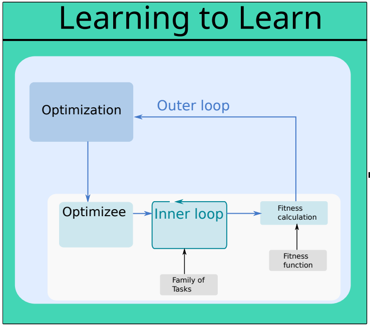
Optimizing Spiking Neural Networks with L2L on HPC systems

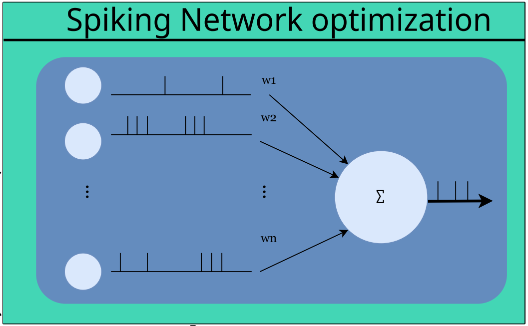
End of year colloquium

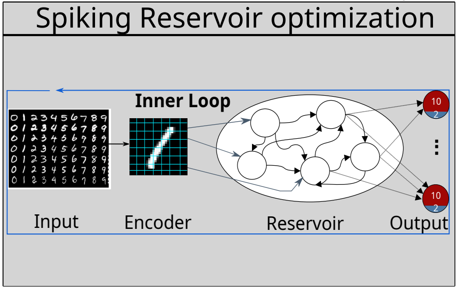
December 8, 2022 | **Alper Yeğenoğlu**[1,2] |

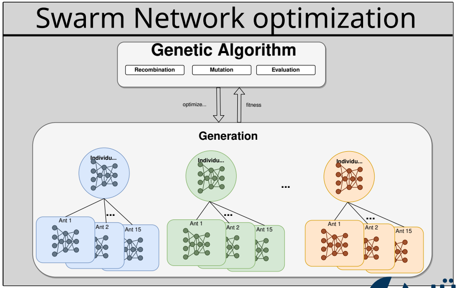
1. SDL Neurosciene, Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich
2. Institute of Geometry and Applied Mathematics, Department of Mathematics, RWTH Aachen|

a.yegenoglu@fz-juelich.de

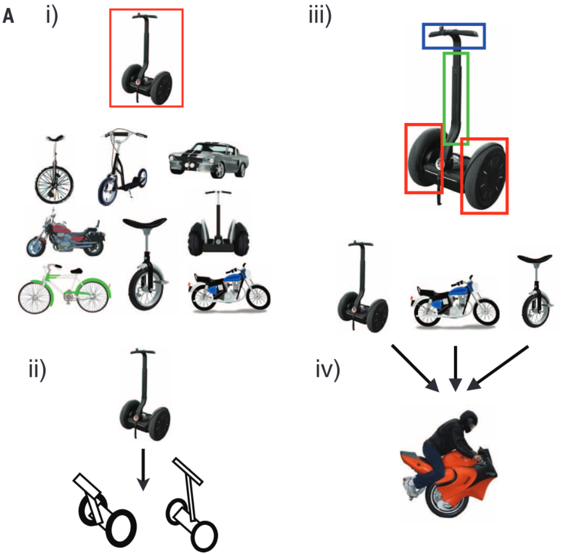

Learning to Learn
Optimization
Outer loop
Optimizee
Inner loop
Fitness calculation
Fitness function
Family of Tasks


Spiking Network optimization
w1
w2
wn
Σ


Spiking Reservoir optimization
Inner Loop
Input
Encoder
Reservoir
Output

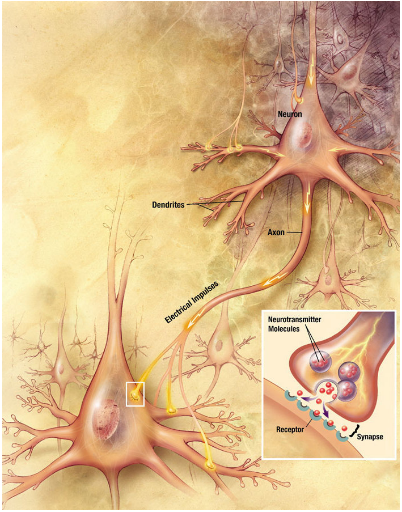

Swarm Network optimization
Genetic Algorithm
Recombination
Mutation
Evaluation
optimize...
fitness
Generation
Ant 1
Ant 2
Ant 15

Learning via Generalization and Concepts



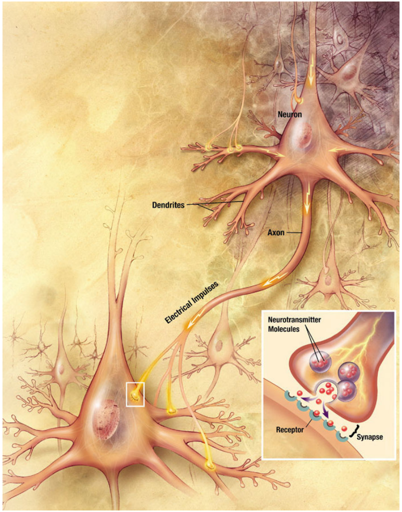
[Lake et al., Science 2015]

Neurons and Action Potentials

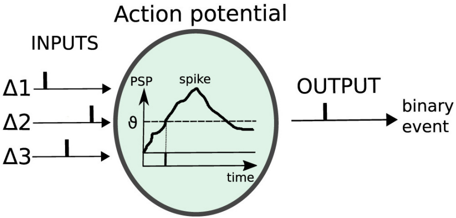


[US National Institute of Health]

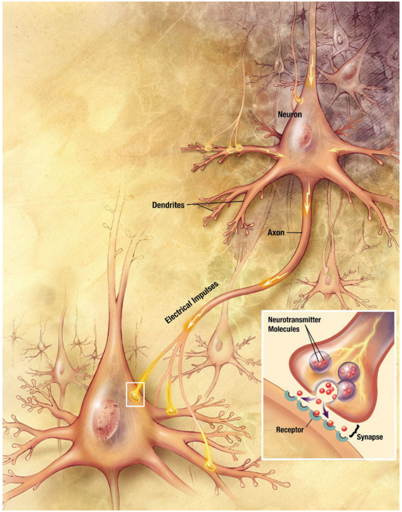
Neurons and Action Potentials



[US National Institute of Health]

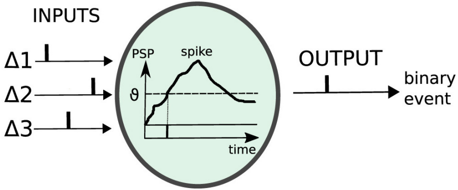


Neurons and Action Potentials

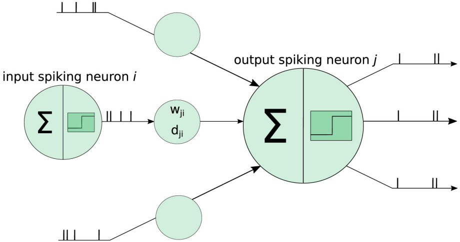


[US National Institute of Health]

Action potential



Spiking Neural Network (SNN)

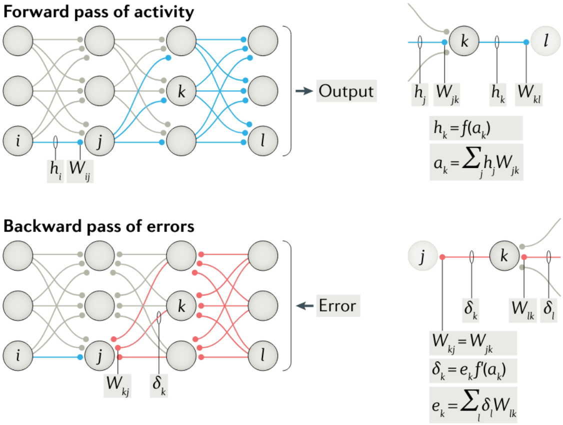


[Lobo et al., Neural Networks 121, 2020]

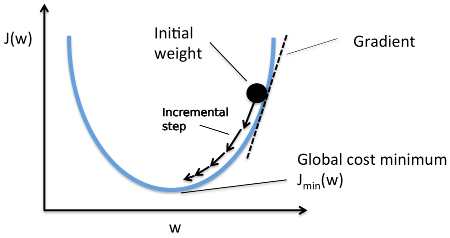
JÜLICH Forschungszentrum

Gradient Descent not possible on SNNs

ANN



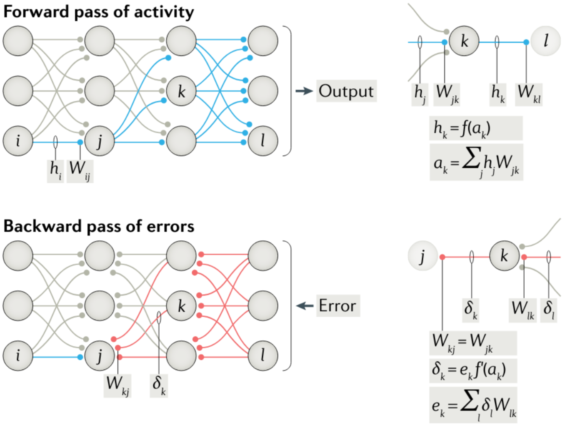
[Lillicrap et al., Nature Reviews NS, 2020]



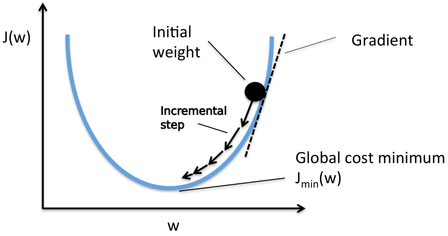
https://rasbt.github.io/mlxtend

Gradient Descent not possible on SNNs

ANN

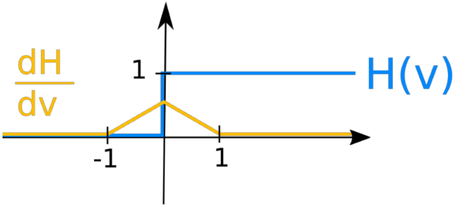


[Lillicrap et al., Nature Reviews NS, 2020]



https://rasbt.github.io/mlxtend

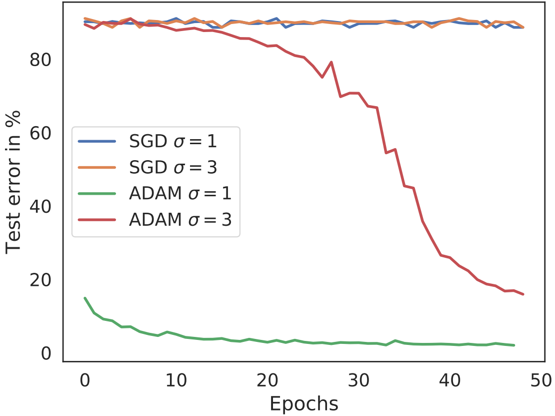
SNN



Step function over spike. Gradient descent and backprop not possible.

Gradient Descent Issues with ANNs

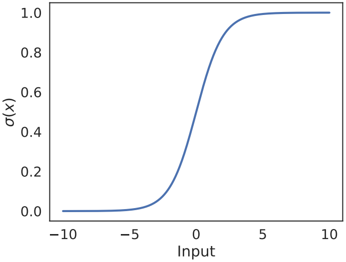
- Problem of Vanishing and Exploding Gradients
- In backpropagation step $\rightarrow$ zero or huge gradients



[Yegenoglu et al., LOD 2020]

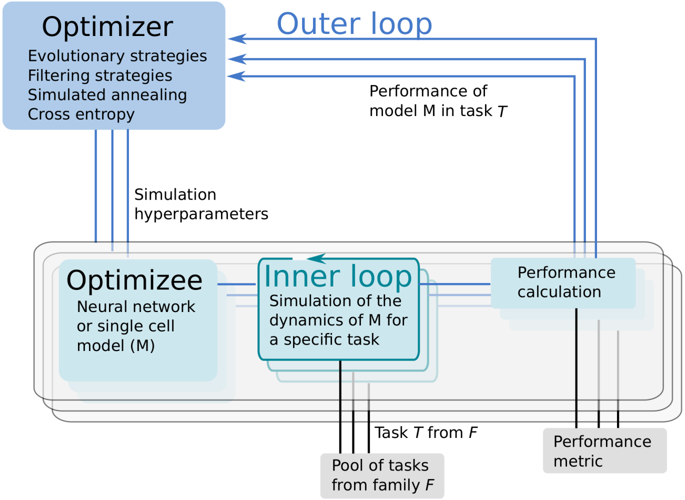
Problem depends on:

- Initialization of weights e.g. $w_{i,j} \sim \mathcal{N}(0,1)$
- Activation Functions



Logistic Function: $\sigma(x) = \frac{1}{1+e^{-x}}$

Optimization with Learning to Learn

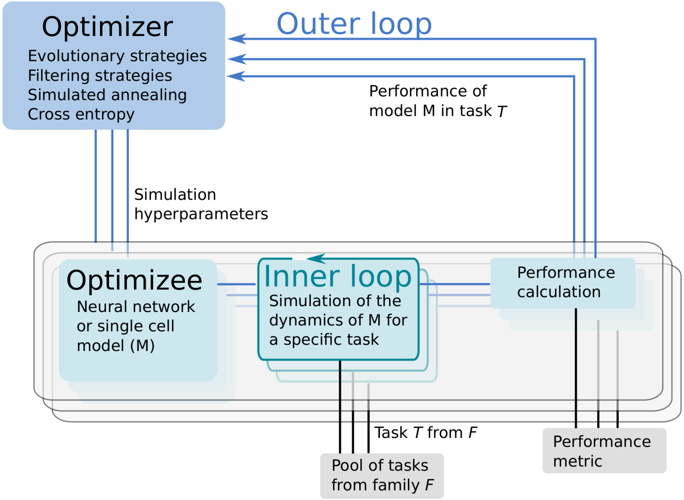


[Yegenoglu et al., Front. Comput. Neurosci. 2022]

Learning to Learn (L2L)

- Generalization on new data sets via experience
- Parameter space exploration
- Variety of optimization algorithms
- e.g. Ensemble Kalman Filter (**EnKF**)

Optimization with Learning to Learn

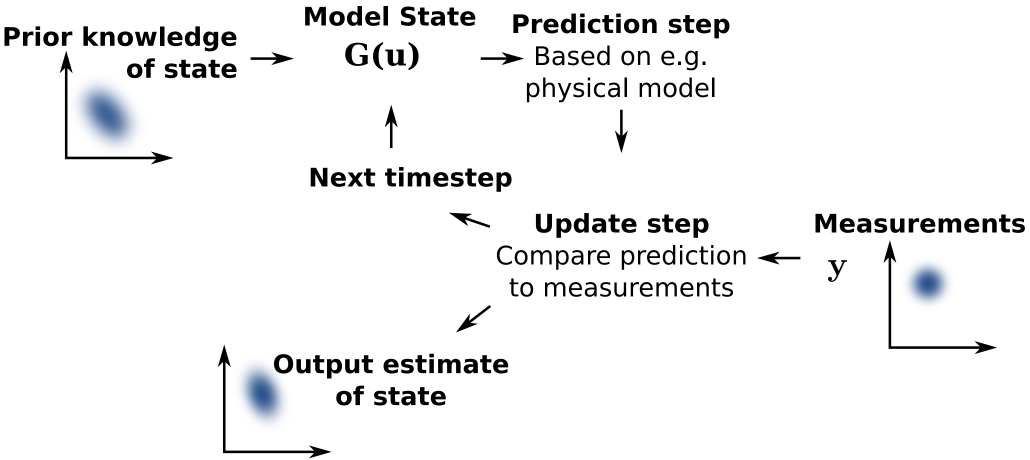


[Yegenoglu et al., Front. Comput. Neurosci. 2022]

Learning to Learn (L2L)

- Generalization on new data sets via experience
- Parameter space exploration
- Variety of optimization algorithms
- e.g. Ensemble Kalman Filter (**EnKF**)

Kalman Filter - Intuition



[https://en.wikipedia.org/wiki/Kalman_filter] modified

Ensemble Kalman Filter [Iglesias et al., 2013]

$$\mathbf{u}_j^{n+1} = \mathbf{u}_j^n + \mathbf{C}(\mathbf{U}^n)\left(\mathbf{D}(\mathbf{U}^n) + \boldsymbol{\Gamma}^{-1}\right)^{-1}\left(\mathbf{y} - \mathcal{G}(\mathbf{u}_j^n)\right) \tag{1}$$

- where $\mathbf{U}_j^n = \{\mathbf{u}\}_{j=1}^J$, n iteration index, J number of ensembles, $\mathbf{y}$ target
- $\mathbf{C}(\mathbf{U}^n) = \frac{1}{J}\sum_{j=1}^{J}(\mathbf{u}_j - \overline{\mathbf{u}}) \otimes (\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}})^T$
- $\mathbf{D}(\mathbf{U}^n) = \frac{1}{J}\sum_{j=1}^{J}(\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}}) \otimes \left(\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}}\right)^T$
- $\boldsymbol{\Gamma} = \gamma \mathbf{I}$

Ensemble Kalman Filter [Iglesias et al., 2013]

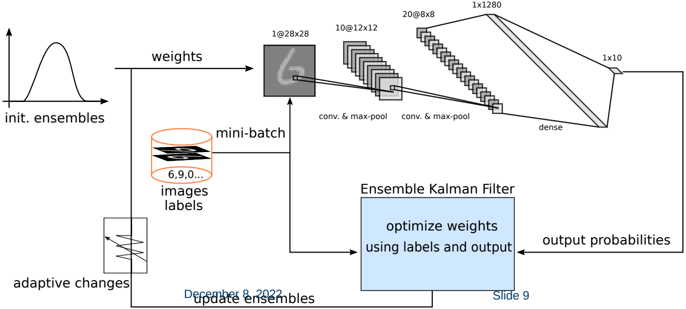
$$\mathbf{u}_j^{n+1} = \mathbf{u}_j^n + \mathbf{C}(\mathbf{U}^n)\left(\mathbf{D}(\mathbf{U}^n) + \boldsymbol{\Gamma}^{-1}\right)^{-1}\left(\mathbf{y} - \mathcal{G}(\mathbf{u}_j^n)\right) \tag{1}$$

- where $\mathbf{U}_j^n = \{\mathbf{u}\}_{j=1}^J$, n iteration index, J number of ensembles, $\mathbf{y}$ target
- $\mathbf{C}(\mathbf{U}^n) = \frac{1}{J}\sum_{j=1}^{J}(\mathbf{u}_j - \overline{\mathbf{u}}) \otimes (\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}})^T$
- $\mathbf{D}(\mathbf{U}^n) = \frac{1}{J}\sum_{j=1}^{J}(\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}}) \otimes \left(\mathcal{G}(\mathbf{u}_j) - \overline{\mathcal{G}}\right)^T$
- $\boldsymbol{\Gamma} = \gamma \mathbf{I}$

Minimization problem: $\Phi(\boldsymbol{u}, \boldsymbol{y}) = ||\mathbf{y} - \mathcal{G}(\boldsymbol{u}_j^n)||_{\Gamma}^2$

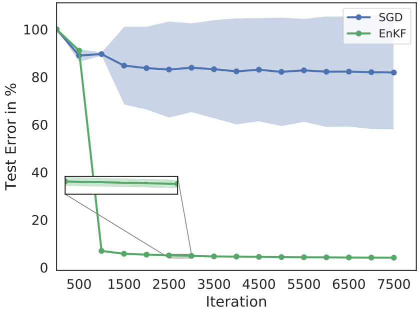
Classification with EnKF

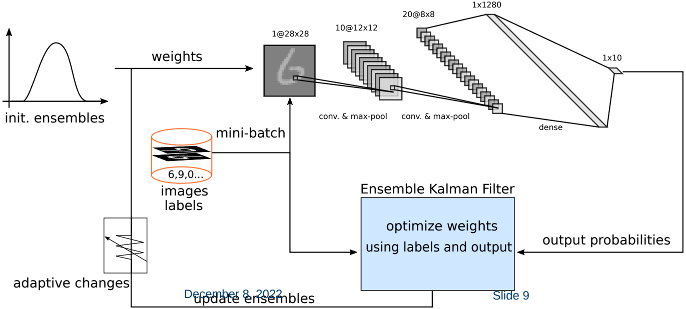
- MNIST & Letter dataset
- Logistic function
- Optimizer: EnKF
- [Yegenoglu et al., LOD 2020]



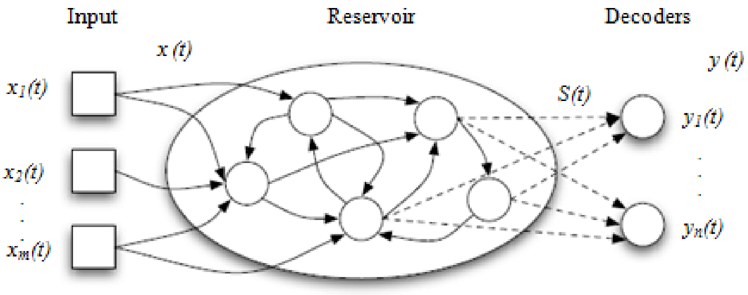
Classification with EnKF

- MNIST & Letter dataset
- Logistic function
- Optimizer: EnKF
- [Yegenoglu et al., LOD 2020]





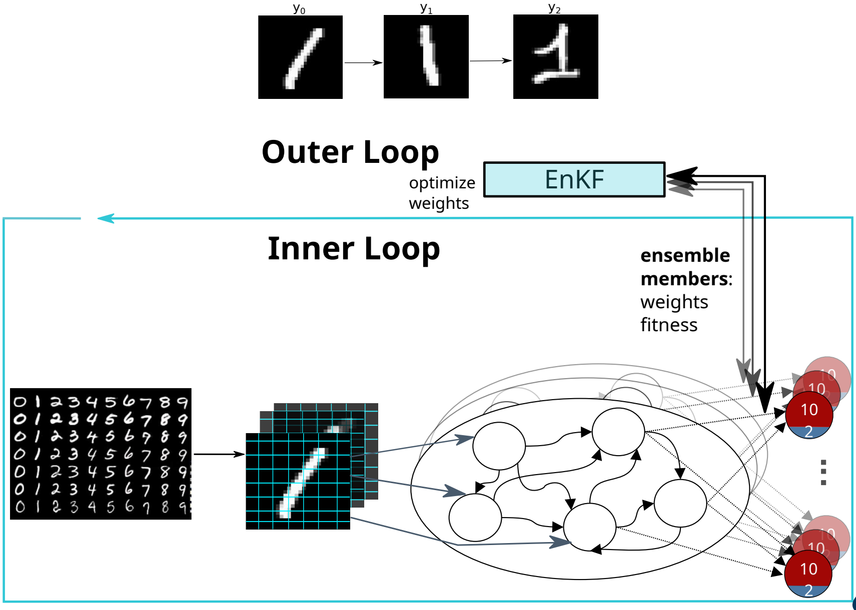
Reservoir Computing



[Avesani et al., Neural Networks 2015]

- input image encoded into firing rates
- fixed reservoir
- output connections are trained

Optimizing with L2L a Spiking Neural Network

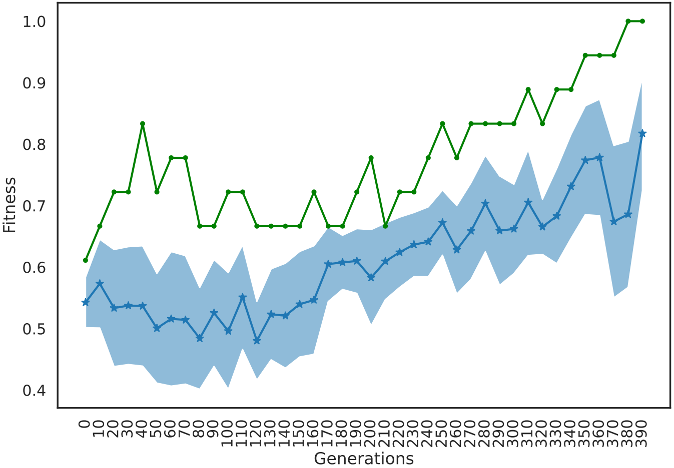


Reservoir Fitness

- 98 individuals (connection weights)
- 7 nodes à 16 tasks

Reservoir Fitness

- 98 individuals (connection weights)
- 7 nodes à 16 tasks



Swarm Optimization

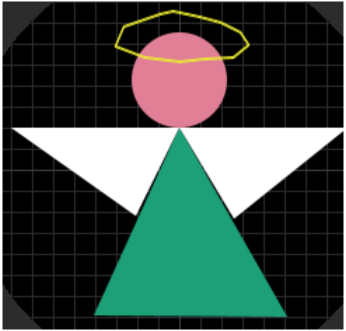
- Foraging for food and avoiding obstacles
- Collaboration and communication
- Evolution over (long) time/generations

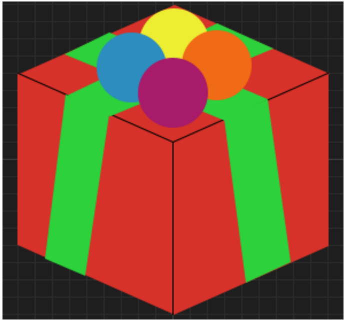
Swarm Optimization

- Foraging for food and avoiding obstacles
- Collaboration and communication
- Evolution over (long) time/generations
- **here**: optimize *agents* to help *Nikolaus* to collect *presents*

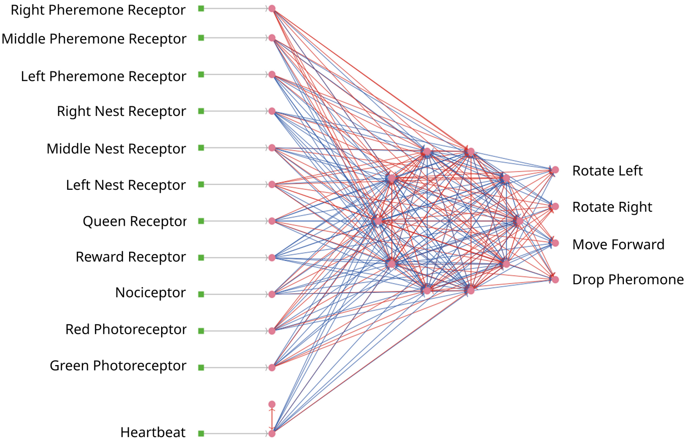
Setting – Agents


(a) Nikolaus


(b) Agent of Nikolaus aka angel


(c) Presents

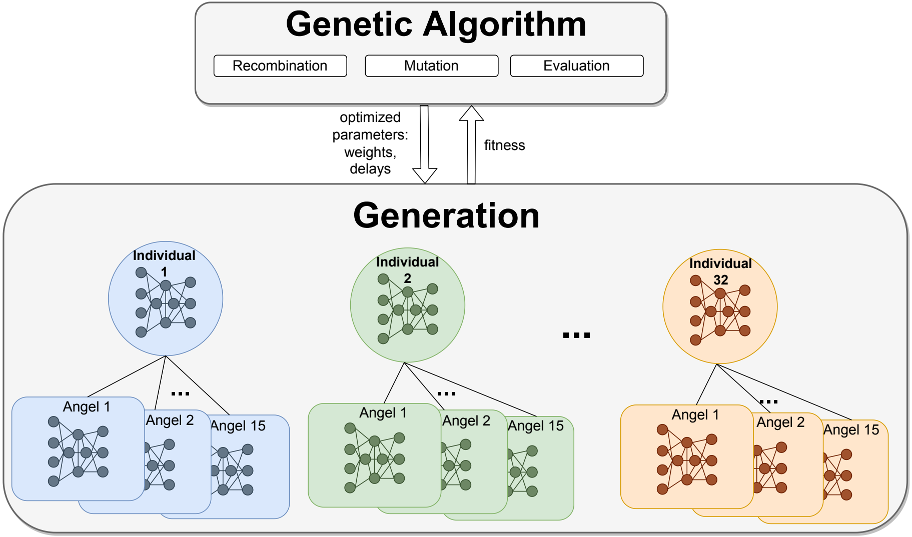
Setting – Network & Environment



(a) Agent brain

(b) Swarm collecting gifts

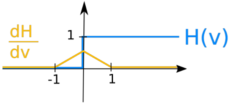
Optimization



Outlook

- Extend to different datasets
- Learning parameter mapping with ANNs (e.g. auto-encoder)
- Neuro-architecture search with evolutionary algorithms (neuroevolution)
- Swarm evolution using stigmergy: ants (multiple pheromones), drones

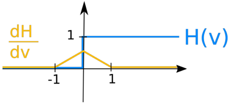
Summary

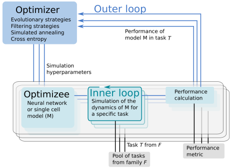


- Training SNNs is not straightforward

Summary

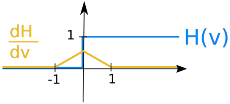
- Training SNNs is not straightforward
- Optimization via L2L and EnKF



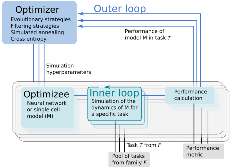


Summary

- Training SNNs is not straightforward



- Optimization via L2L and EnKF

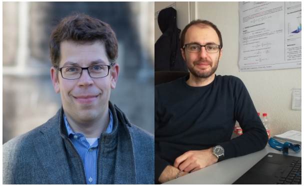


- Different learning tasks/applications

Abigail Morrison

Sandra Diaz Pier

Kai Krajsek



Michael Herty

Giuseppe Visconti

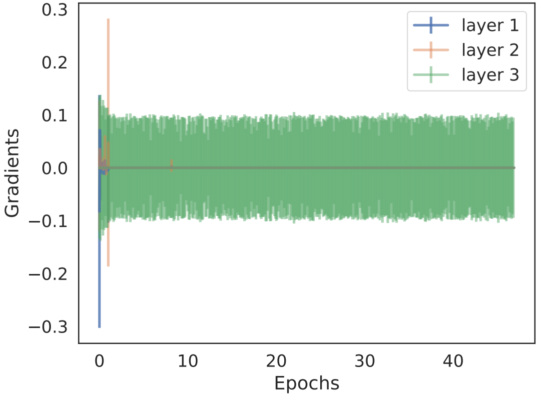
Thank you for your attention

and a happy Christmas & new year

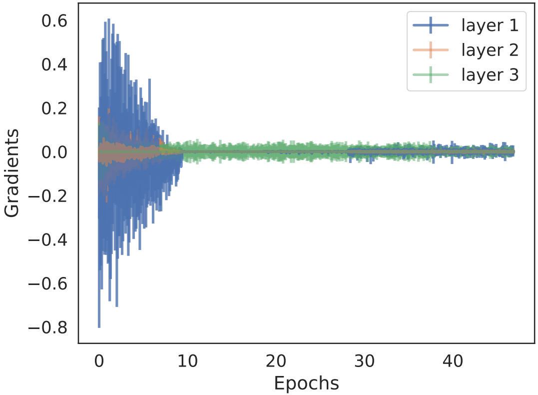
contact: a.yegenoglu@fz-juelich.de

Appendix

SGD & Adam: Backpropagated Gradients over Epochs

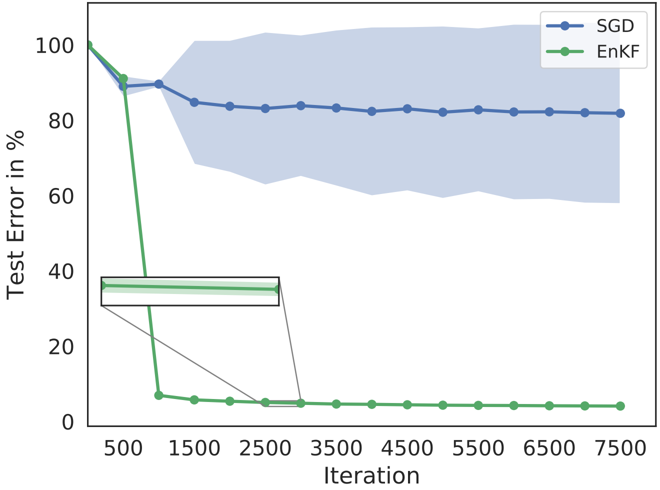


(a) SGD: Mean and standard deviation of the backpropagated gradients.



(b) Adam: Mean distributions of activation values over 50 epochs.

Test Error for different Optimizers



Mean test error (dark line) of the network for 10 runs trained for one epoch. The shaded area is the standard deviation. [Yegenoglu et al., LOD 2020]

Reservoir dynamics

[Maass et al., 2002]:

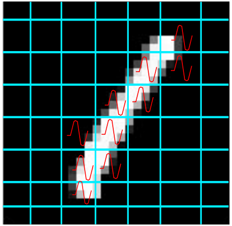
$$x^M = (L^M u)(t) \tag{2}$$

x^M reservoir state (activation patterns), $u(\cdot)$ spike sequence encoded input, L^M filter for transforming from input to reservoir

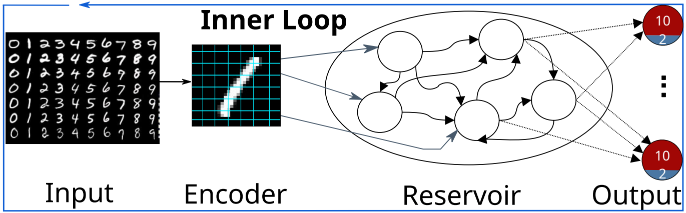
$$y(t) = f^M(x^m(t)) \tag{3}$$

$y(t)$ output, f^M memory-less readout map

Spiking Network Input Output Transformation



(a) Spike on intensity change

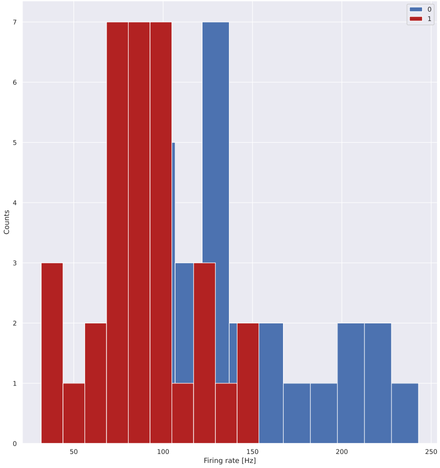


(b) Highest activity (FR) is captured on the output via softmax

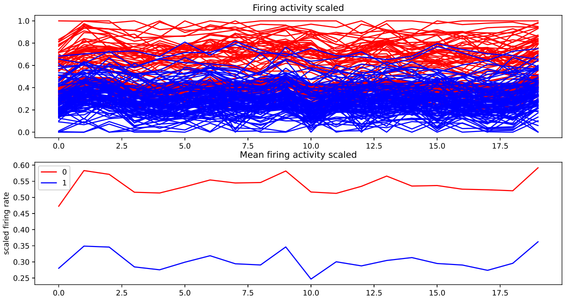
Sampling step

- Problem of convergence
- Sampling step in which worst ρ_{best} individuals are replaced by best ρ_{worst} ones
  - adding gaussian noise univariate or mulitvariate (gaussian mixture)
  - random pick of best or best pick first
  - e.g. 10% best replacing 10% worst

Seperability of Firing Rates



(a) Distribution over time



(b) Firing rates scaled and averaged over time

Algorithm for PCA on Covariance Matrix

- calculate/fit the PCA for cov mat on one output
- get n components of PCA
- check min and max create a mask with certain ranges e.g. $-0.015 < n$ or $n < 0.015$
- apply the mask on the cov matrix to get important values
- repeat for other outputs as well

Fitness function

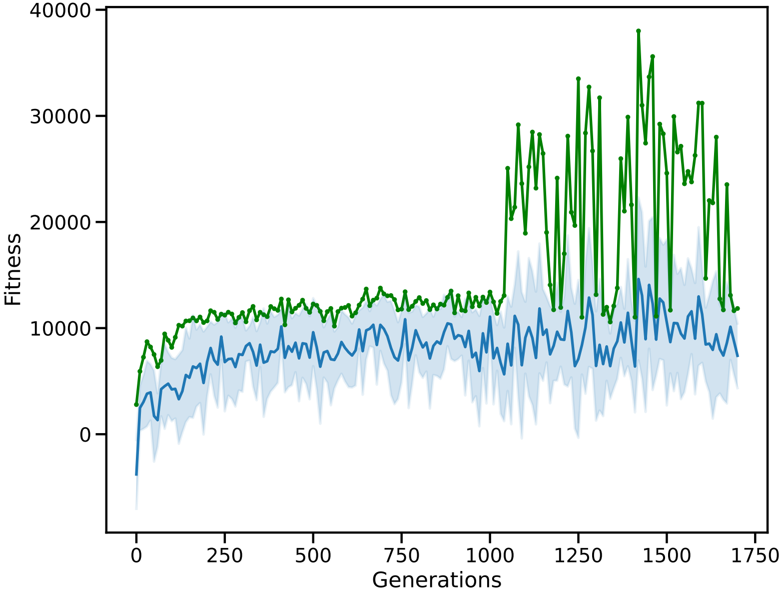
Cost & Values of Ant Colony:

Behavior	Value
Resting	-0.5
Dropping Pheromone	-0.05
Rotation	-0.02
Movement	-0.25
Return nest $\mathcal{N}$	220
Touch food $\mathcal{F}$	1.5
η	30.0
Timestep à 20ms	2000

$$f_i = \sum_{t=1}^{T_s} \left(\sum_{j=1}^{J} \mathcal{N}_{i,j}^{(t)} + \mathcal{F}_{i,j}^{(t)} - \mathcal{C}_{i,j}^{(t)} \right) + \eta \left(T - T_s \right),$$

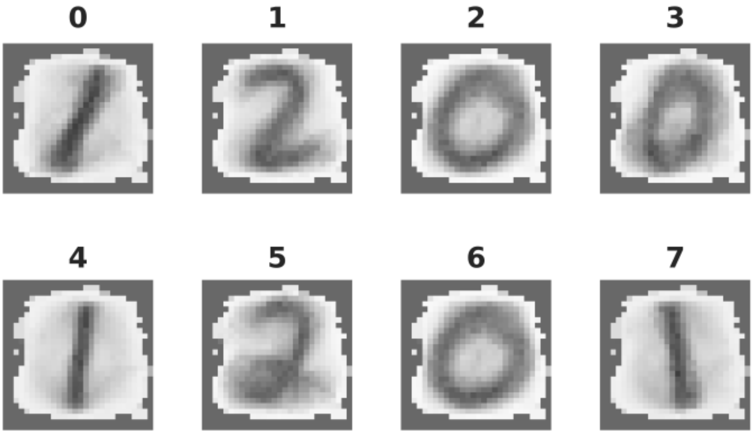
where i is individual, j is ant, T total simulation time, T_s spend simulation time

Fitness Swarm



JÜLICH Forschungszentrum

Representation space



[Weidel et al., 2020]

- Specialized for particular sub-categories of the input space