Experiments on BrainScaleS

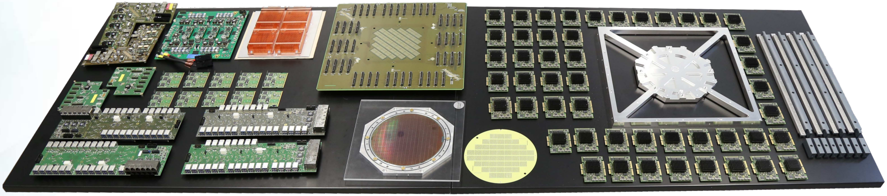
Electronic Vision(s)
Kirchhoff-Institute for Physics
Heidelberg University

2018-03-01

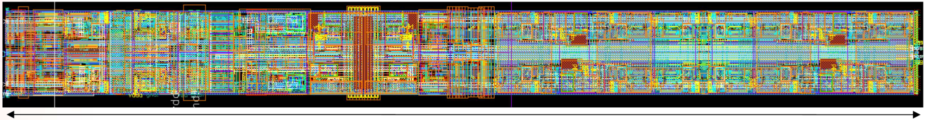
Machine Room with 20 BrainScaleS Wafer Modules

BrainScaleS Wafer Module

- 20 cm Wafer, 180 nm CMOS
- Main PCB
- 48 Kintex-7 FPGAs (TU Dresden)
- Power supplies
- Aux. Boards (e.g. for analog readout)
- In development for > 10 years

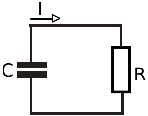


Analog Neuron Circuit



layout drawing of two neurons: 150x20 µm^2

- Adaptive Exponential Integrate and Fire (AdEx) Model
- Dedicated circuits in every neuron for:
  - resting potential
  - reset potential
  - threshold potential
  - reversal potentials
  - refractory period
  - membrane time constant
  - synaptic time constants
  - adaption
  - exponential term
- Accelerated dynamics compared to biological real-time:

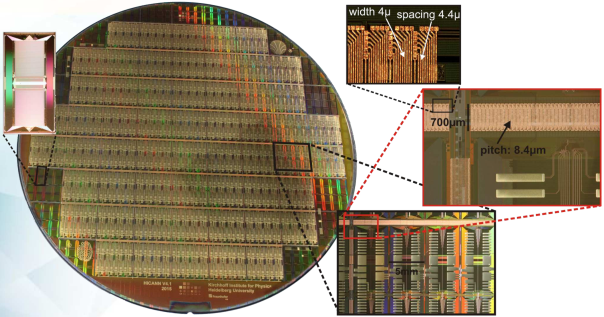


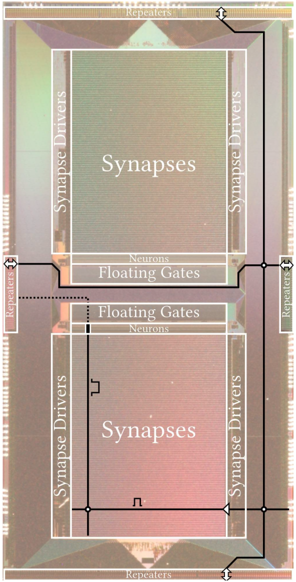
$\tau = C \cdot R, \frac{\tau_{\text{hw}}}{\tau_{\text{bio}}} = 10^{-4}$

$\tau_{\text{hw}} = 1\,\mu\text{s} \Rightarrow \tau_{\text{bio}} = 10\,\text{ms}$

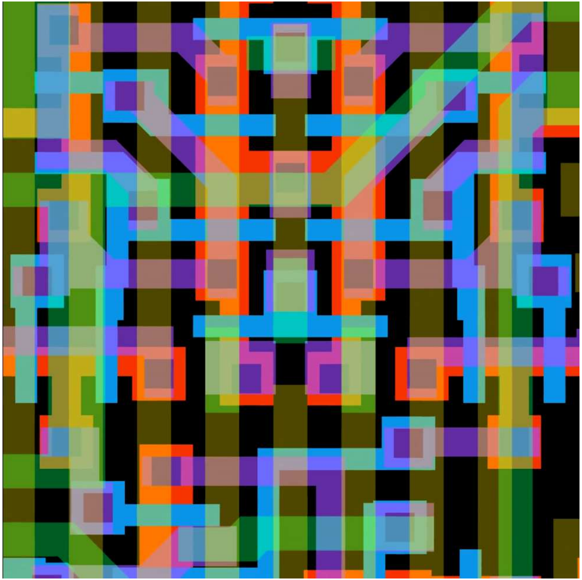
HICANN: High Input Count Analog Neural Network Chip

- 512 analog neurons, 110 000 plastic synapses
- Digital communication $\rightarrow$ mixed-signal system
- Sparse crossbar switches connecting busses $\rightarrow$ programmable network connections
- Analog parameter storage (floating gates)
- Postprocessing (IZM Berlin) $\rightarrow$ wafer scale networks





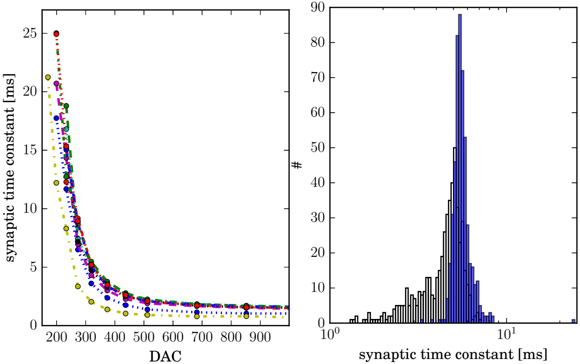
From Transistors to Wafer



Configuration and Calibration

Configuration:

- Large configuration space per HICANN: > 10000 parameters
- Floating gate analog storage:
  - Voltages (max. 1800 mV)
  - Currents (max. 2500 nA)
- Both programmed via a DAC (0...1023)



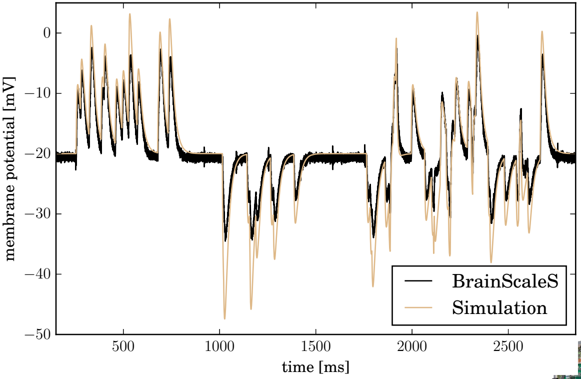
Calibration:

- Analog circuits are subject to process dependent device mismatch, i.e. variations from one transistor to the other
- For same value of supplied parameter, the neuron response varies
- For all neurons and parameters: set DAC values, measure and fit

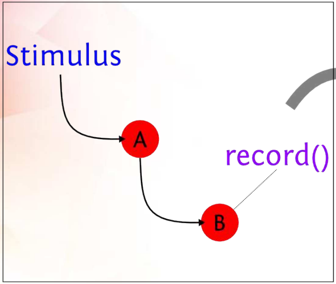
pyNN: The Network Description Language

- PyNN is a simulator independent network description language
- Programs can be executed on different simulators and different hardware without (large) changes

```
import pyhmf as pynn
# import pyNN.nest as pynn

stimulus = pynn.Population(1,
pynn.SpikeSourceArray, {
'spike_times': exc_spike_times})

pop = pynn.Population(1,
pynn.IF_cond_exp,
neuron_parameters)

pynn.Projection(stimulus,
pop, con, target='excitatory')
```

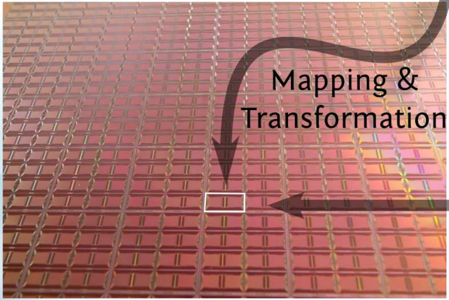


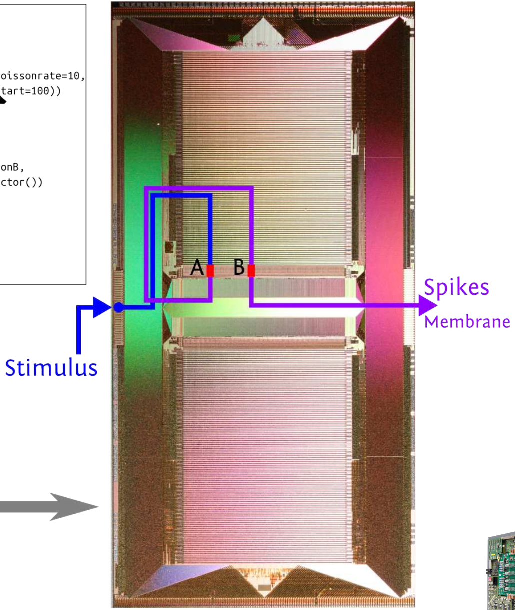
Mapping a Network to Hardware



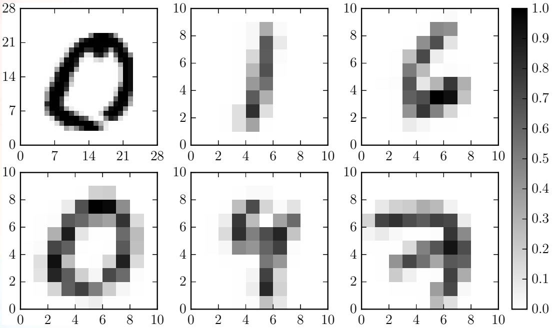
PyNN

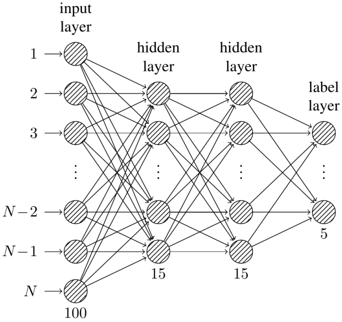
```
import pyhmf as sim

sim.setup()

stimulus = Population(1, SpikeSourcePoissonrate=10,
                      duration=100, start=100))

neuronA   = Population(1, IF_cond_exp)
neuronB   = Population(1, IF_cond_exp)

projection = Projection(neuronA, neuronB,
                        AllToAllConnector())


neuronB.record()
neuronB.record_v()

sim.run(1000)
sim.end()
```





MNIST Handwritten Digit Recognition with a Deep Neural Network

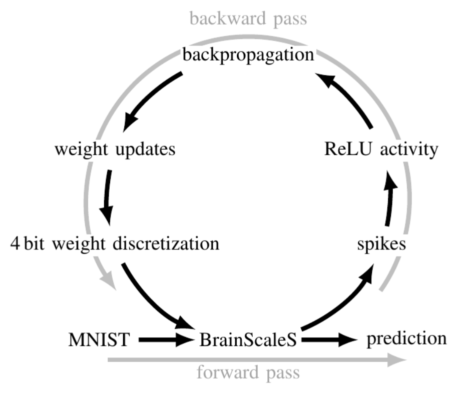


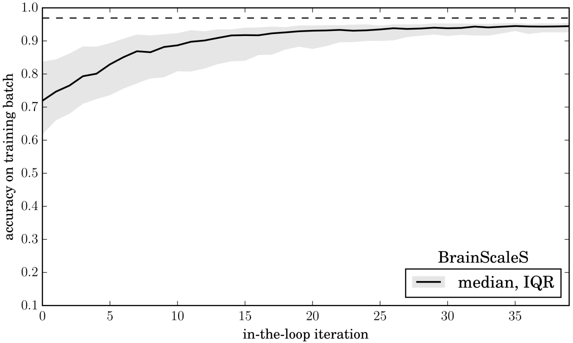


- Fully-connected feed forward network
- $100 + 15 + 15 + 5 = 135$ neurons, 3700 synapses

(Schmitt and Klaehn et al., 2017)

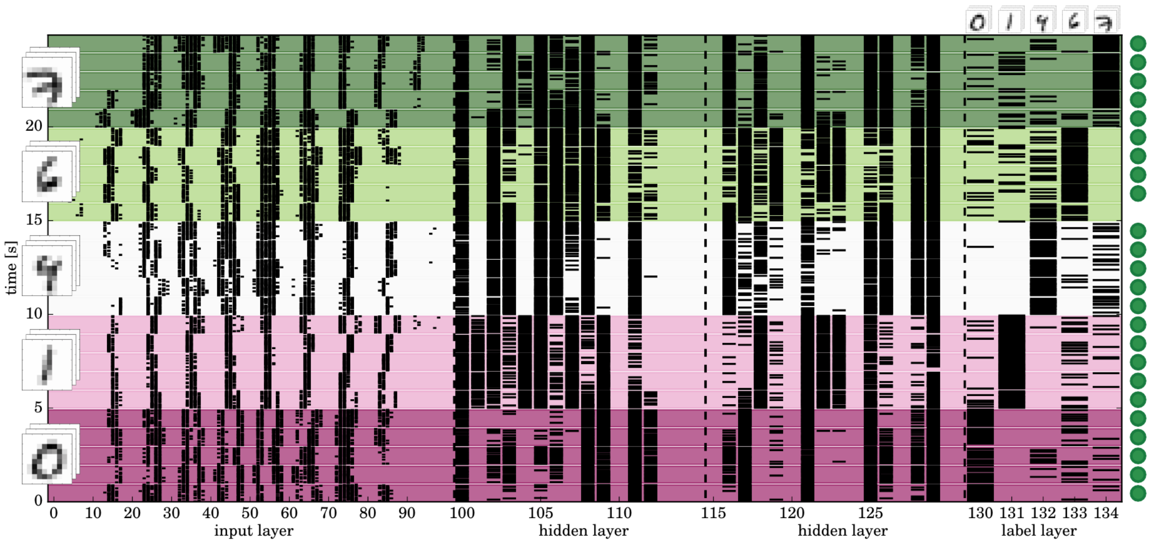
In-The-Loop Training



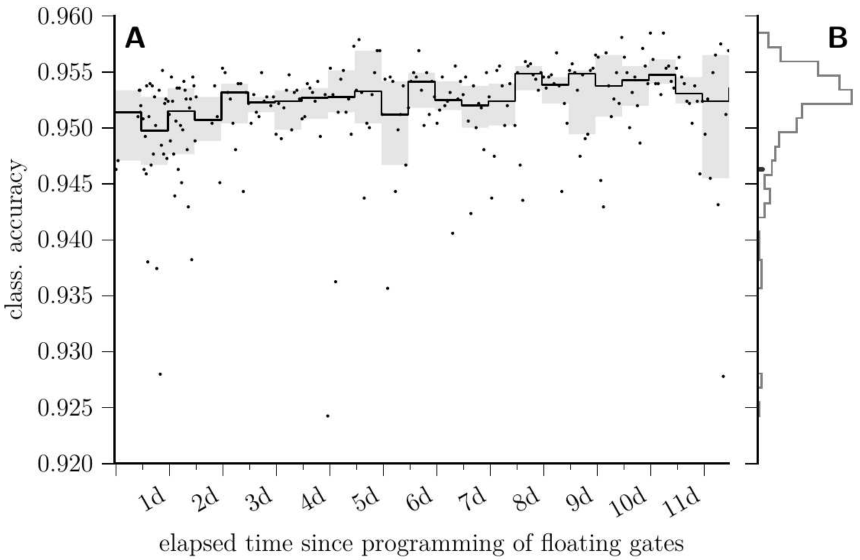


- Conversion to spiking neurons: accuracy reduced to $72^{+12}_{-10}\,\%$
- Continue training with the hardware in the loop: accuracy recovered to $95^{+1}_{-2}\,\%$

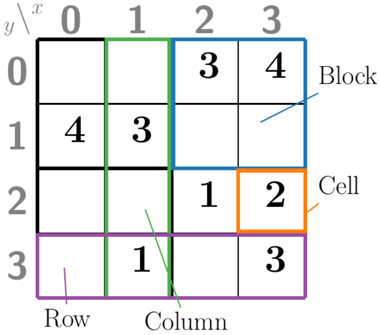
Raster Plot after In-The-Loop Training

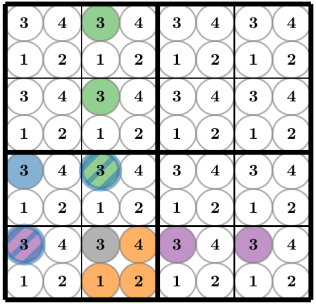


Floating Gate Time Stability



Solving the Constraint Satisfaction Problem Sudoku

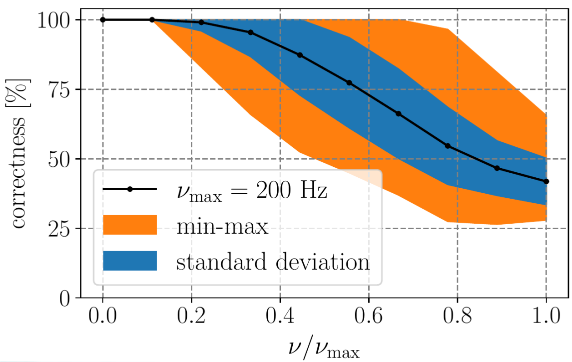


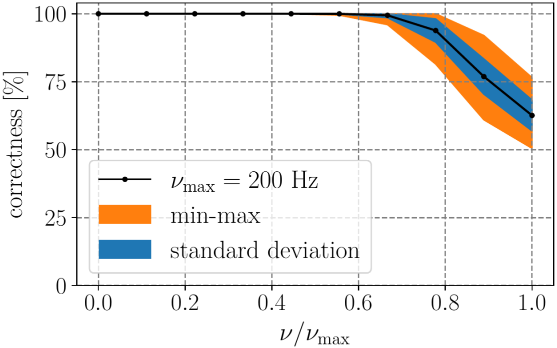


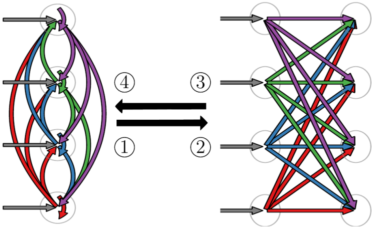
- Winner-take-all structure represent Sudoku rules (Jonke et al., 2016), (Guerra et al., 2017)
- Minimally represent each cell with 4 neurons $\rightarrow 4 \times 4 \times 4 = 64$ neurons, 1000 synapses

(A. Kugele, master thesis, 2018)

Training the WTA motifs (left: before, right: after training)



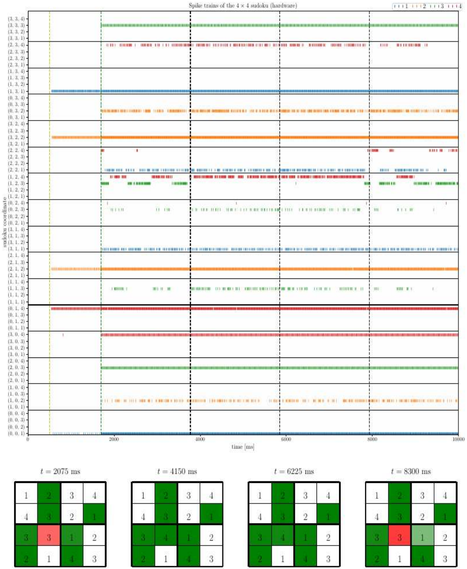




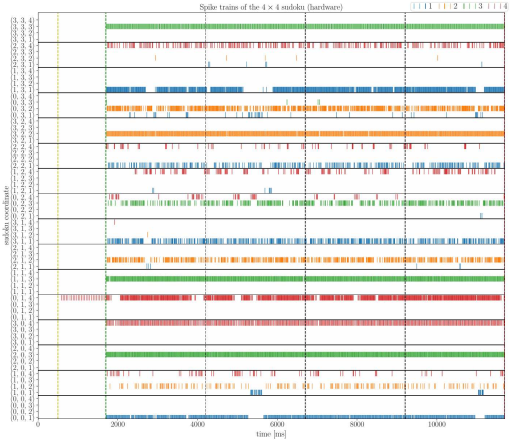
1. Set input rates, measure output rates
2. Plug both rates in the unfolded network
3. Calculate a new weight matrix
4. Use new weight matrix for the training

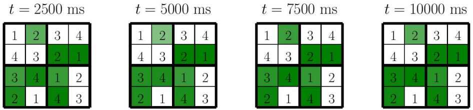
Repeat steps until convergence

Before Training

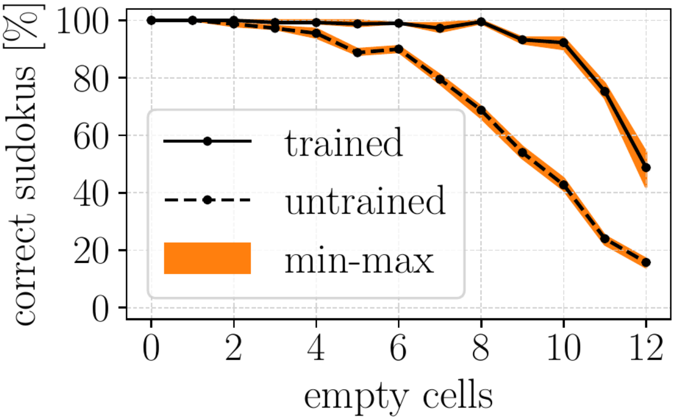


After Training



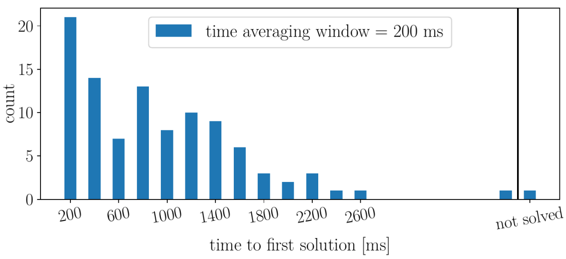


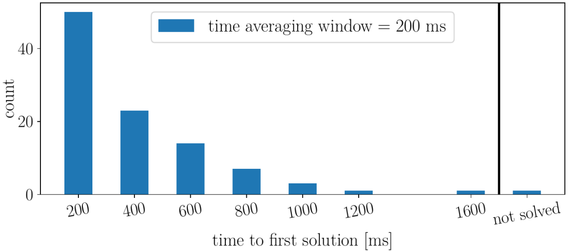
Performance on Solving Sudokus



- Increased performance after training
- Sudokus with 8 (of 16) empty cells safely solved

Time to First Solution (left: untrained, right: trained)





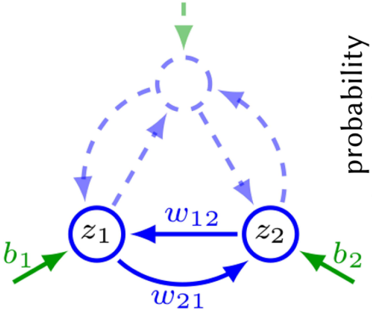
- Time to first solution is greatly decreased after training

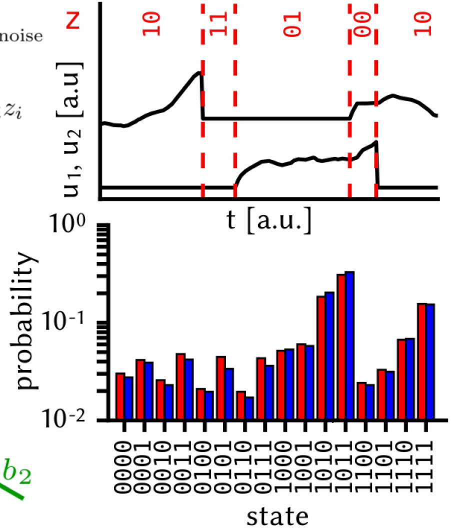
Neural Sampling (Buesing et al., 2011), (Petrovici et al., 2016)

$$\frac{du}{dt} C_m = g_l \left(E_l - u\right) + I_{\text{ext}} + I_{\text{noise}}$$

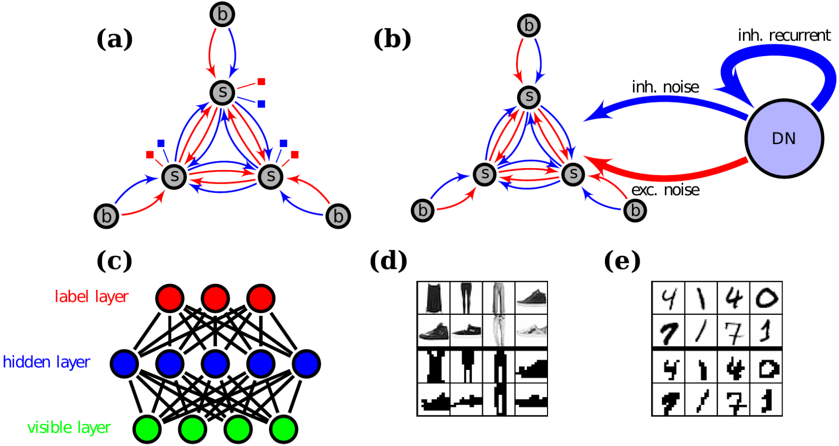
$$E\left(\vec{z}\right) = \sum_{i,j} W_{ij} z_i z_j + \sum_i b_i z_i$$

$$p\left(\vec{z}\right) \propto e^{-E(\vec{z})}$$



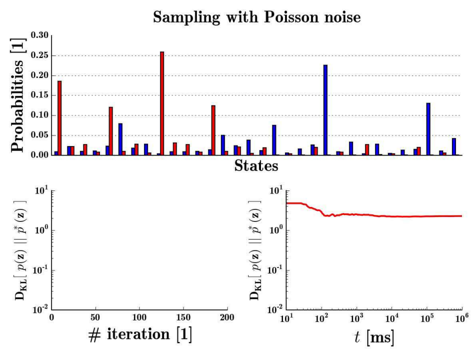


Sampling from Restricted Boltzmann Machines



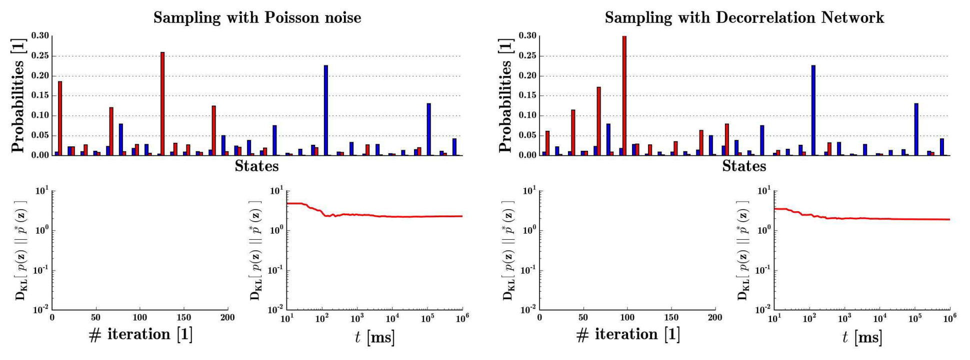
- Stochasticty supplied by a kind of Sea-of-Noise network (Jordon et al., 2017)
- Experimental results (Kungl et al., 2018 in preparation)

Training an Example Network



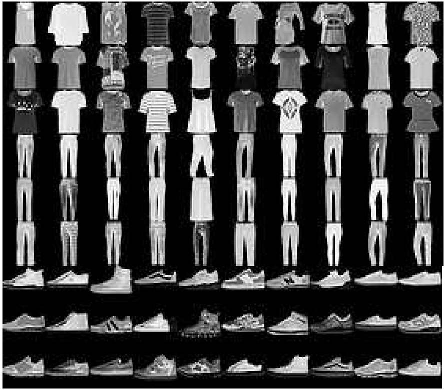
- Train the hardware in the loop with the wake-sleep algorithm (Hinton et al., 1995)

Training an Example Network



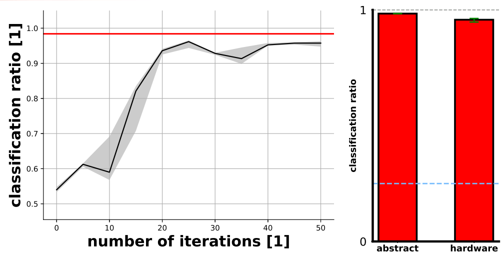
- Train the hardware in the loop with the wake-sleep algorithm (Hinton et al., 1995)

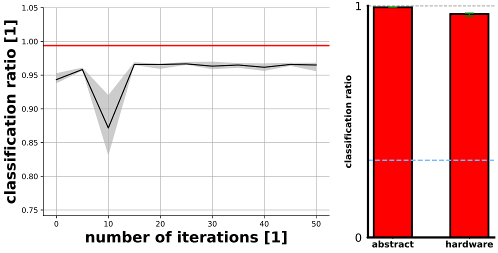
Datasets



- MNIST (LeCun et al., 1998)
- Fashion-MNIST (Xiao et al., 2017)

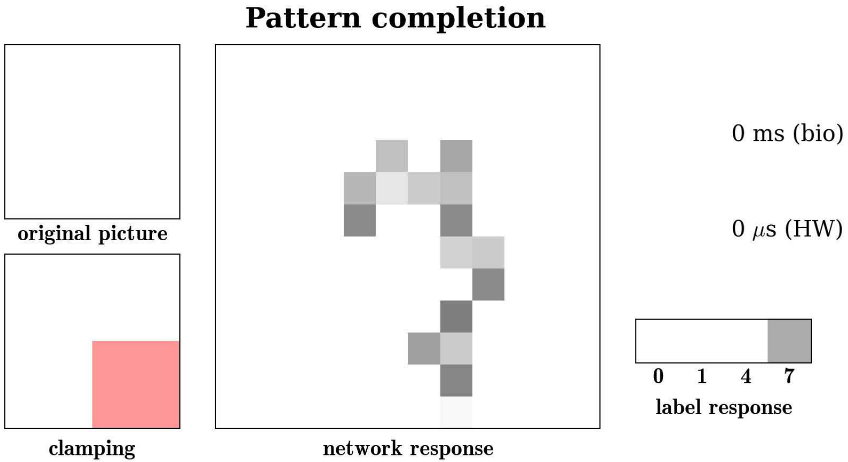
Classification (left: MNIST, right: Fashion-MNIST)



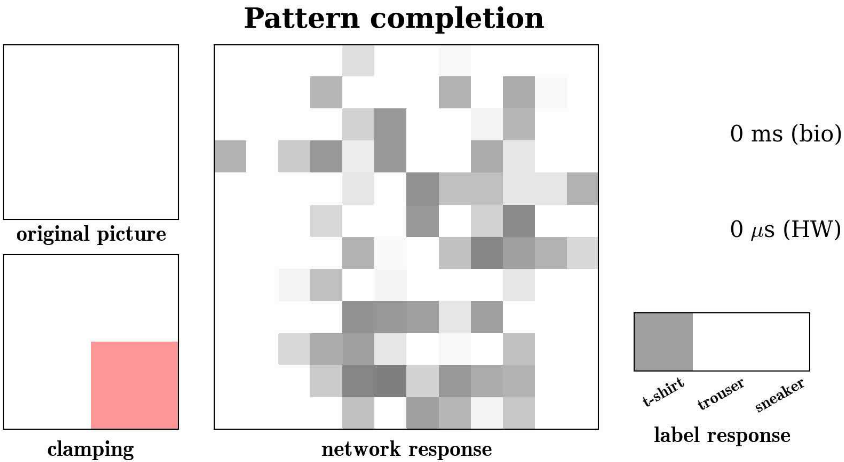


- MNIST: 0 1 4 7
- Fashion-MNIST: T-shirt/top, Trouser, Sneaker
- 400 Sea-of-Noise, 200 sampling neurons (visible 12×12, hidden, label), 50000 synapses

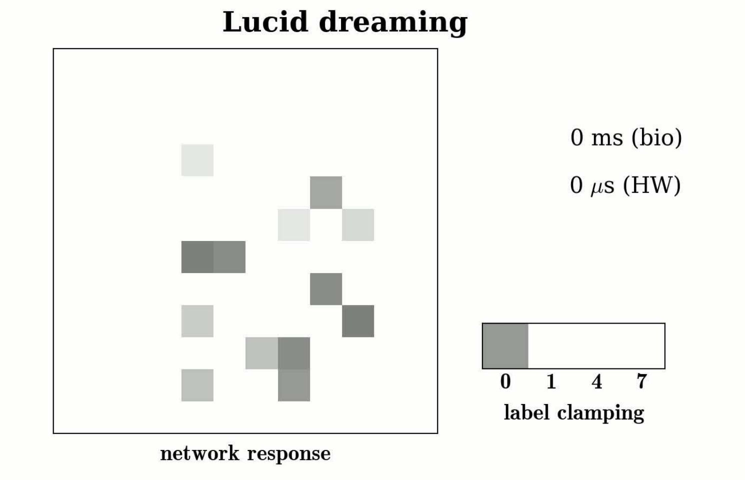
Pattern Completion (MNIST)



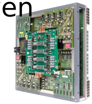
Pattern Completion (Fashion-MNIST)



Lucid Dreaming (MNIST)



- By clamping the label layer, the visible layer can be driven to dream of the given class



Summary

- Experiments on the BrainScaleS Wafer Scale System:
  - Spiking Deep Neural Network classifies MNIST
  - Winner-Take-All-like units solve the Constraint Satisfaction Problem Sudoku
  - Sea-of-Noise driven Restricted Boltzmann Machine classifies, completes patterns and dreams
- All rely on training the hardware in the loop; on-chip learning/calibration work in progress
- Accelerated dynamics ($10000\times$ faster w.r.t. biology) pay off for inference and generation
- You are welcome to request access at `http://www.neuromorphic.eu`