KU LEUVEN imec UC San Diego

Towards Chip-in-the-loop Spiking Neural Network Training via Metropolis-Hastings Sampling

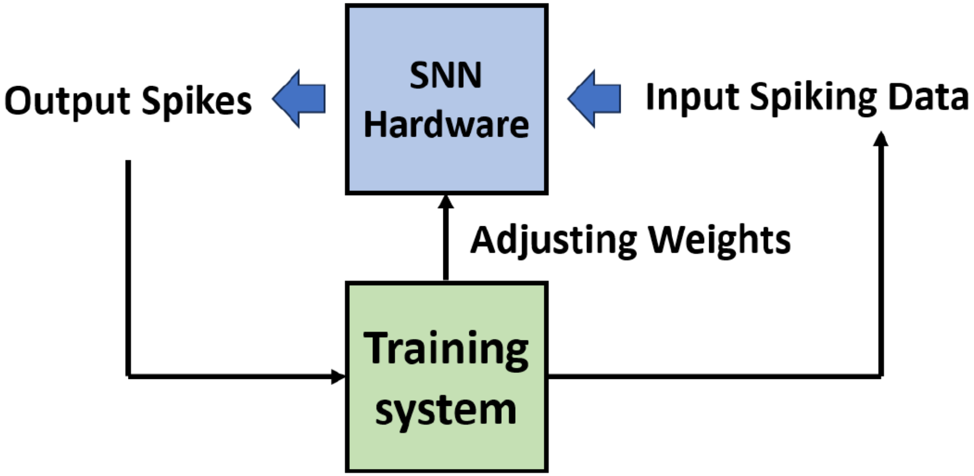
Ali Safa, *Vikrant Jaltare, Samira Sebt, Kameron Gano, Johannes Leugering, Georges Gielen, Gert Cauwenberghs*

imec, KU Leuven, Belgium & Institute for Neural Computation, UC San Diego

Introduction

Model-free SNN training via Metropolis-Hastings Sampling

- SNNs have gained huge attention for **ultra-low power AI** application in extreme edge domains *such as Personalized Healthcare and IoT*.
- A general promising path for pushing the boundaries of SNN hardware efficiency lies in the use of **“unconventional” computing technologies** *such as e.g., Analog Sub-threshold designs and Memristor hardware* [Indiveri et al. 2011, Payvand et al. 2022].
- But because of their **increased variability** vs. digital designs, a **precise model** of the underlying SNN hardware is more challenging to obtain.
- This motivates the exploration of *chip-in-the-loop* [Mitchell & Schuman, 2021] *and* **<u>model-free</u> SNN training**



$P(W|D) \sim L(D|W)P(W)$

Metropolis-Hastings

(Bayesian Inference)

vs.

$$W \leftarrow W - \eta \frac{\partial E}{\partial W}$$

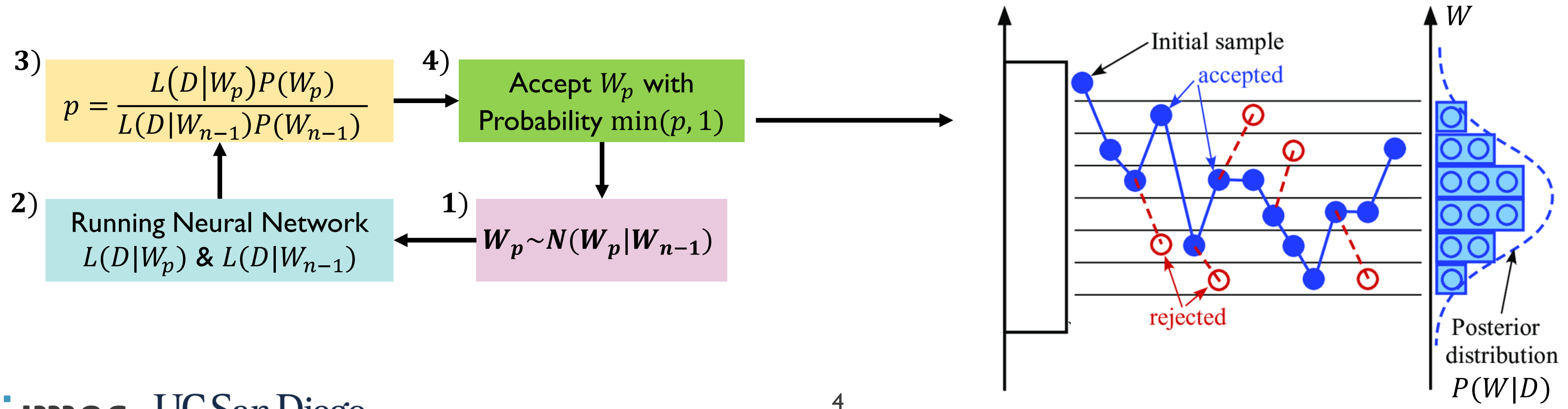
***Backprop** – gradient descent*

Bayesian Inference: a short recap

- Bayesian Inference for *inferring the weights of a Neural Network.*
- Bayes Rule: $P(W_{n+1}|D) = \frac{L(D|W_n)P(W_n)}{P(D)} = \frac{L(D|W_n)P(W_n)}{\int P(D,W')dW'}$
- $L(D|W_n)$ is the Likelihood of the Data D given the model W_n (linked to Loss function).
- $P(W_n)$ is the Prior distribution (belief) over W_n.
- $P(W_{n+1}|D)$ is the Posterior distribution of W_{n+1} after integrating the Prior with the data.
- $P(D)$ is the Evidence (expensive to compute).

Metropolis-Hastings Sampling

- Explicitly solving Bayesian Inference is highly compute expensive in high-dimensional spaces due to the Evidence Density $P(D) = \int P(D, W')dW'$ [Jospin et al. 2022].
- **Metropolis-Hastings:** a popular Markov Chain Monte Carlo (MCMC) method for drawing samples from the Posterior $W \sim P(W|D)$.
- **Algorithm:**
  1. Get a new weight sample **proposal** $W_p \sim Q(W_p|W_{n-1})$ (Q is centered around previous W_{n-1})
  2. Compute Likelihoods $L(D|W_p)$ & $L(D|W_{n-1})$ **using the training data $\boldsymbol{D}$.**
  3. Compute the "new posterior" vs. "old posterior" **ratio**: $p = \frac{L(D|W_p)P(W_p)}{L(D|W_{n-1})P(W_{n-1})}$
  4. With Probability $\min(p, 1)$, **accept the proposal** $W_n \leftarrow W_p$ **Else** $W_n \leftarrow W_{n-1}$

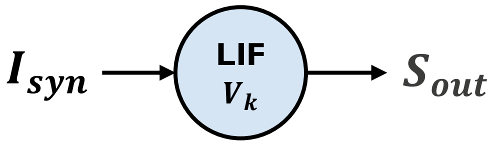


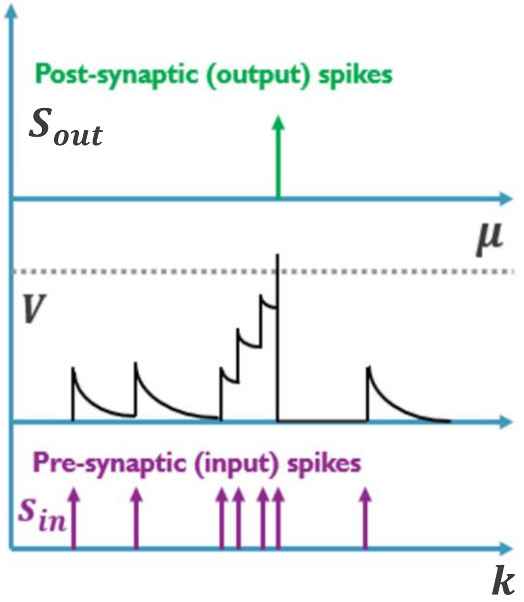
Metropolis-Hastings SNN Architecture

SNN Architecture *with LIF neuron non-ideality*

- SNN architecture using Leaky Integrate and Fire (LIF) Neurons:

$$\begin{cases} V_{k+1} = \alpha V_k + (1-\alpha) I_{syn} \\ S_{out} = 1 \textbf{ if } V_{k+1} \geq \mu \textbf{ else } 0 \\ V_{k+1} = 0 \textbf{ if } V_{k+1} \geq \mu \textbf{ or } \ V_{k+1} < 0 \end{cases}$$





Where α is the membrane decay and μ is the LIF threshold.

- We **simulate hardware non-ideality** by applying an **arbitrary hard non-linearity** on the neuron's *membrane potential*: $f_\sigma(V) = V + \frac{\sigma}{2}V^2 + \frac{\sigma}{6}V^3$
- During our experiments, we will explore the impact of non-linearity strength σ on SNN training with Metropolis-Hastings vs. Surrogate Gradient Descent backprop.

➢ In the **backprop** setup, the **non-ideality** is **not included** in the model, to *simulate the training of SNN hardware* **with incomplete knowledge** of the underlying SNN hardware model.

Metropolis-Hastings SNN training

- We set up a small SNN composed of two fully-connected layers (3 LIF → 2 LIF).
- The SNN is assessed *within a biomedical scenario* on the 2-class Wisconsin Breast Cancer detection dataset as Poisson spike trains of length $T = 10$ time steps.
- The **learning goal** is to steer the **output spike rates** $\bar{r}_{out}$ to the **one-hot label** $\bar{y}$.

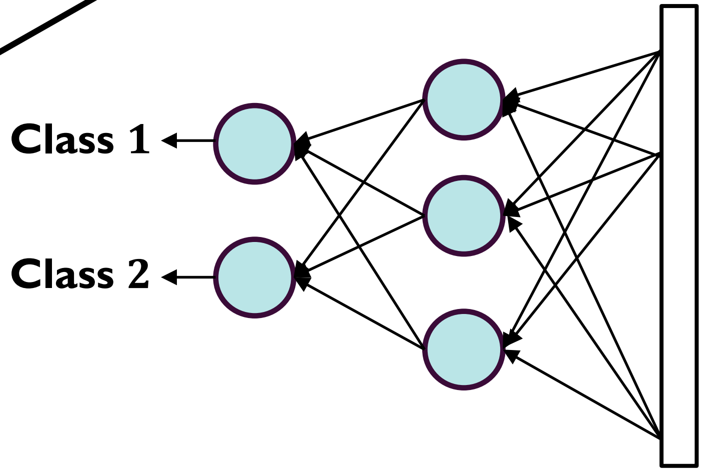
Target Label $\bar{y} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$$MSE = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} ||\bar{y}_i - \bar{r}_{out,i}||_2^2$$

Corresponding Likelihood

$$L(D|W) = \frac{1}{\sqrt{2\pi v}} e^{\frac{-MSE}{2v^2}}$$

$\bar{r}_{out} = \begin{bmatrix} r_{out}^1 \\ r_{out}^2 \end{bmatrix}$



Input Dimension 30

Algorithm 1 Metropolis-Hastings SNN Training

Input: D : training data and labels, c^2 : weight sampling variance, v^2 : Likelihood density variance in (5), p^2 : Prior density variance in (6).

$\Lambda \leftarrow c^2 I$ //diagonal covariance matrix with variance c^2
$W_1 \sim N(0, \Lambda)$
for $n = 1$ to end **do**
  $W_p \sim N(W_n, \Lambda)$ // Sample random proposal W_p
  Run SNN with weights W_p and W_n and data D to compute $L(D|W_p)$ and $L(D|W_n)$ (see Section II-A1)
  $\alpha(W_p|W_n) = \min\{1, \frac{L(D|W_p)P(W_p)}{L(D|W_n)P(W_n)}\}$
  With probability $\alpha(W_p|W_n)$ set $W_{n+1} = W_p$, **else** keep $W_{n+1} = W_n$
end for

Results

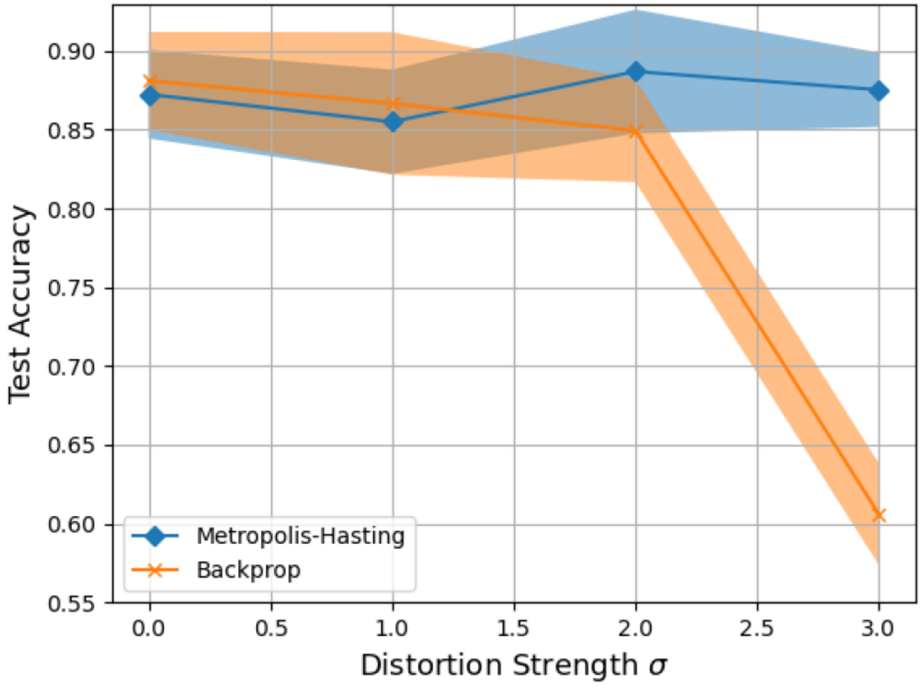
Experimental Setup

- The goals of our experiments are to study *how Metropolis-Hastings compares to Surrogate Gradient Descent* [Neftci et al. 2019, Eshraghian et al. 2023] for SNNs under:
  1. Varying **model non-ideality** σ.
  2. In terms of **data efficiency** and **SNN generalization** performance (i.e., how much training data is needed to achieve satisfactory test accuracy).
- We consistently follow a **5-fold train-test procedure** with different train-test splits and model initialization and report the **average accuracy and standard deviation**.
- As **LIF neuron parameters**, we arbitrary choose $\alpha = 0.9$ as the decay and $\mu = 1$ as the threshold.
- Metropolis-Hastings is run for 50000 steps with the first half being discarded as burn-out period.
- During our comparison study between Metropolis-Hastings and Backprop, we use a **Gaussian Surrogate Gradient** and the Adam optimizer with learning rate $\eta = 0.001$ for a total of 100 epochs with batch size 32.

$$S'_{out}(V) \approx \frac{1}{\sqrt{2\pi}} e^{-2V^2}$$

Varying the LIF non-ideality σ

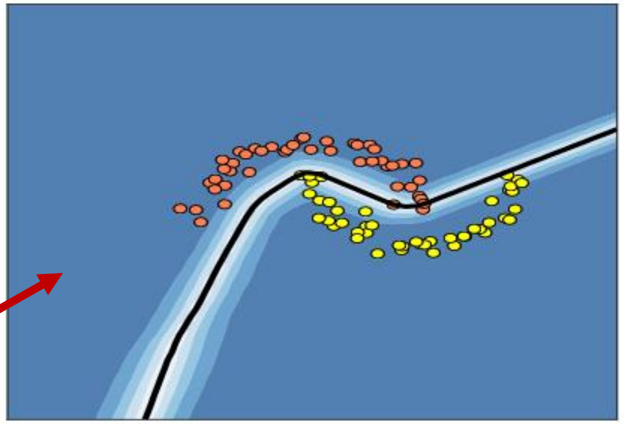
- 80%-20% Train-Test split (455 training sample, 114 test samples) using Wisconsin Breast Cancer dataset.
- As the LIF model non-ideality strength σ is increased, the *SNN test accuracy using Metropolis-Hastings* **stays within $\sim 87\%$** *while the Surrogate Gradient backprop SNN* **significantly drops for $\sigma = 3$.**
- Still, it is *remarkable to* see the **resilience** of **Surrogate Gradient backprop** for $\sigma \leq 2$.



Impact of the number of training data on SNN accuracy

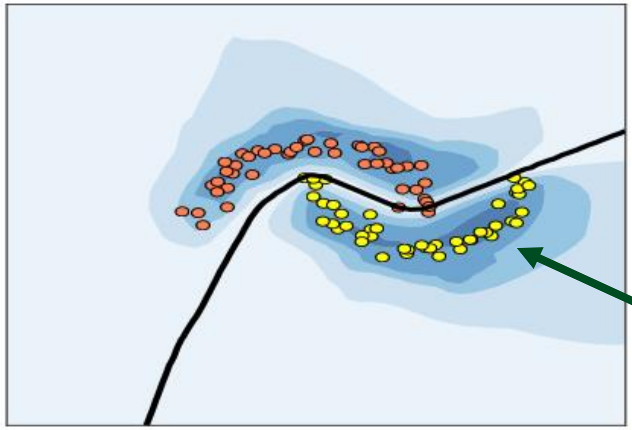
- An important aspect of Bayesian techniques is their potential for **better data efficiency**, <u>needing less training samples</u> for achieving usable accuracy [Jospin et al. 2022].
- This is because Metropolis-Hastings exactly samples the Posterior [Hastings 1970] and has **better control over model uncertainty**, *making models less over- and under-confident* [Kristiadi et al. 2020].
- This property is specially interesting for **reconfigurable** ultra-low-power edge AI SNN systems.
- E.g., in **personalized healthcare**, where the goal is to **deploy SNN models** in wearables **that can be personalized** to the **domain specificities of each patient**.
- ➢ Next, we study *how Metropolis-Hastings compares to Surrogate Gradient backprop* in term of **SNN generalization** and **training data efficiency**.

Non-Bayesian Approach: SGD-based backprop learning for MAP estimation of weights.



Overconfidence in out-of-sample region!

Bayesian Approach: E.g., using Metropolis-Hastings sampling for learning weights.

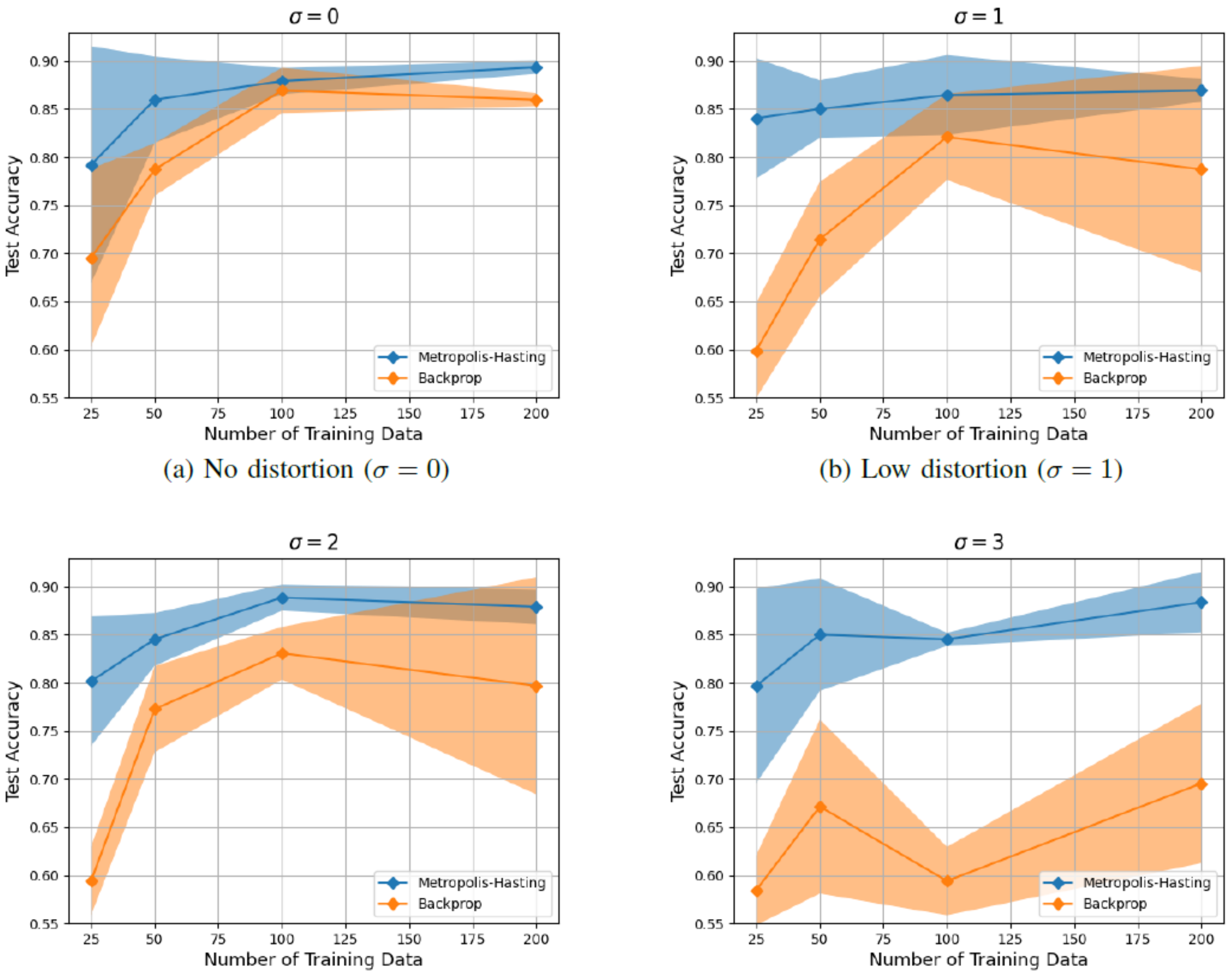


High confidence in *in-sample* region, low confidence in *out-of-sample* region!

[Kristiadi et al. 2020] A. Kristiadi, M. Hein, and P. Hennig. 2020. "Being Bayesian, even just a bit, fixes overconfidence in ReLU networks." In Proceedings of the 37th International Conference on Machine Learning (ICML'20).

Metropolis-Hastings leads to a better SNN generalization performance.

- We use a **small portion** N_t of the dataset as train set and use **all the remaining data** $N_{tot} - N_t$ as test set.
- We clearly see the **high data efficiency of Metropolis-Hastings** in the **SNN** context *vs. Surrogate Gradient.*
- This data efficiency holds across model non-ideality.
- This **confirms and extends** the *observations on data efficiency* done in the *non-spiking DNN context* to the **SNN context** [Depeweg et al. 2018].



(a) No distortion ($\sigma = 0$)

(b) Low distortion ($\sigma = 1$)

(c) Mild distortion ($\sigma = 2$)

(d) Strong distortion ($\sigma = 2$)

Conclusion

- This work has investigated the use of **Metropolis-Hastings Sampling** for **training SNNs** in a **model-free** fashion.
- Under strong LIF neuron non-ideality ($\sigma = 3$), the use of *Surrogate Gradient backprop* suffers from **large losses in accuracy** while Metropolis-Hastings is **not affected** thanks to its **model-free nature**.
- In addition, the use of **Metropolis-Hastings leads to better data efficiency** *and SNN generalization*, needing $> \mathbf{10} \times$ **less training data** for achieving usable ($\sim 90\%$) test accuracy.
- ➢ This makes Metropolis-Hastings interesting for **chip-in-the-loop training** of ultra-low-power SNNs using *less conventional technologies* such as **analog, memristive devices**, and so on.
- ➢ Metropolis-Hastings might also be **specially interesting** for applications where **embedded SNNs must be personalized** to each user, thanks to its remarkable **data efficiency**.
- ❑ As **future work**, we plan to study Bayesian training using more complex SNN architectures and exploring other Sampling methods such as Hamiltonian Monte Carlo methods.

This research was partially funded by a Long Stay Abroad grant from the Flemish Fund of Research - Fonds Wetenschappelijk Onderzoek (FWO) – grant V413023N.
This research received funding from the Flemish Government under the “Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen” programme.

References

References

- **[Indiveri et al. 2011]:** *Indiveri, G., Linares-Barranco, B., Hamilton, T., Schaik, A., Etienne-Cummings, R., Delbruck, T., Liu, S.C., Dudek, P., Häfliger, P., Renaud, S., Schemmel, J., Cauwenberghs, G., Arthur, J., Hynna, K., Folowosele, F., SAÏGHI, S., Serrano-Gotarredona, T., Wijekoon, J., Wang, Y., & Boahen, K. (2011). "Neuromorphic Silicon Neuron Circuits." Frontiers in Neuroscience, 5.*
- **[Payvand et al. 2022]:** *Payvand, M., Moro, F., Nomura, K. et al. Self-organization of an inhomogeneous memristive hardware for sequence learning. Nat Commun **13**, 5793 (2022).*
- **[Mitchell & Schuman, 2021]:** *J. Parker Mitchell and Catherine Schuman. 2021. "Low Power Hardware-In-The-Loop Neuromorphic Training Accelerator." In International Conference on Neuromorphic Systems 2021 (ICONS 2021). Association for Computing Machinery, New York, NY, USA.*
- **[Jospin et al. 2022]:** *L. V. Jospin, H. Laga, F. Boussaid, W. Buntine and M. Bennamoun, "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users," in IEEE Computational Intelligence Magazine, vol. 17, no. 2, pp. 29-48, May 2022.*
- **[Neftci et al. 2019]:** *E. O. Neftci, H. Mostafa and F. Zenke, "Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-Based Optimization to Spiking Neural Networks," in IEEE Signal Processing Magazine, vol. 36, no. 6, pp. 51-63, Nov. 2019.*
- **[Eshraghian et al. 2023]:** *J. K. Eshraghian et al., "Training Spiking Neural Networks Using Lessons From Deep Learning," in Proceedings of the IEEE, vol. 111, no. 9, pp. 1016-1054, Sept. 2023*
- **[Hastings 1970]:** *Hastings, W. K. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." Biometrika, vol. 57, no. 1, 1970, pp. 97–109.*
- **[Kristiadi et al. 2020]:** *A. Kristiadi, M. Hein, and P. Hennig. 2020. "Being Bayesian, even just a bit, fixes overconfidence in ReLU networks." In Proceedings of the 37th International Conference on Machine Learning (ICML'20).*
- **[Depeweg et al. 2018]:** *S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning," in Proceedings of the 35th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, vol. 80, 2018.*