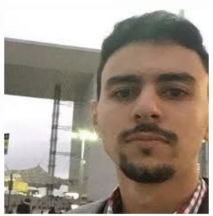
Learning Delays in Spiking Neural Networks using Dilated Convolutions with Learnable Spacings

Ilyass Hammouamri,



PhD Student
CNRS-CerCo
ilyass.hammouamri@cnrs.fr

Ismail Khalfaoui-Hassani,

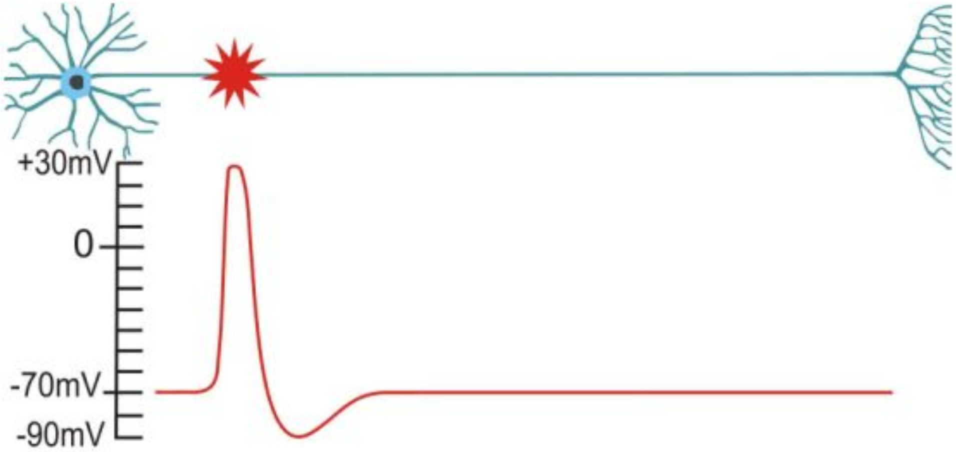
Researcher
University of Toulouse
ismail.khalfaoui-hassani@univ-tlse3.fr

Timothée Masquelier

Research Director
CNRS-CerCo
timothee.masquelier@cnrs.fr

Delays ?

Delay refers to the time needed for one spike to travel between the presynaptic and postsynaptic neurons.



Why delays matter ?

Biological perspective

- Evidence of heterogeneous delays in the brain
- Learning is associated with changes in myelin in brain regions relevant for performing a task.
- Blocking myelin production impairs new learning in some tasks.

Theoretical perspective

- Delays can greatly increase the expressive power in Spiking Neural Networks (Maass & Schmitt 1999)
- Delayed dynamical systems can exhibit astonishingly rich and complex dynamics (e.g. Foss & Milton 2000)

SNN model with delays

A feed-forward SNN with delays is parameterized with $W = (w_{ij}^{(l)})$ and $D = (d_{ij}^{(l)})$

where the input of neuron i at layer l is :

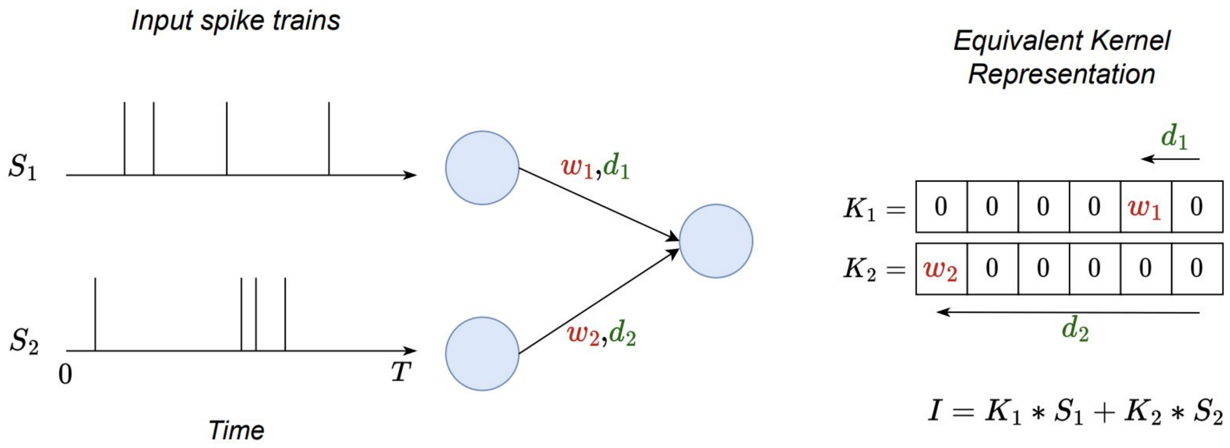
$$I_i^{(l)}[t] = \sum_j w_{ij}^{(l)} S_j^{(l-1)}[t - d_{ij}^{(l)}]$$

Learning weights ➔ surrogate gradient (Neftci et al., 2018; Shrestha & Orchard, 2018)

Learning delays ➔ ?

Delays as a 1D Temporal Convolution

We model every synaptic connection which is a couple (w, d) by a kernel k.



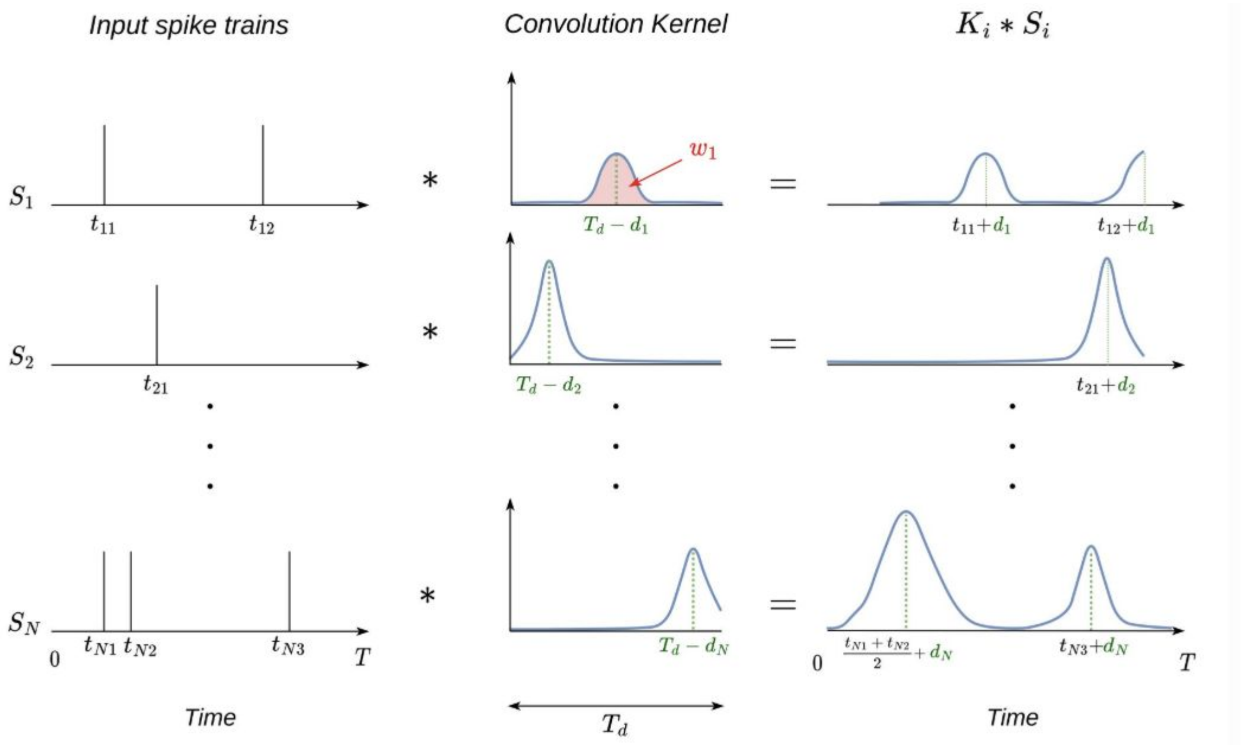
In general:

$$k_{ij}^{(l)}[n] = \begin{cases} w_{ij}^{(l)} & \text{if } n = T_d - d_{ij}^{(l)} - 1 \\ 0 & \text{otherwise} \end{cases}$$

$$I_i^{(l)} = \sum_j k_{ij}^{(l)} * S_j^{(l-1)}$$

Where T_d is the kernel size or maximum delay plus one

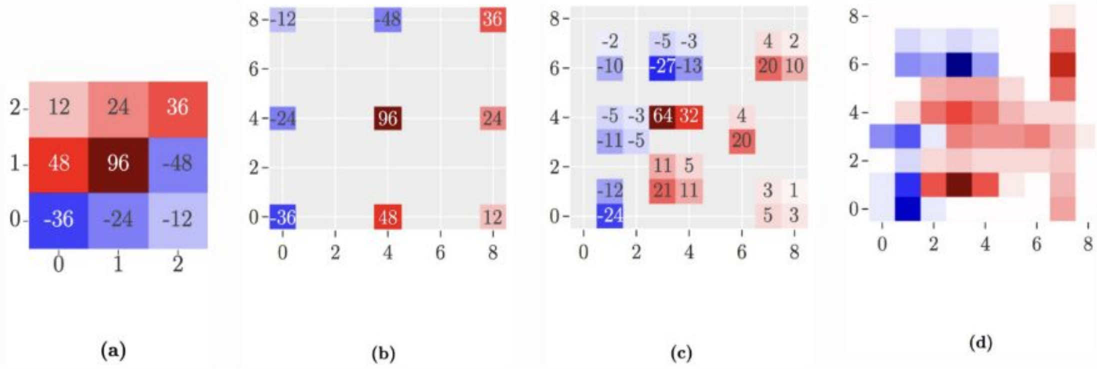
Gaussian Kernel



$$k_{ij}^{(l)}[n] = \frac{w_{ij}^{(l)}}{c} \exp\left(-\frac{1}{2}\left(\frac{n - T_d + d_{ij}^{(l)} + 1}{\sigma_{ij}^{(l)}}\right)^2\right)$$

Dilated Convolutions with Learnable Spacings (DCLS)

DCLS is a recent method for learning the spacings between the non-zero elements in an n-dimensional dilated convolution (Khalfaoui-Hassani et al 2023) .



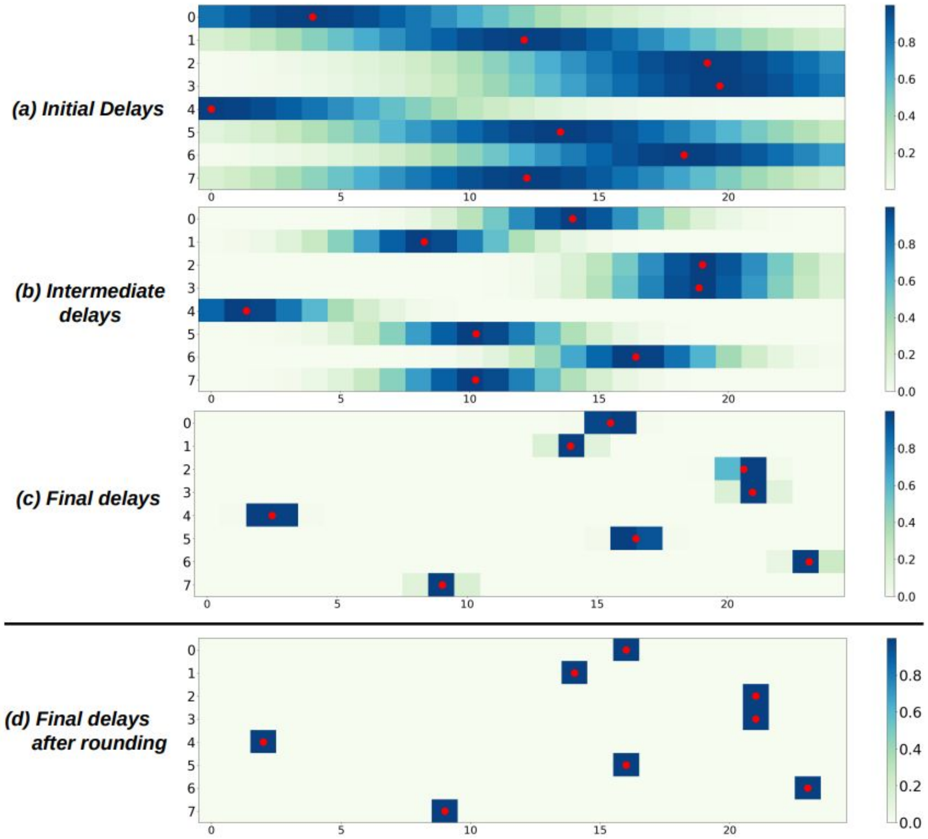
(a): a standard 3 × 3 kernel.
(b): a dilated 3 × 3 kernel with dilation rate 4.
*(c): a 2D-DCLS kernel with a **kernel count** of 9 and a **dilated kernel size** of 9.*
Each weight is spread over up to four adjacent pixels.
(d) the same kernel as (c) with Gaussian interpolation.

The numbers have been rounded in all figures and omitted in (d) for readability.

Learning process

By adjusting the standard deviation σ of the gaussian, we can regulate the temporal scale of dependencies.

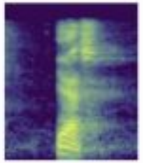
We start with a high sigma and exponentially reduce it throughout the training process, until we end up with a single element rounded value for the delay position.



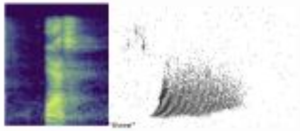
x-axis: time, y-axis: synapse id, red point: position-delay

Tasks and datasets

- **Audio classification** on **GSC (Google Speech Commands v0.02)** [Pete Warden , 2018].
  Metric of interest: accuracy.



- **Audio classification** on **SSC (Spiking Speech Commands)** [Cramer et al. , 2020].
  Metric of interest: accuracy.



- **Audio classification** on **SHD (Spiking Heidelberg Digits)** [Cramer et al. , 2020].
  Metric of interest: accuracy.

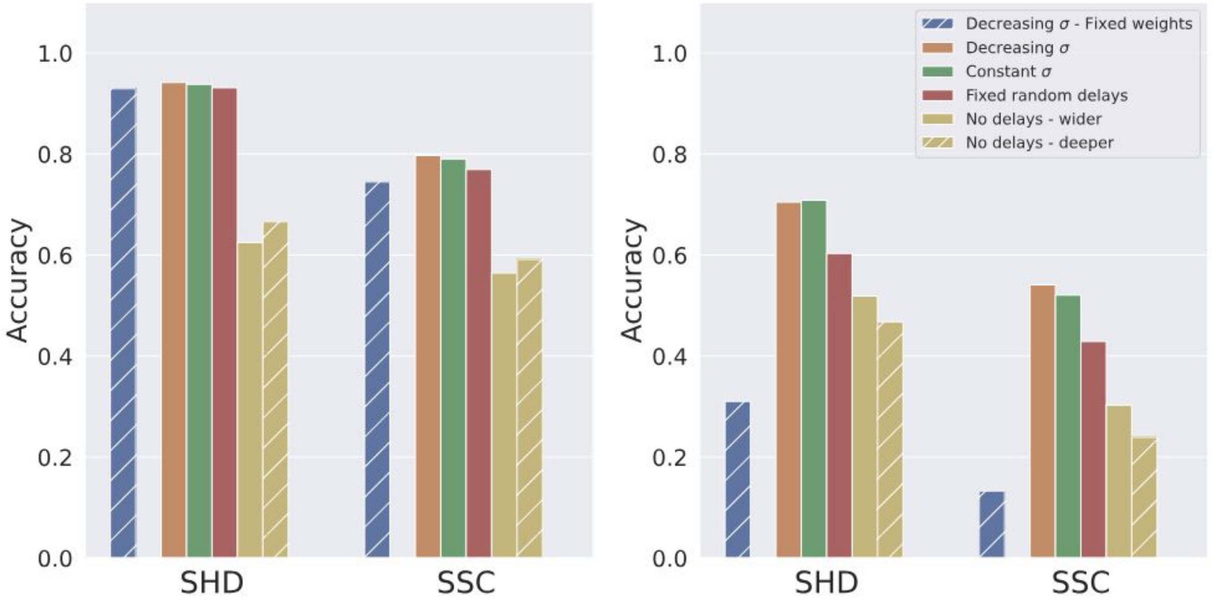
Results

Dataset	Method	Rec.	Delays	#Params	Top1 Acc.
SHD	EventProp-GeNN (Nowotny et al., 2022)	✓	✗	N/a	84.80±1.5%
	Cuba-LIF (Dampfhoffer et al., 2022)	✓	✗	0.14M	87.80±1.1%
	Adaptive SRNN (Yin et al., 2021)	✓	✗	N/a	90.40%
	SNN+Delays (Patiño-Saucedo et al., 2023)	✗	✓	0.1M	90.43%
	TA-SNN (Yao et al., 2021)	✗	✗	N/a	91.08%
	STSC-SNN (Yu et al., 2022)	✗	✗	2.1M	92.36%
	Adaptive Delays (Sun et al., 2023b)	✗	✓	0.1M	92.45%
	DL128-SNN-Dloss (Sun et al., 2023a)	✗	✓	0.14M	92.56%
	Dense Conv Delays (ours)	✗	✓	2.7M	93.44%
	RadLIF (Bittar & Garner, 2022)	✓	✗	3.9M	94.62%
	DCLS-Delays (2L-1KC)	✗	✓	**0.2M**	**95.07±0.24%**
SSC	Recurrent SNN (Cramer et al., 2022)	✓	✗	N/a	50.90 ± 1.1%
	Heter. RSNN (Perez-Nieves et al., 2021)	✓	✗	N/a	57.30%
	SNN-CNN (Sadovsky et al., 2023)	✗	✓	N/a	72.03%
	Adaptive SRNN (Yin et al., 2021)	✓	✗	N/a	74.20%
	SpikGRU (Dampfhoffer et al., 2022)	✓	✗	0.28M	77.00±0.4%
	RadLIF (Bittar & Garner, 2022)	✓	✗	3.9M	77.40%
	Dense Conv Delays 2L (ours)	✗	✓	10.9M	77.86%
	Dense Conv Delays 3L (ours)	✗	✓	19M	78.44%
	DCLS-Delays (2L-1KC)	✗	✓	**0.7M**	**79.77±0.09%**
	DCLS-Delays (2L-2KC)	✗	✓	**1.4M**	**80.16±0.09%**
	DCLS-Delays (3L-1KC)	✗	✓	**1.2M**	**80.29±0.06%**
	DCLS-Delays (3L-2KC)	✗	✓	**2.5M**	**80.69±0.21%**
GSC-35	MSAT (He et al., 2023)	✗	✗	N/a	87.33%
	Dense Conv Delays 2L (ours)	✗	✓	10.9M	92.97%
	Dense Conv Delays 3L (ours)	✗	✓	19M	93.19%
	RadLIF (Bittar & Garner, 2022)	✓	✗	1.2M	94.51%
	DCLS-Delays (2L-1KC)	✗	✓	**0.7M**	**94.91±0.09%**
	DCLS-Delays (2L-2KC)	✗	✓	**1.4M**	**95.00±0.06%**
	DCLS-Delays (3L-1KC)	✗	✓	**1.2M**	**95.29±0.11%**
	DCLS-Delays (3L-2KC)	✗	✓	**2.5M**	**95.35±0.04%**

nL-mKC stands for a model with n hidden layers and kernel count m, where kernel count denotes the number of non-zero elements in the kernel. "Rec." denotes recurrent connections.

Ablation Study

- Using Delays significantly outperforms the No delays SNN.
- Using Fixed random delays is only slightly less better than learning them in the FC case.
- Learning delays is more important on the sparse case, and decreasing sigma helps
- The optimal is to learn jointly delays and weights



(a) FC: Fully Connected (b) S: Sparse connections

Conclusion

- Learning delays helps, especially in temporal tasks.
- We proposed an efficient way to do it, leading to new SOTAs on audio classification tasks.
- Future work could be focused on trying our method on other network architectures like convolutional SNNs.

Thanks !

github.com/Thvnvtos/SNN-delays

github.com/K-H-Ismail/Dilated-Convolution-with-Learnable-Spacings-PyTorch

ilyass.hammouamri@cnrs.fr
ismail.khalfaoui-hassani@univ-tlse3.fr
timothee.masquelier@cnrs.fr