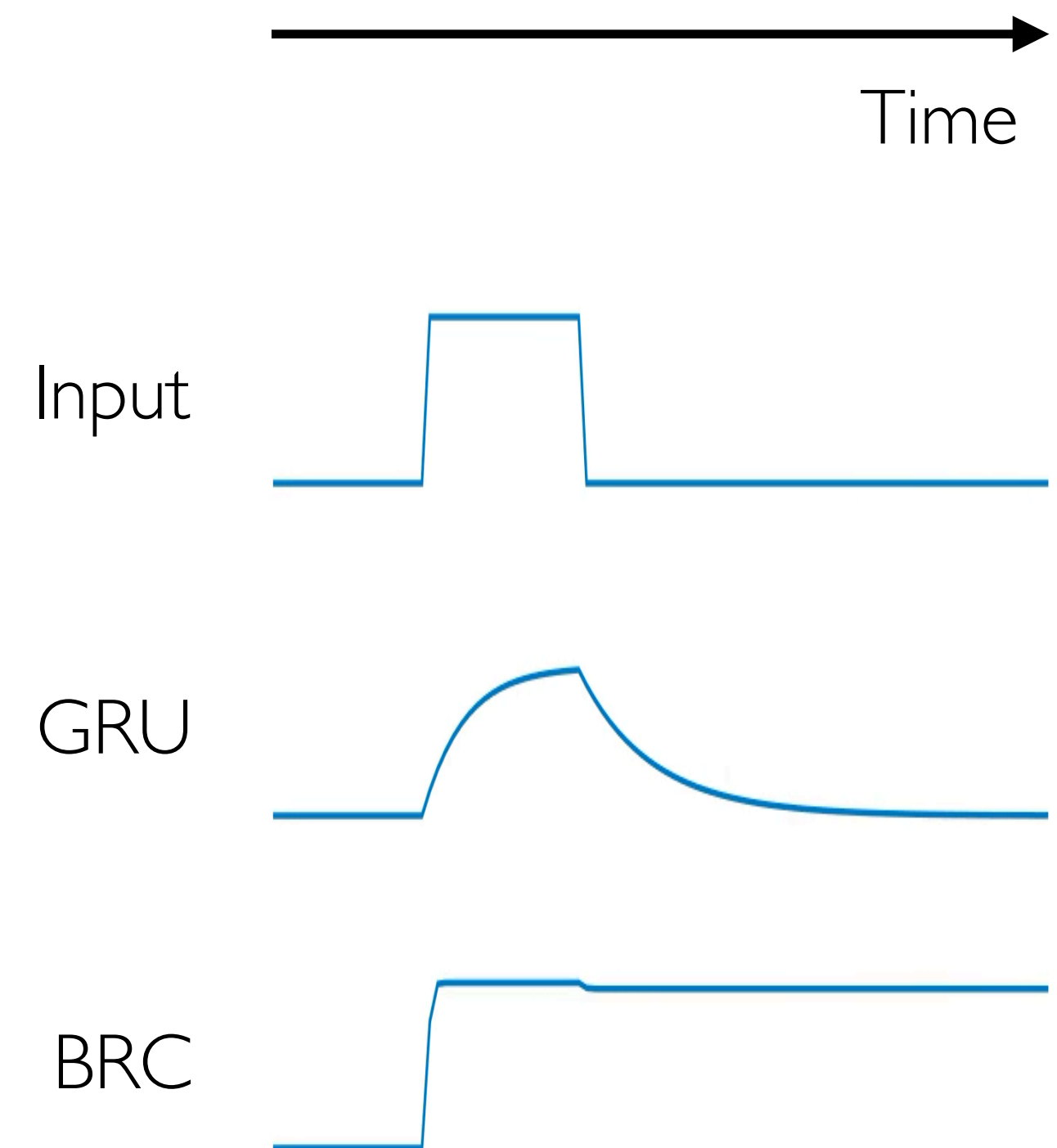
$$r_s = -5$$



$$\hat{h}[t] = \tanh (x + r \cdot h[t-1] + r_s \cdot h_s[t-1])$$

$$h[t] = z \cdot h[t-1] + (1 - z) \cdot \hat{h}[t-1]$$

$$h_s[t] = z_s \cdot h_s[t-1] + (1 - z_s) \cdot h[t-1]$$

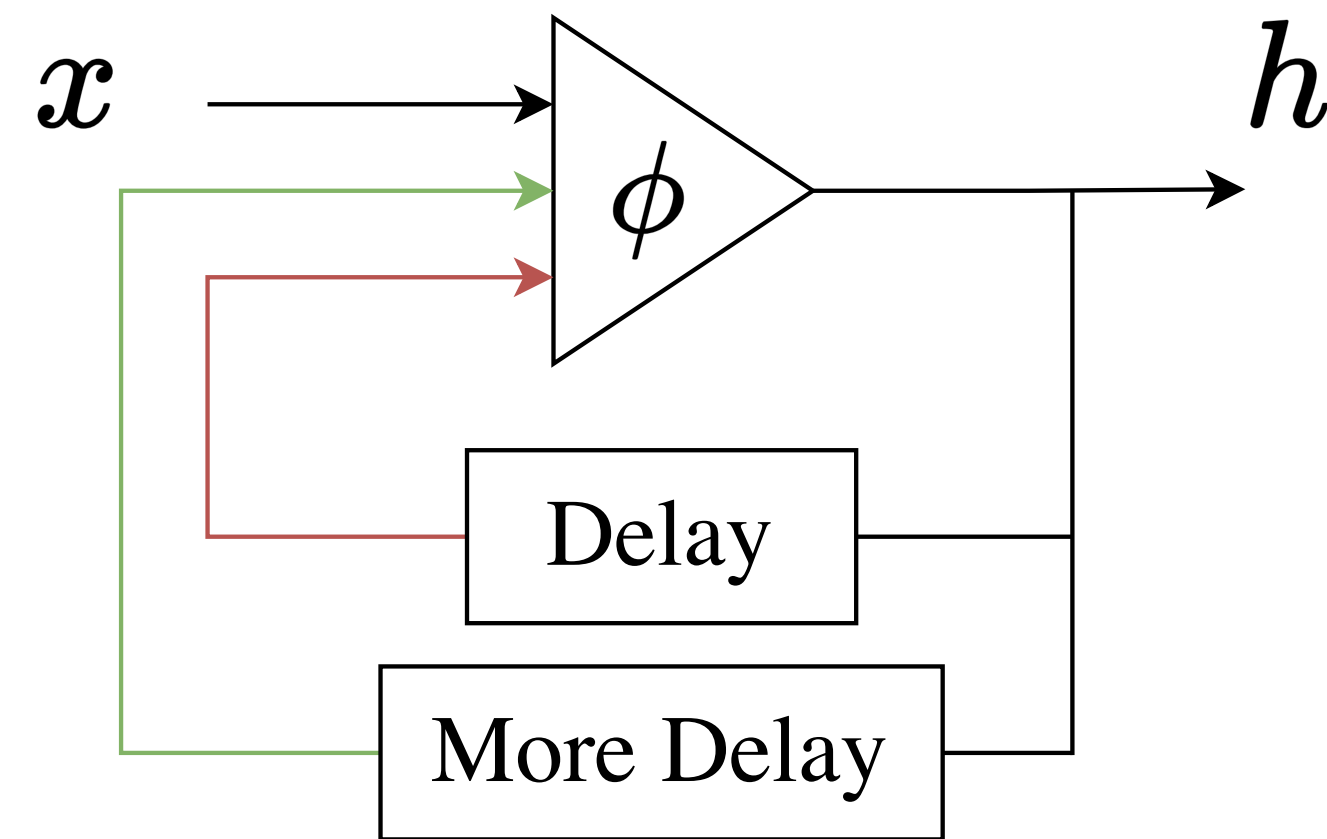


Controlling feedbacks leads to a spiking RNN

SRC

$$r > 1$$

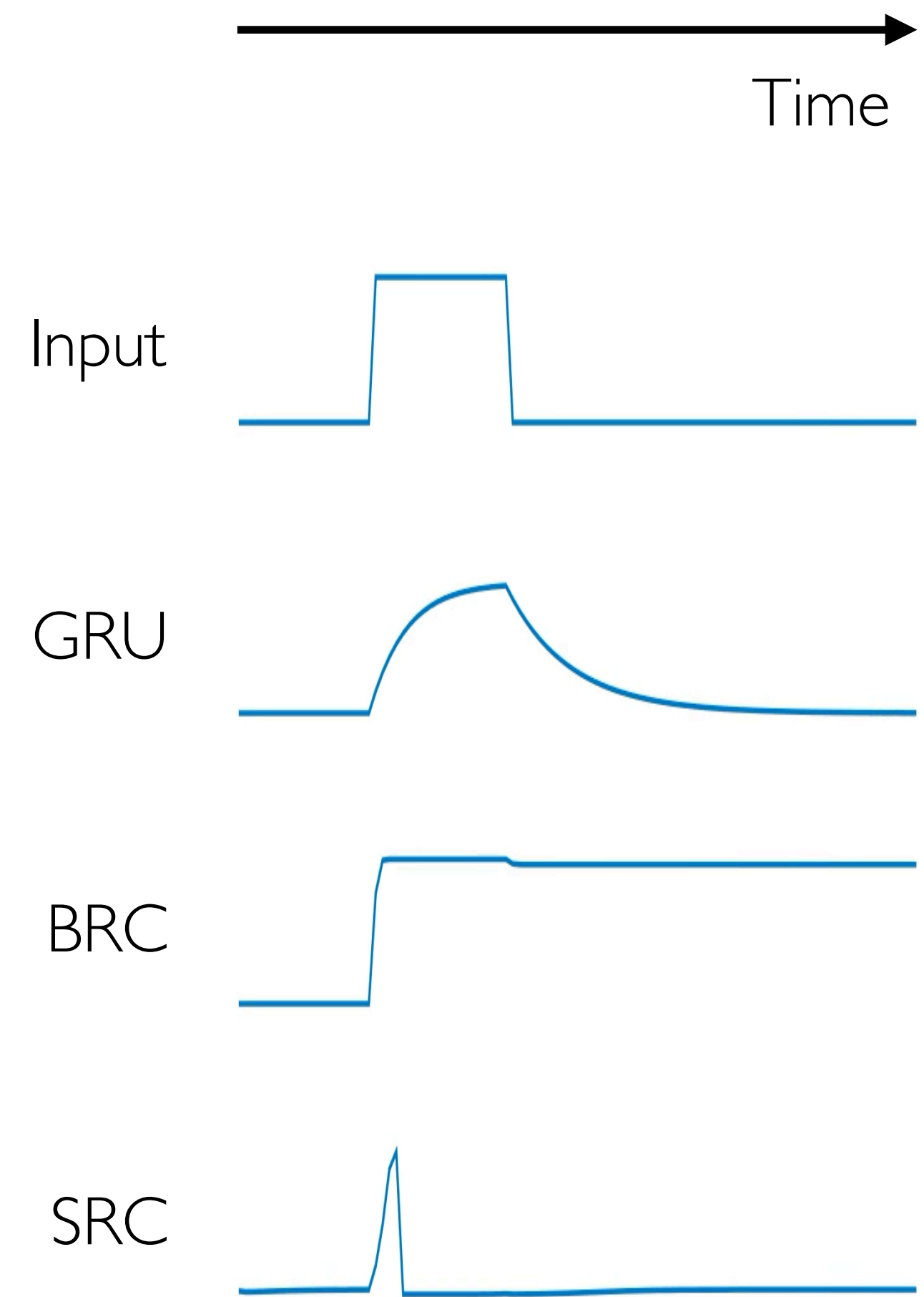
$$r_s = -5$$



$$\hat{h}[t] = \tanh (x + r \cdot h[t-1] + r_s \cdot h_s[t-1])$$

$$h[t] = z \cdot h[t-1] + (1 - z) \cdot \hat{h}[t-1]$$

$$h_s[t] = z_s \cdot h_s[t-1] + (1 - z_s) \cdot h[t-1]$$



SRC networks can be trained like usual recurrent networks

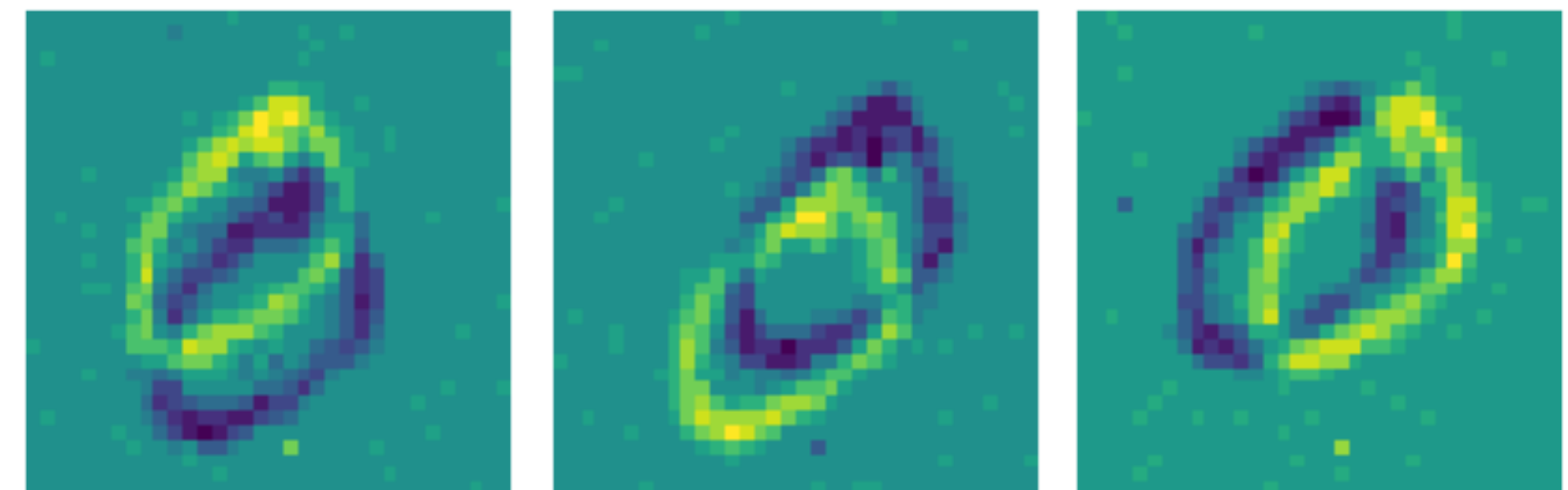
- Tested on simple benchmarks (MNIST and variants)
 - Same performance than LIFs
- Training seems to be more stable
 - Especially with deeper networks
- Resilient to noise in dynamics



MNIST



Fashion-MNIST



Neuromorphic MNIST

3 main directions for future works

3 main directions for future works

Machine Learning

- Try other architectures
 - CNN, etc
- Try more complex benchmarks

3 main directions for future works

Machine Learning

- Try other architectures
 - CNN, etc
- Try more complex benchmarks

Neuro-inspired

- Modify SRC equations to add new firing patterns
- See if model learns to use them

3 main directions for future works

Machine Learning

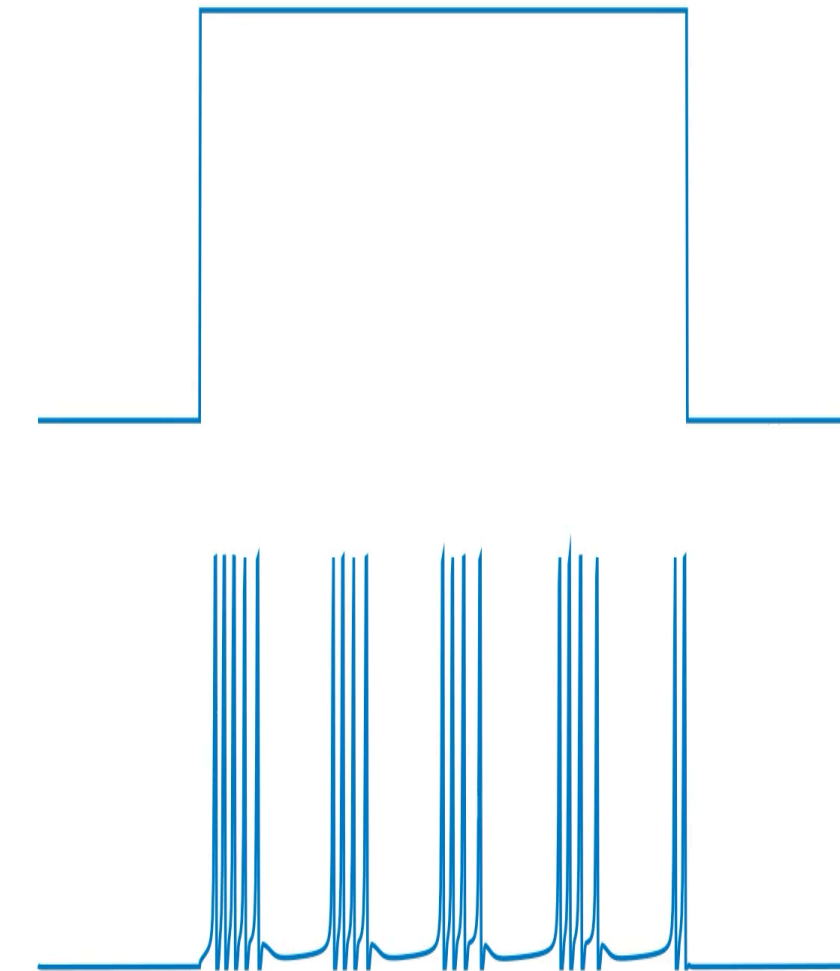
- Try other architectures
 - CNN, etc
- Try more complex benchmarks

Neuro-inspired

- Modify SRC equations to add new firing patterns
- See if model learns to use them

Input

Output



3 main directions for future works

Machine Learning

- Try other architectures
 - CNN, etc
- Try more complex benchmarks

Neuro-inspired

- Modify SRC equations to add new firing patterns
- See if model learns to use them

Electronics

- Try SRC on hardware
 - FPGA
 - Analog circuit

3 main directions for future works

Machine Learning

- Try other architectures
 - CNN, etc
- Try more complex benchmarks

Neuro-inspired

- Modify SRC equations to add new firing patterns
- See if model learns to use them

Electronics

- Try SRC on hardware
 - FPGA
 - Analog circuit

Thank you !

References

- Recurrent neural networks
 - GRU
Cho, K., et al. “Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation.” In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1724–34.
 - BRC
Vecoven, N., et al. “A Bio-Inspired Bistable Recurrent Cell Allows for Long-Lasting Memory.” PLOS ONE 16, no. 6 (June 8, 2021): e0252676.
 - Bursting SRC
Bernard, A., Drion, G. “Design and Performance Analysis of a Neuromodulable Spiking Recurrent Cell in the Context of Supervised Learning,” September 3, 2023.
- Benchmarks
 - MNIST
LeCun, Y., et al. “Gradient-Based Learning Applied to Document Recognition.” Proceedings of the IEEE 86, no. 11 (November 1998): 2278–2324.
 - Fashion-MNIST
Xiao, H., et al. “Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms.” arXiv, September 15, 2017.
 - Neuromorphic MNIST
Orchard, G., et al. “Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades.” Frontiers in Neuroscience 9 (2015).