Efficient GPU training of SNNs

Jamie Knight

UNIVERSITY OF SUSSEX

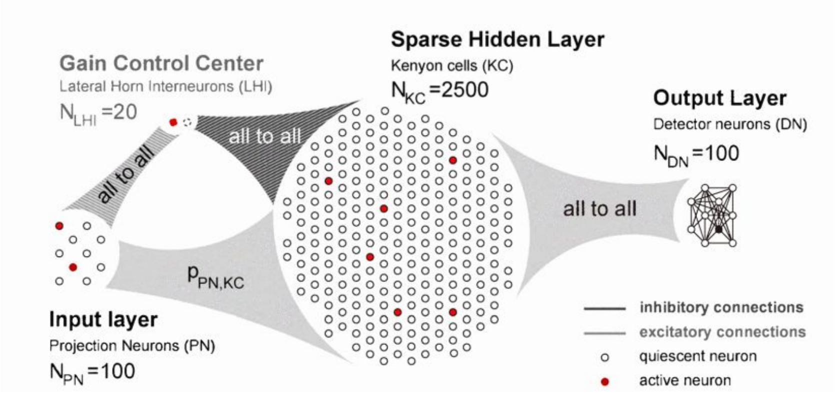
Engineering and Physical Sciences Research Council

SNNs for computational neuroscience

Where we’re coming from!

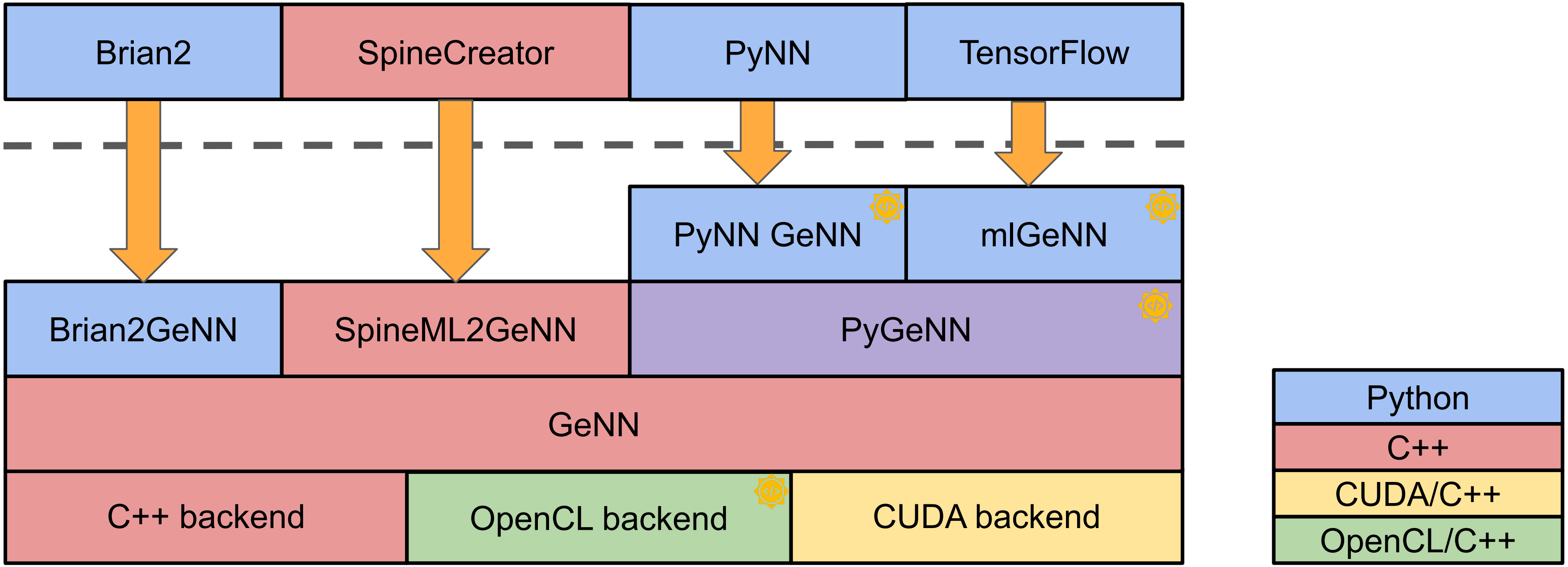
GPU-enhanced Neural Networks (GeNN) origin story

- 24x speed up over CPU
- It took a month to implement an existing model (after learning how to use CUDA)
- The program was optimised for a particular GPU
- It was designed for one size of the simulation



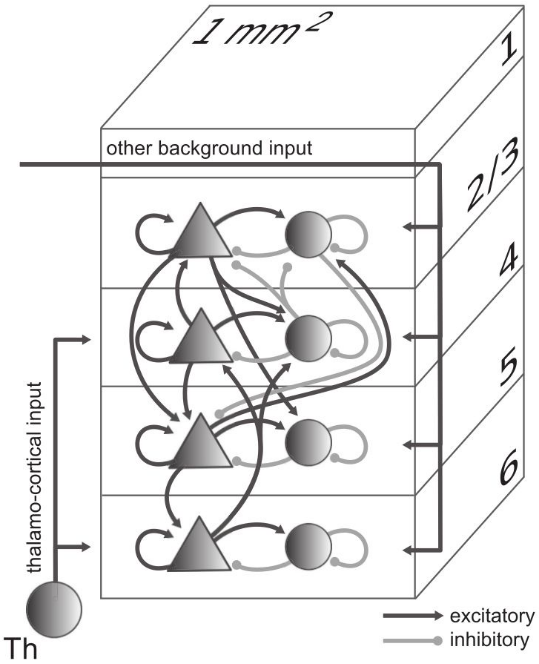
T. Nowotny, "Parallel implementation of a spiking neuronal network model of unsupervised olfactory learning on NVidia®CUDA™," The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, 2010, pp. 1-8.

GeNN - current status



https://github.com/genn-team/genn/

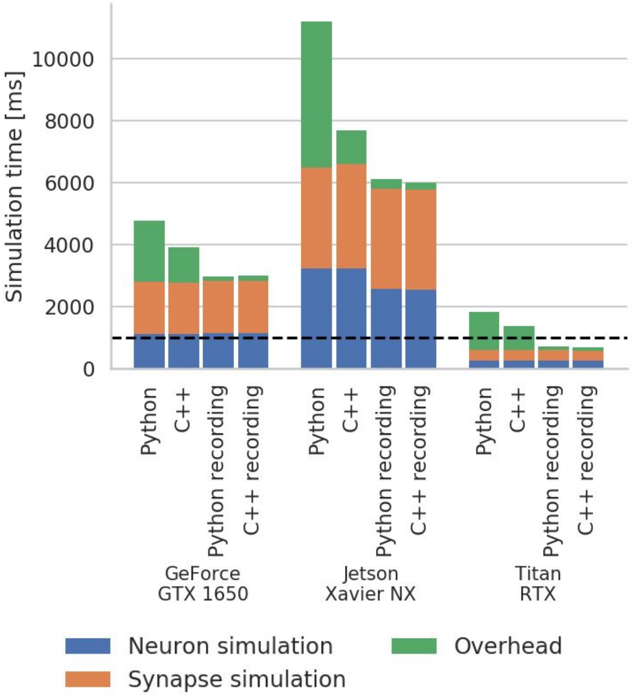
Cortical microcircuit simulations: 1mm^3 of brain



- 1mm^3 of cortex
  - Neural recording + microscopy + heuristics gave estimates of connectivity densities between cell types
  - Calcium imaging gave firing rates of cell types in awake animals
- 80×10^3 neurons
- 0.3×10^9 sparse synapses
- 3 seconds/second on CPU-based HPC

Potjans, T. C., & Diesmann, M. (2012). The Cell-Type Specific Cortical Microcircuit: Relating Structure and Activity in a Full-Scale Spiking Network Model. Cerebral Cortex

Cortical microcircuit simulations: 1mm^3 of brain



- On workstation GPU takes < 0.7 second/second
- Uses 10× less energy than CPU-based HPC (energy = power × time)

Knight, J. C., & Nowotny, T. (2018). GPUs Outperform Current HPC and Neuromorphic Solutions in Terms of Speed and Energy When Simulating a Highly-Connected Cortical Model. Frontiers in Neuroscience, 12(December), 1–19. https://doi.org/10.3389/fnins.2018.00941

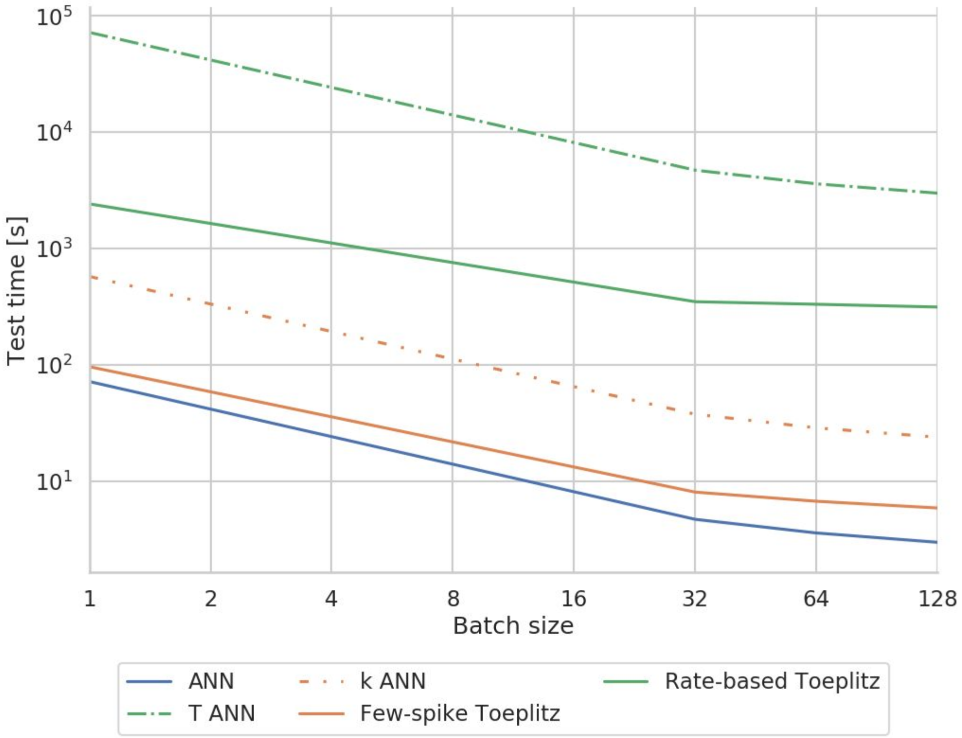
SNNs for machine learning

Where we’re going!

Spiking Neural Networks for ML

- Repeated claims in the literature that SNNs can save energy vs standard ANNs
- Brain-like sparse activity and connectivity **should** reduce computation and thus energy
- Very few actual demonstrations of this on **real tasks** and **standard hardware**
- Using GeNN we want to change this!

Converting ANNs to SNNs: CIFAR10



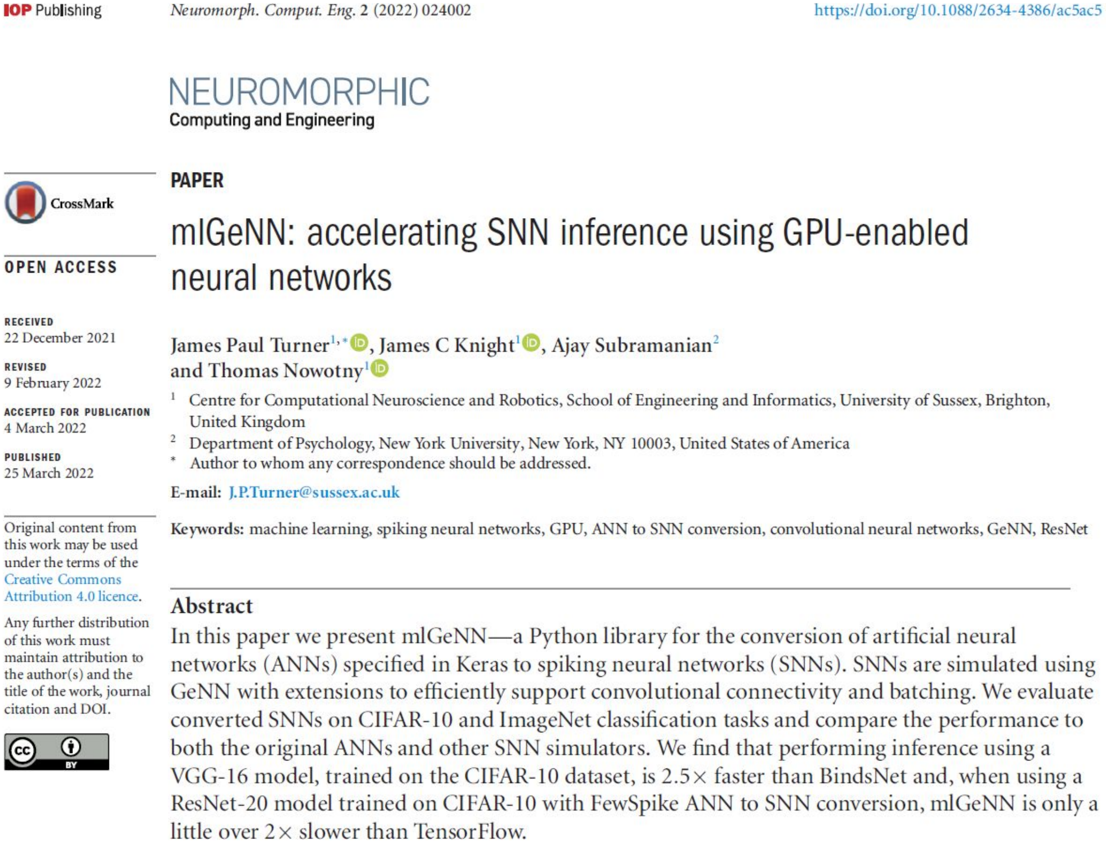
- No significant drop in performance
- ≈Half speed of TensorFlow ANN
- ≈10✕ faster than other solutions using ANN tools for SNN

Stöckl, C., & Maass, W. (2021). Optimized spiking neurons can classify images with high accuracy through temporal coding with two spikes. Nature Machine Intelligence, 3(3), 230–238. https://doi.org/10.1038/s42256-021-00311-4

Converting ANNs to SNNs: ImageNet

- 100 times slower than TensorFlow
- Efficiency savings due to sparsity need to counteract:
    - 10 SNN timesteps need to be simulated for each ANN iteration
    - Lower algorithmic complexity of ANN convolutions
    - Highly optimised TensorFlow code
- Deep, feedforward SNN architectures are not the answer!

Converting ANNs to SNNs: Paper



IOP Publishing Neuromorph. Comput. Eng. 2 (2022) 024002 https://doi.org/10.1088/2634-4386/ac5ac5

NEUROMORPHIC
Computing and Engineering

CrossMark

OPEN ACCESS

RECEIVED
22 December 2021

REVISED
9 February 2022

ACCEPTED FOR PUBLICATION
4 March 2022

PUBLISHED
25 March 2022

PAPER

mlGeNN: accelerating SNN inference using GPU-enabled neural networks

James Paul Turner[1,*], James C Knight[1], Ajay Subramanian[2] and Thomas Nowotny[1]

[1] Centre for Computational Neuroscience and Robotics, School of Engineering and Informatics, University of Sussex, Brighton, United Kingdom
[2] Department of Psychology, New York University, New York, NY 10003, United States of America
[*] Author to whom any correspondence should be addressed.

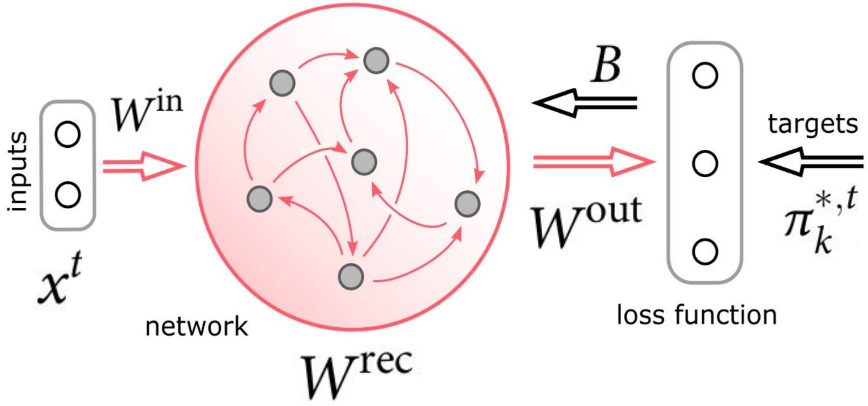
E-mail: J.P.Turner@sussex.ac.uk

Keywords: machine learning, spiking neural networks, GPU, ANN to SNN conversion, convolutional neural networks, GeNN, ResNet

Abstract

In this paper we present mlGeNN—a Python library for the conversion of artificial neural networks (ANNs) specified in Keras to spiking neural networks (SNNs). SNNs are simulated using GeNN with extensions to efficiently support convolutional connectivity and batching. We evaluate converted SNNs on CIFAR-10 and ImageNet classification tasks and compare the performance to both the original ANNs and other SNN simulators. We find that performing inference using a VGG-16 model, trained on the CIFAR-10 dataset, is 2.5× faster than BindsNet and, when using a ResNet-20 model trained on CIFAR-10 with FewSpike ANN to SNN conversion, mlGeNN is only a little over 2× slower than TensorFlow.

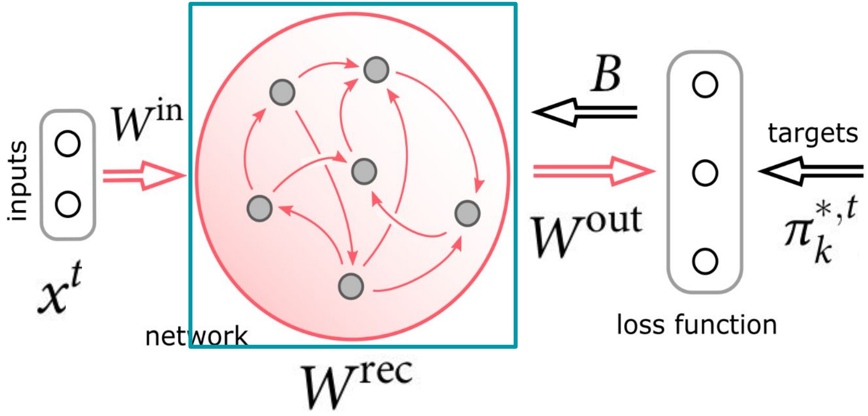
Training recurrent SNNs: model



Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: model

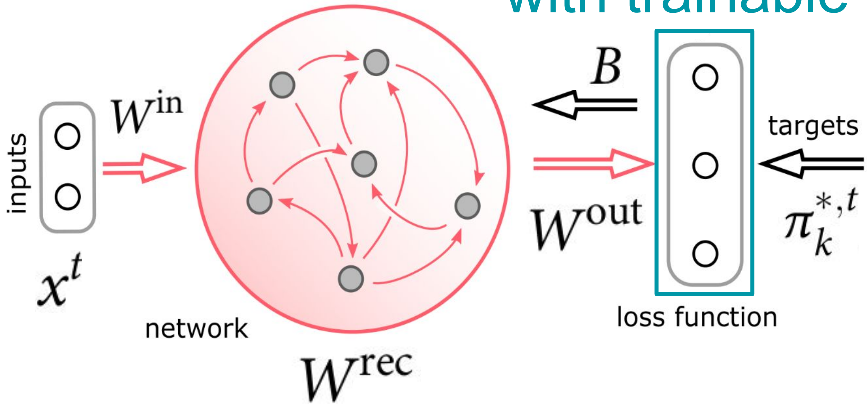
LIF neuron with adaptation and relative reset



Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: model

Non-spiking
output neuron
with trainable bias



Softmax π_k^t calculated
from membrane voltage

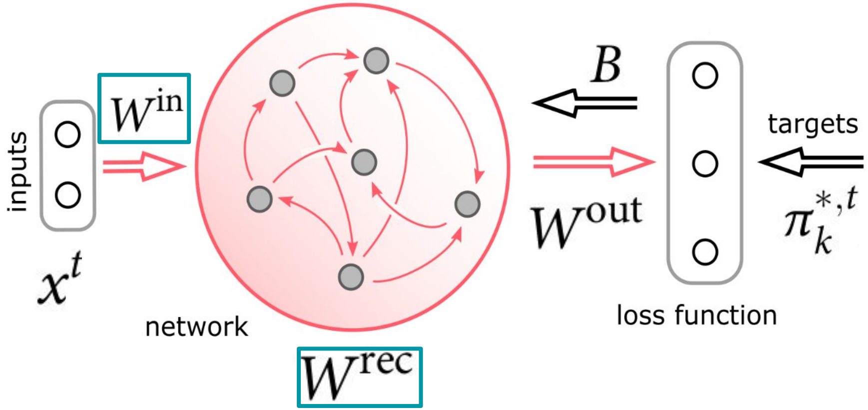
Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: eProp

Per-synapse eligibility traces and supervised learning rule

$$\epsilon_{ji,a}^{t+1} = \psi_j^t \bar{z}_i^{t-1} + (\rho - \psi_j^t \beta)\epsilon_{ji,a}^t \quad e_{ji}^t = \psi_j^t \left(\bar{z}_i^{t-1} - \beta \epsilon_{ji,a}^t\right)$$

$$\Delta W_{ji}^{\mathrm{rec}} = -\eta \sum_t \underbrace{\left(\sum\nolimits_k B_{jk}(\pi_k^t - \pi_k^{*,t})\right)}_{=L_j^t} \bar{e}_{ji}^t.$$



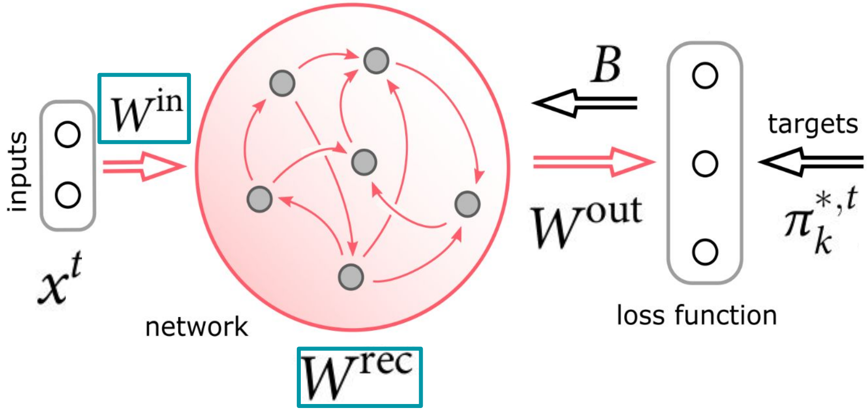
Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: eProp

Per-synapse eligibility traces and supervised learning rule

$$\epsilon_{ji,a}^{t+1} = \boxed{\psi_j^t}\bar{z}_i^{t-1} + (\rho - \boxed{\psi_j^t}\beta)\epsilon_{ji,a}^t \qquad e_{ji}^t = \boxed{\psi_j^t}\left(\bar{z}_i^{t-1} - \beta\epsilon_{ji,a}^t\right)$$

$$\Delta W_{ji}^{\mathrm{rec}} = -\eta \sum_t \underbrace{\left(\sum_k B_{jk}(\pi_k^t - \pi_k^{*,t})\right)}_{=L_j^t} \bar{e}_{ji}^t.$$



Postsynaptic neuron surrogate gradient

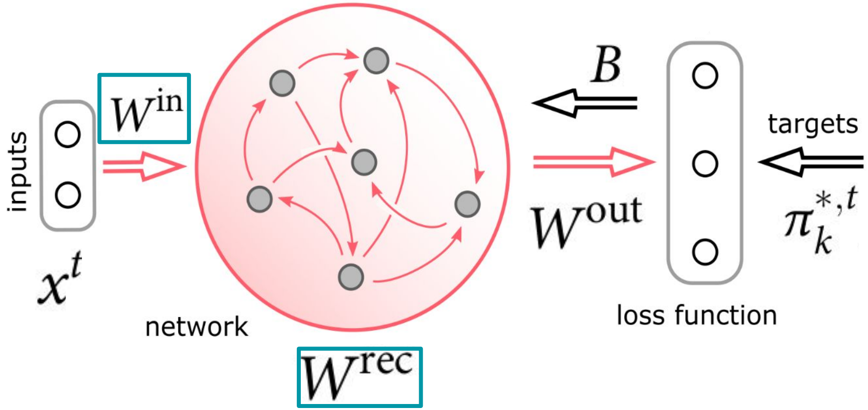
Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: eProp

Per-synapse eligibility traces and supervised learning rule

$$\epsilon_{ji,a}^{t+1} = \psi_j^t \boxed{\bar{z}_i^{t-1}} + (\rho - \psi_j^t \beta)\epsilon_{ji,a}^t \qquad e_{ji}^t = \psi_j^t \left(\boxed{\bar{z}_i^{t-1}} - \beta \epsilon_{ji,a}^t \right)$$

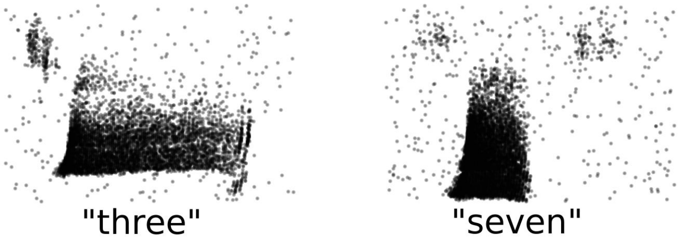
$$\Delta W_{ji}^{\mathrm{rec}} = -\eta \sum_t \underbrace{\left(\sum\nolimits_k B_{jk} (\pi_k^t - \pi_k^{*,t}) \right)}_{=L_j^t} \bar{e}_{ji}^t .$$

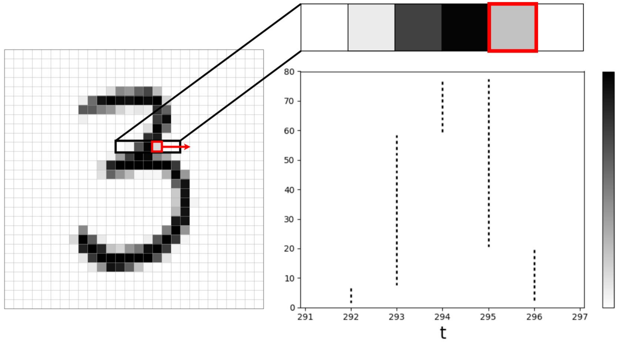


Filtered presynaptic activity

Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, *11*(1), 3625. https://doi.org/10.1038/s41467-020-17236-y

Training recurrent SNNs: datasets





Spiking Heidelberg Digits

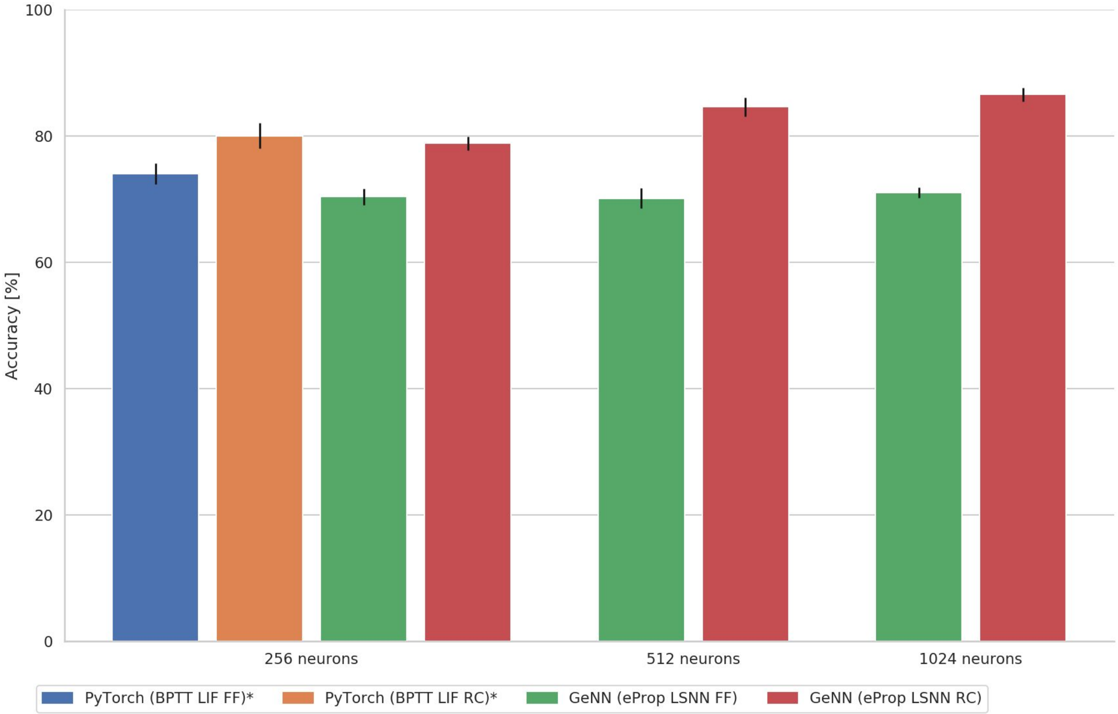
- English & German spoken digits
- 700 spike trains

Spiking Sequential MNIST

- Pixels ‘scanned’ in fixed order
- 79 neurons representing intensity thresholds

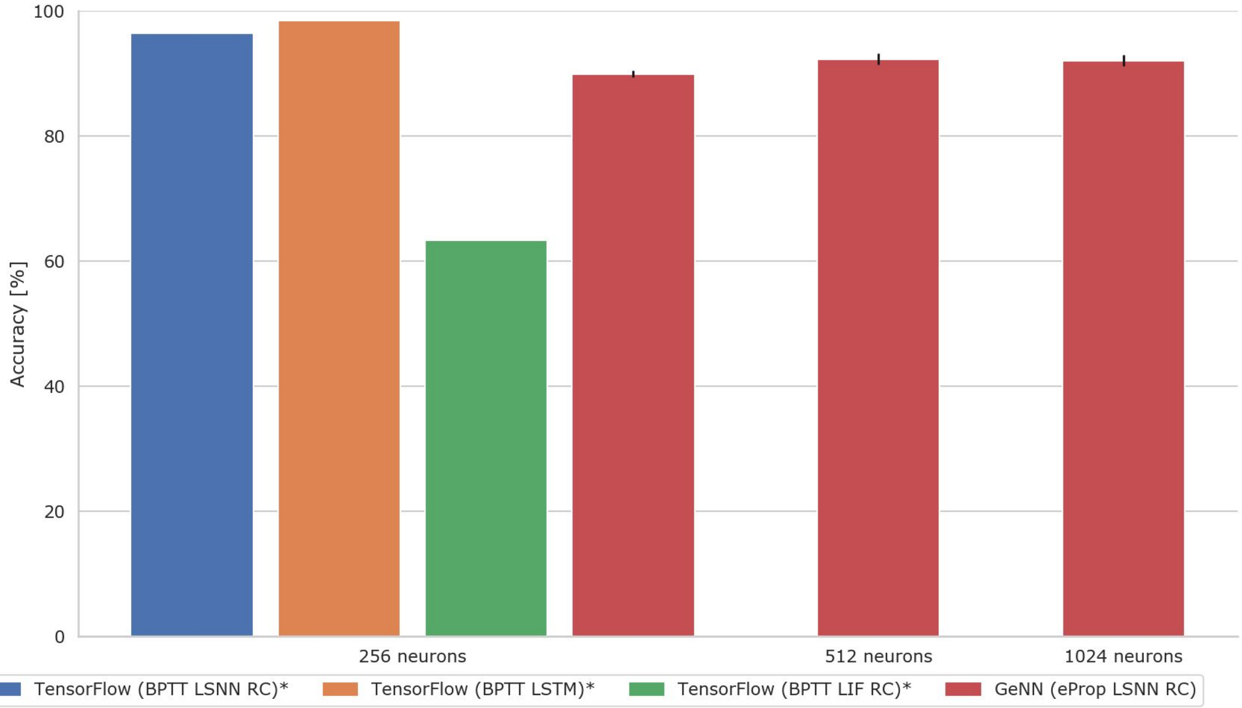
Cramer, B., Stradmann, Y., Schemmel, J., & Zenke, F. (2020). The Heidelberg Spiking Data Sets for the Systematic Evaluation of Spiking Neural Networks. IEEE Transactions on Neural Networks and Learning Systems. https://doi.org/10.1109/TNNLS.2020.3044364

Training recurrent SNNs: SHD accuracy



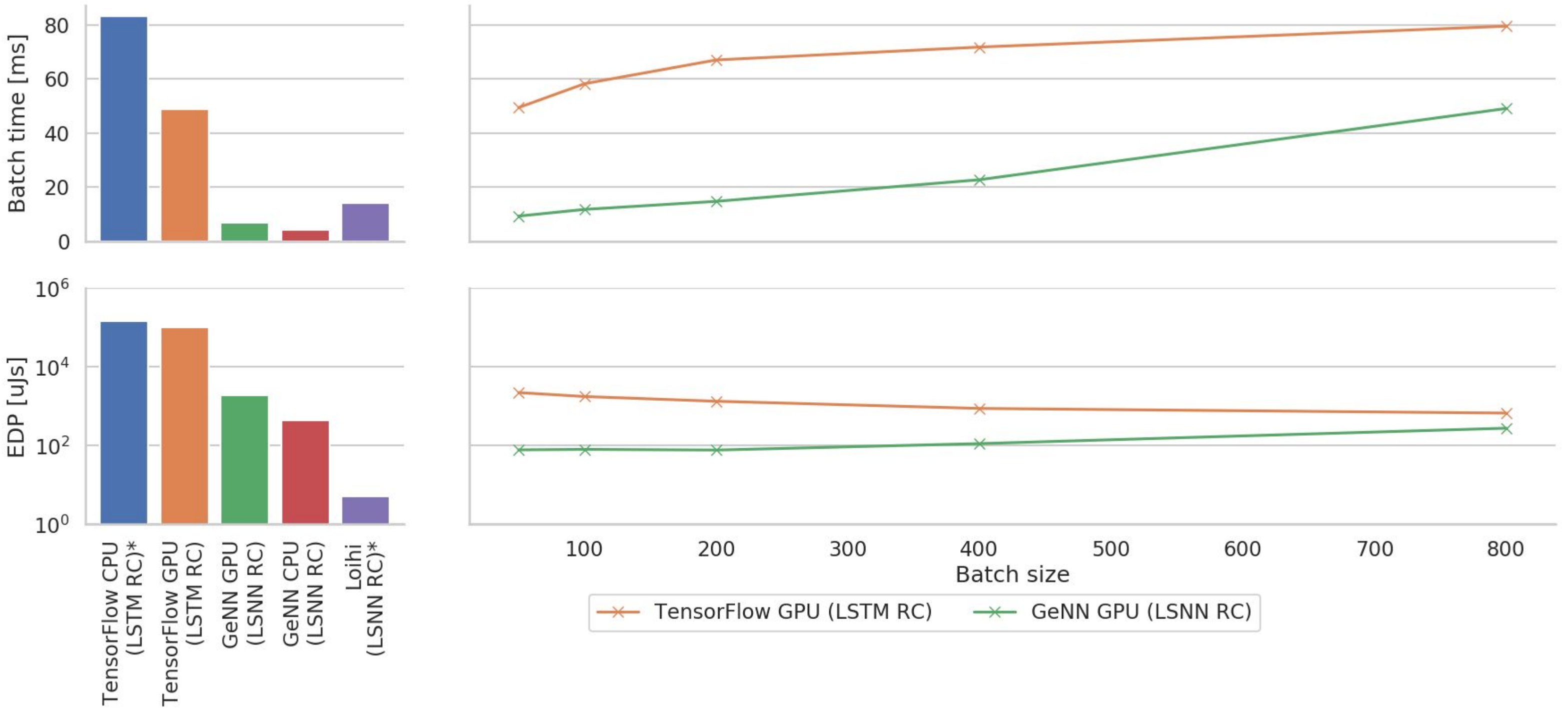
* Zenke, F., & Vogels, T. P. (2020). The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *BioRxiv*, 1–22. https://doi.org/10.1101/2020.06.29.176925

Training recurrent SNNs: SMNIST accuracy



* Plank, P., Rao, A., Wild, A., & Maass, W. (2021). A Long Short-Term Memory for AI Applications in Spike-based Neuromorphic Hardware. Retrieved from http://arxiv.org/abs/2107.03992

Training recurrent SNNs: Performance



* Plank, P., Rao, A., Wild, A., & Maass, W. (2021). A Long Short-Term Memory for AI Applications in Spike-based Neuromorphic Hardware. Retrieved from http://arxiv.org/abs/2107.03992

Training recurrent SNNs: Paper

Efficient GPU training of LSNNs using eProp

James C Knight
J.C.Knight@sussex.ac.uk
University of Sussex
School of Engineering and Informatics
Brighton, United Kingdom

Thomas Nowotny
T.Nowotny@sussex.ac.uk
University of Sussex
School of Engineering and Informatics
Brighton, United Kingdom

ABSTRACT

Taking inspiration from machine learning libraries – where techniques such as parallel batch training minimise latency and maximise GPU occupancy – as well as our previous research on efficiently simulating Spiking Neural Networks (SNNs) on GPUs for computational neuroscience, we have extended our GeNN SNN simulator to enable spike-based machine learning research on general purpose hardware. We demonstrate that SNN classifiers implemented using GeNN and trained using the eProp learning rule can provide comparable performance to those trained using Back Propagation Through Time and show that the latency and energy usage of our SNN classifiers is up to 7× lower than an LSTM running on the same GPU hardware.

CCS CONCEPTS

• **Computing methodologies** → *Bio-inspired approaches*; *Supervised learning*; *Vector / streaming algorithms.*

KEYWORDS

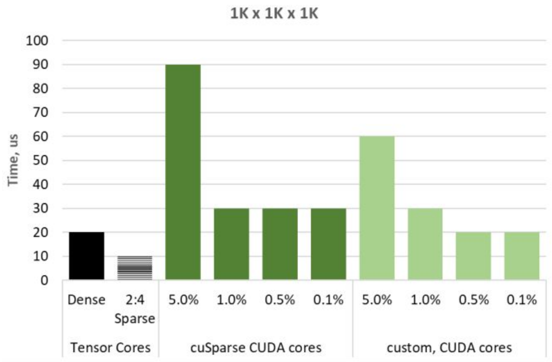
spiking neural networks, efficient simulation, GPU

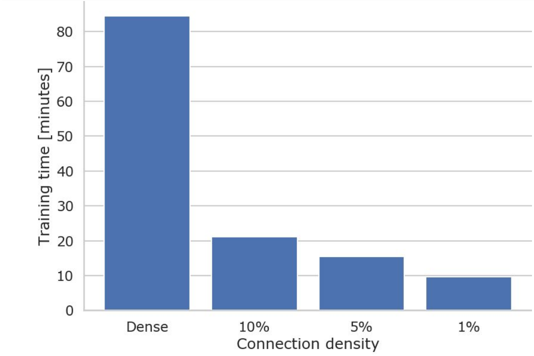
ACM Reference Format:
James C Knight and Thomas Nowotny. 2022. Efficient GPU training of LSNNs using eProp. In *Neuro-Inspired Computational Elements Conference (NICE 2022), March 28-April 1, 2022, Virtual Event, USA*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3517343.3517346

RTRL [21] is an alternative 'forward mode' algorithm for training RNNs but, in its general form, it is too computationally expensive to be practical. However, if the gradients flowing through the 'explicit' recurrent connections are ignored and only those flowing through the 'implicit' recurrence represented by the dynamics of individual neurons are considered, much more computationally tractable learning rules can be derived [26]. Learning rules of this sort include SuperSpike [25], eProp [4] and Decolle [15]. However, in order to apply these new spike-based machine learning techniques to larger models and data-sets as well as prototyping algorithms for neuromorphic hardware [8, 11, 18], new tools are required which can efficiently simulate SNNs on existing hardware. The development of efficient SNN simulators has been a key area of computational neuroscience research for several decades [1, 6, 12, 13, 23] but, these simulators are not well-suited to the types of model and the workflows required for spike-based machine learning research. As such, many ML researchers have chosen to build libraries [9, 10, 14, 19] on top of more familiar tools such as PyTorch. However, while libraries like PyTorch are highly-optimised for rate-based models, they does not take advantage of the spatio-temporal sparsity of SNNs which have the potential to enable massive computational savings over rate-based networks [24].

While our GeNN simulator [16, 17, 23] was originally developed for Computational Neuroscience research, its longstanding focus on flexibility and its targeting of GPU accelerators has made it easily adaptable to the needs of spike-based ML. Specifically, we have

Training recurrent SNNs: Sparse connectivity



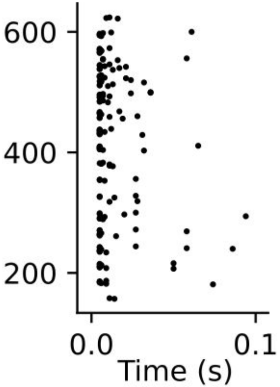


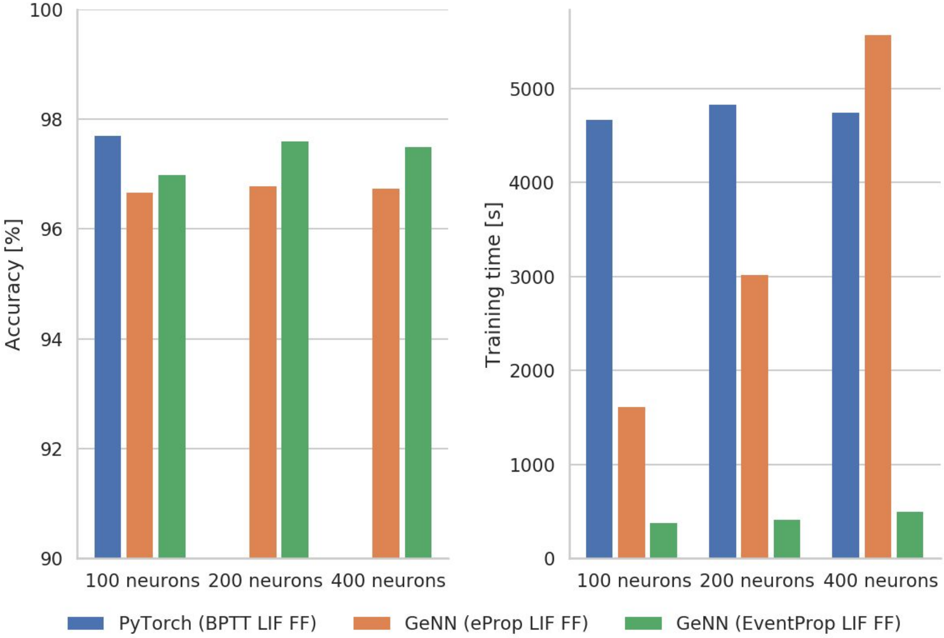
- Basic ANN operation is matrix multiply
  - Dense - very fast
  - Sparse - tricky to optimise
- Spike transmission (is not multiplication)
  - Dense - can't match ANN
  - Sparse - much easier to optimise!

Micikevicius , P. (2021). Sparsity, structure and performance. Sparsity in Neural Networks Workshop

Fully event-driven learning

$$T(x) = \begin{cases} \tau_{\text{eff}} \log\left(\frac{x}{x-\vartheta}\right) & x > \vartheta \\ \infty & \text{otherwise} \end{cases}$$





Wunderlich, T. C., & Pehle, C. (2021). Event-based backpropagation can compute exact gradients for spiking neural networks. Scientific Reports, 11(1), 12829. https://doi.org/10.1038/s41598-021-91786-z

Zenke, F., & Vogels, T. P. (2020). The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. BioRxiv, 1–22. https://doi.org/10.1101/2020.06.29.176925

Future direction

- **Some** supervised learning in brain but unlikely to be end-to-end
  - Self-supervised learning
  - Contrastive learning
- Structural plasticity to optimise sparse connectivity
- FPGA hardware

Thank you!

J.C.Knight@sussex.ac.uk