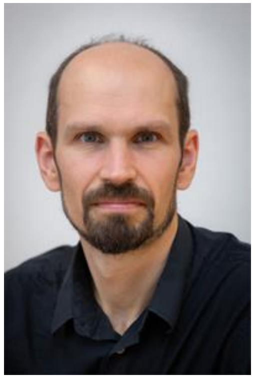
Symbolic computation in Spiking Neural Networks:

Spike-frequency adaptation enables cognitive computations

Ceca Kraišniković
Graz University of Technology, Austria
Lab of Dr. Robert Legenstein




Darjan Salaj Anand Subramoney Guillaume Bellec Robert Legenstein Wolfgang Maass

D. Salaj, A. Subramoney, C. Kraišnikovic, G. Bellec, R. Legenstein, and W. Maass. [Spike-frequency adaptation provides a long short-term memory to networks of spiking neurons.] *bioRxiv*, 2020.

On symbolic computation

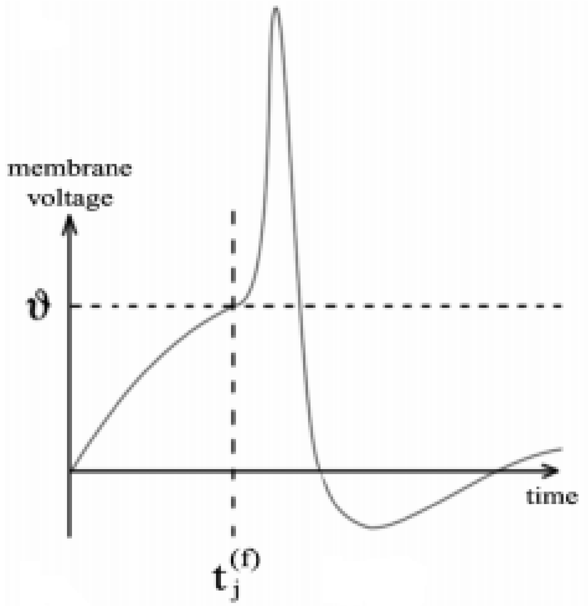
- “Symbolic computation” uses symbolic expressions.
  - Examples: x + y, instruction/rule, algorithm

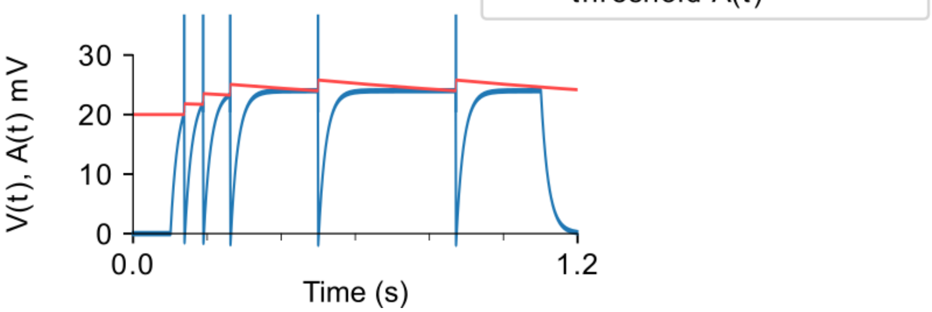
- Challenges for artificial neural networks:
  - Flexible cognitive control

    RED, GREEN, BLUE, PURPLE
  - Free generalization

    e.g., wug - wugged

Leaky Integrate-and-Fire neuron





LIF neurons are augmented with an adaptive threshold (**ALIF**).

Wulfram Gerstner and Werner Kistler. *Spiking Neuron Models: An Introduction*. Cambridge University Press, New York, NY, USA, 2002.

Allen Institute. Allen Cell Types Database Technical white paper: GLIF models. http://help.brainmap.org/download/attachments/8323525/glifmodels.pdf. Technical report, October 2017. v4.

12AX task

- First introduced in (O'Reilly and Frank, 2006)
- Input symbols: {1, 2, A, X, B, Y, C, Z}
- Output symbols: {L, R}

- Input: 1 … A [CZ] X … B ... Y
  Target: L … L … R … L … L

- Input: 2 … A … X … B [CZ] Y
  Target: L … L … L … L … R

- Performance: 97.79% fully correct episodes.

R. C. O'Reilly and M. J. Frank. Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia. Neural computation, 18(2):283-328, 2006.

Brain-like operations on sequences

- Input and output symbols: English alphabet
- Commands: DUPLICATE and REVERSE string

- An example input/output pair for DUPLICATE operation:

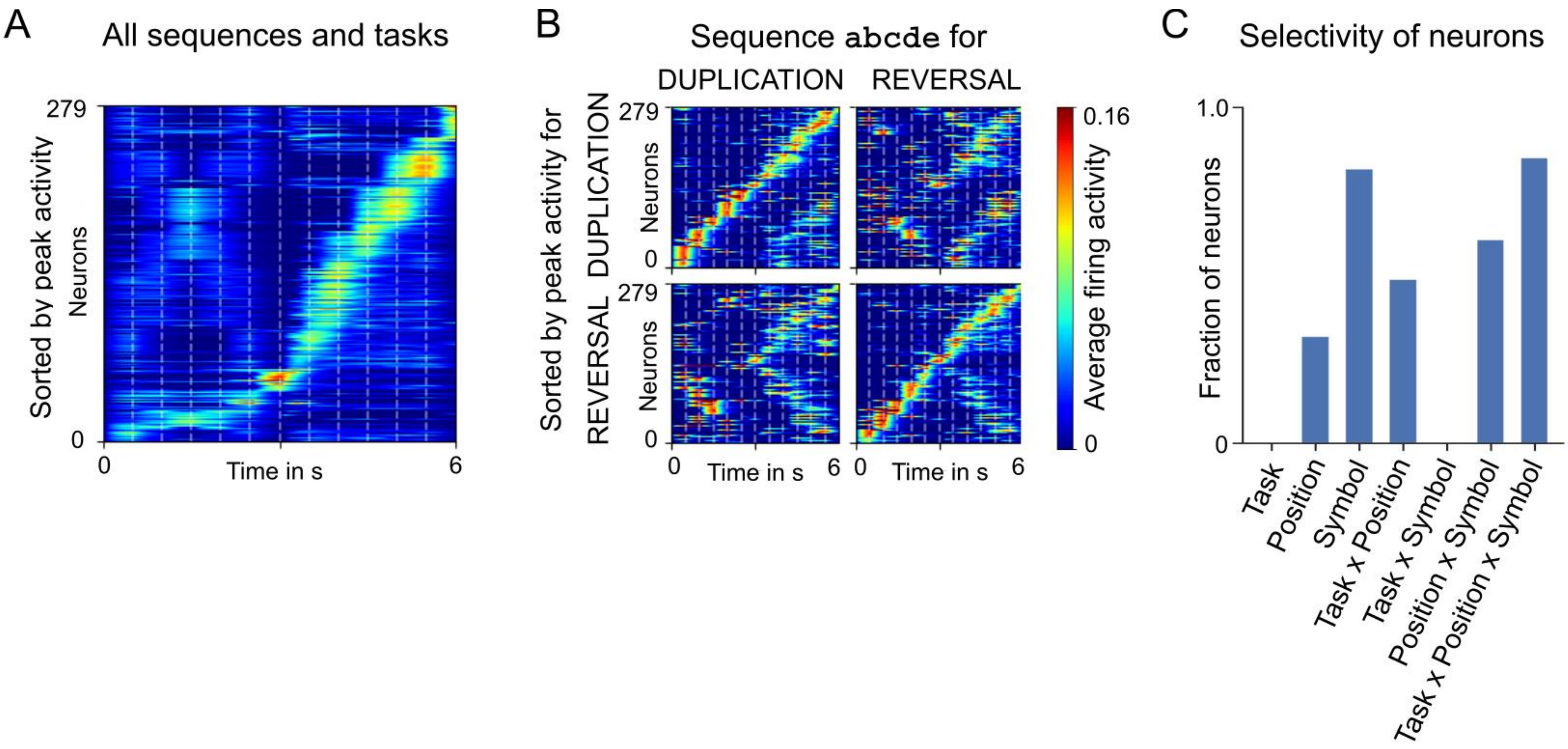
Input	a	b	c	x	y	*	?	?	?	?	?	?
Output							a	b	c	x	y	*

- An example input/output pair for REVERSE operation:

Input	a	b	c	x	y	*	?	?	?	?	?	?
Output							y	x	c	b	a	*

- Performance: 95.88% fully correct output strings, tested on previously unseen strings.

Emergence of neural codes



Thank you for your attention!
Questions?

Neuron Model

LIF

$$\tau_m \frac{du}{dt} = -u(t) + RI(t)$$

$$I_i(t) = \sum_j w_{ij} \sum_f \alpha(t - t_j^{(f)})$$

$$\alpha(s) = A\, exp\left(-\frac{s}{\tau_s}\right)\Theta(s)$$

τ_m- membrane time constant,
$u(t)$ - membrane voltage,
$I(t)$ - current, w_{ij} - efficacy,
$t_j^{(f)}$- firing time of neuron j,
τ_s-time decay constant,
$\Theta(s)$ - Heaviside step function

Adaptive LIF

$$\tau_m \frac{du_j}{dt} = -u_j(t) + R_m I_j(t)$$

$$\tau_{a,j} \frac{db_j}{dt} = b_j^0 - b_j(t)$$

τ_m – membrane time constant
$\tau_{a,j}$ – adaptation time constant

In discrete time:

$$u_j(t+dt) = \alpha u_j(t) + (1-\alpha) R_m I_j(t) - b_j(t) z_j(t)$$

$$b_j(t+dt) = \rho_j b_j(t) + \left(1-\rho_j\right) z_j(t)$$

$$\alpha = exp\left(-\frac{dt}{\tau_m}\right)$$

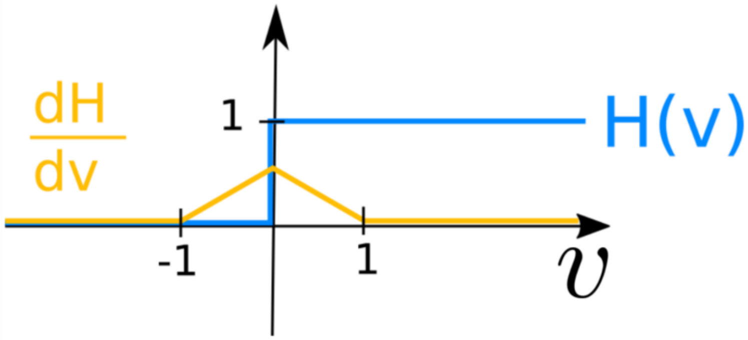
$$\rho_j = \exp\left(-\frac{dt}{\tau_{a,j}}\right)$$

Backpropagation Through Time (BPTT)

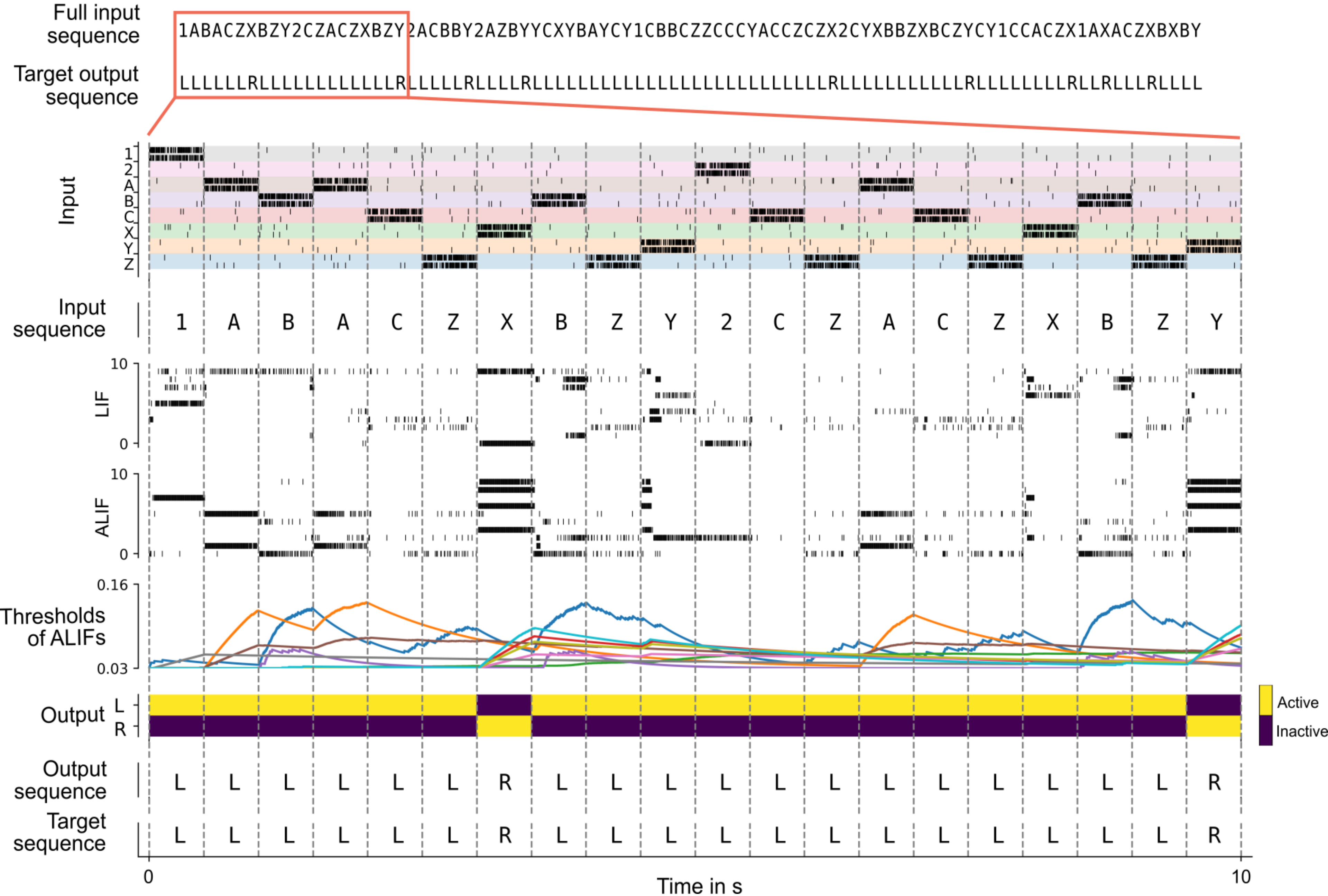
- BPTT is used to train RNNs.
  - Gradients are propagated through many steps.
- Outputs of spiking neurons are non-differentiable.
  - Dampened pseudo-derivative can be used instead.

$$\frac{dH}{dv} = \gamma \max\{0, 1 - |v|\},$$

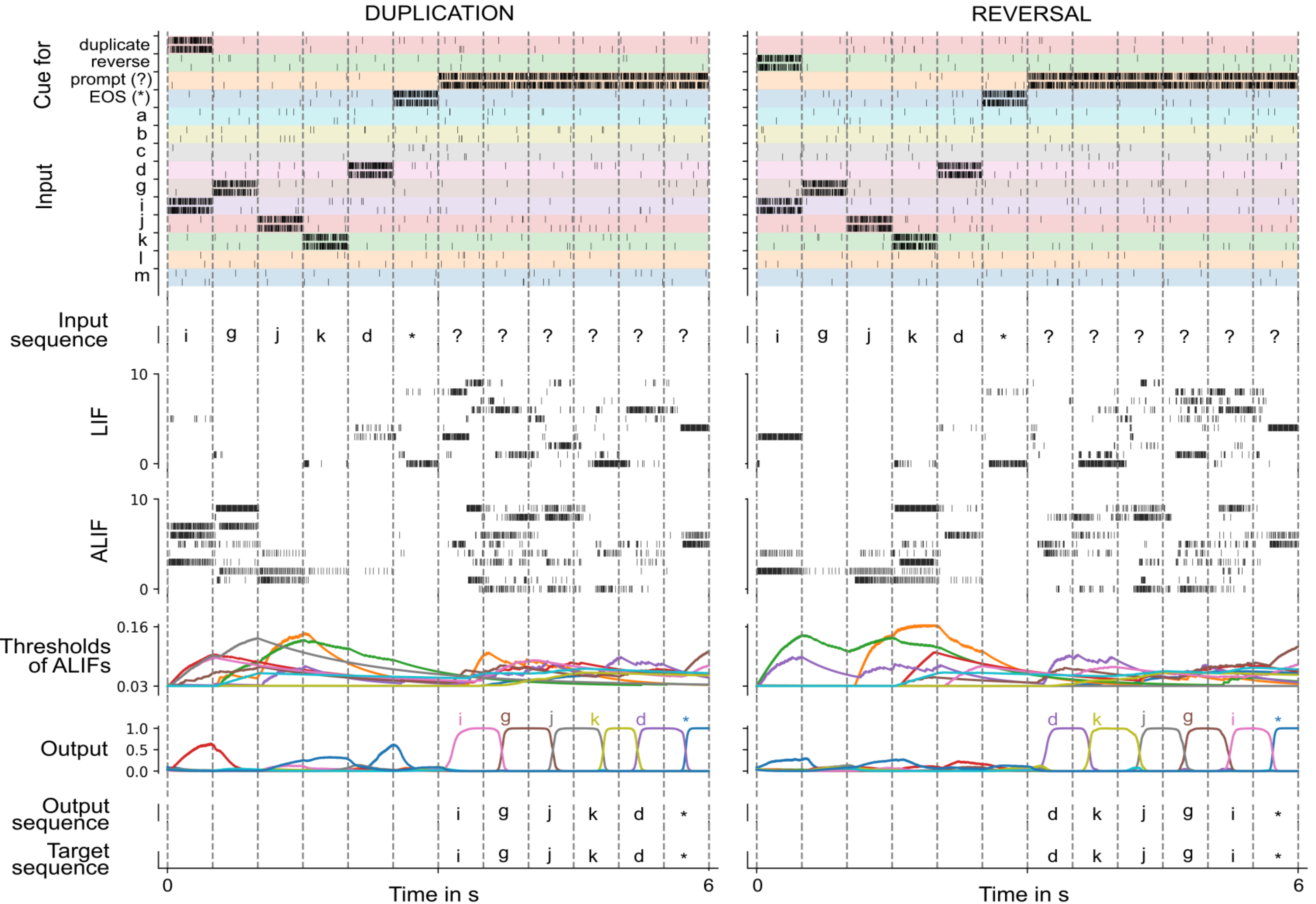
v – normalized membrane potential, γ – dampening factor.



[G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *32nd Conference on Neural Information Processing Systems (NIPS 2018), Montreal, Canada*, 2018.]



Ceca Kraišniković
Graz University of Technology, Austria



Ceca Kraišniković
Graz University of Technology, Austria