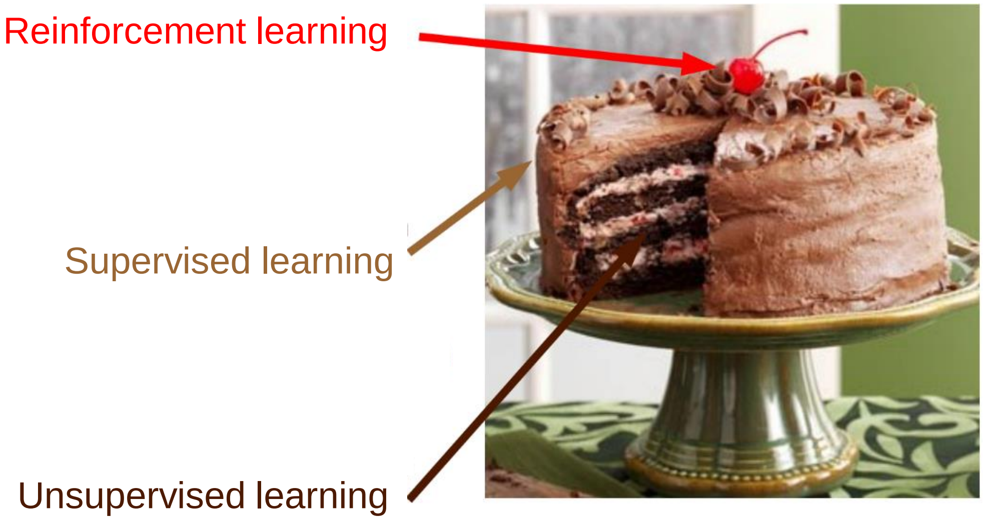
Spiking Convolutional Deep Belief Networks for Unsupervised High Level Feature Extraction and Pattern Reconstruction

FZI FORSCHUNGSZENTRUM INFORMATIK

“Most of human and animal learning is unsupervised learning.

If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake.

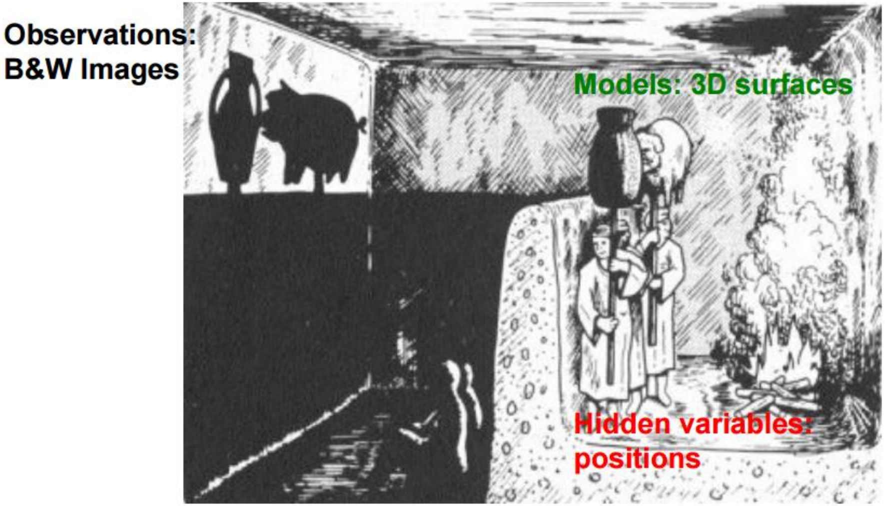
We know how to make the icing and the cherry, but we don't know how to make the cake.”



Y Lecun, NIPS 2016

“Learn the data”

“We believe that this can be achieved by learning a disentangled posterior distribution of the generative factors of the observed sensory input by leveraging the wealth of unsupervised data.”[1]



[1] I. Higgins. Early Visual Concept Learning with Unsupervised Deep Learning. 2016

Overview

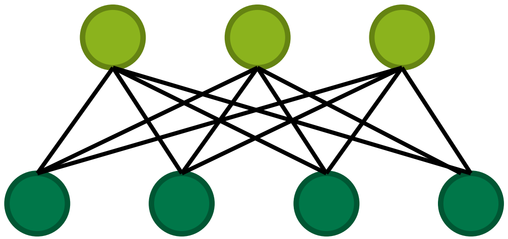
- Restricted Boltzmann Machines (RBMs)
- Convolutional Deep Belief Networks

- Neural Sampling
- Event-Driven Contrastive Divergence
- Conversion

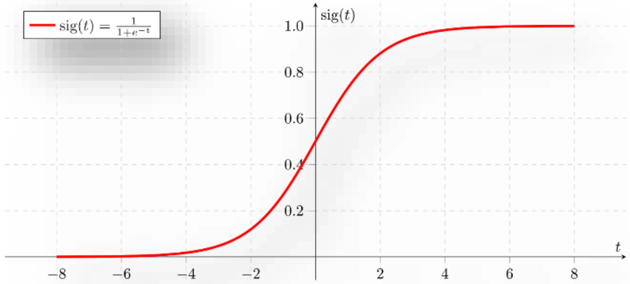
- Experiments & Results
- Conclusion

Restricted Boltzmann Machines

Neural Network:



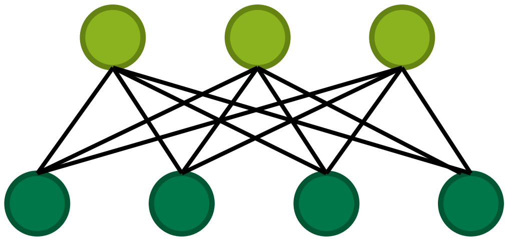
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"}\,active\,\text{"}) = \sigma(\text{"}\,input\,\text{"})$
- Energy-based Model

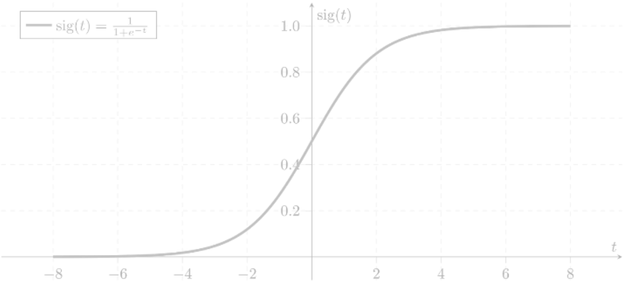


Restricted Boltzmann Machines

Neural Network:

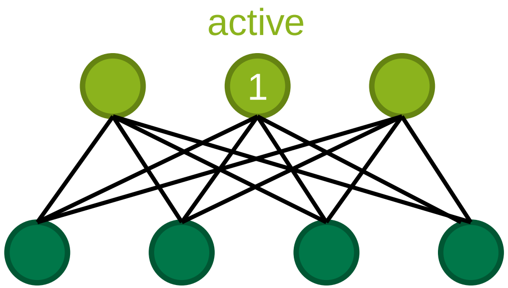
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"}active\text{"}) = \sigma(\text{"}input\text{"})$
- Energy-based Model



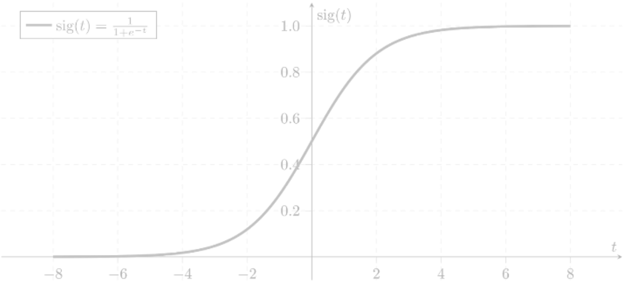


Restricted Boltzmann Machines

Neural Network:



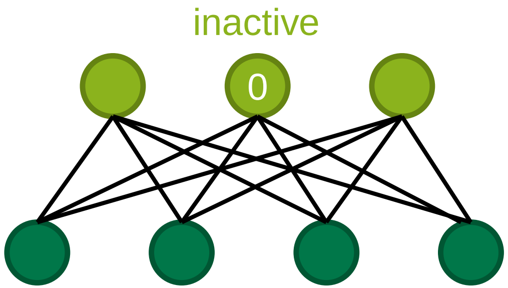
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"}active\text{"}) = \sigma(\text{"}input\text{"})$
- Energy-based Model

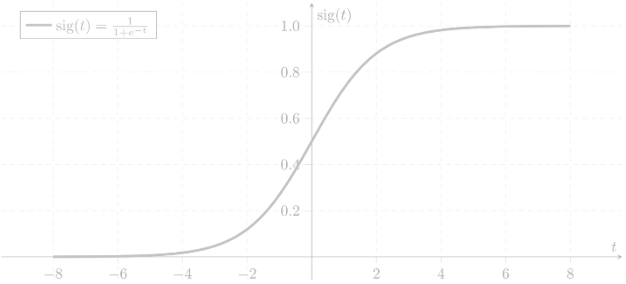


Restricted Boltzmann Machines

Neural Network:

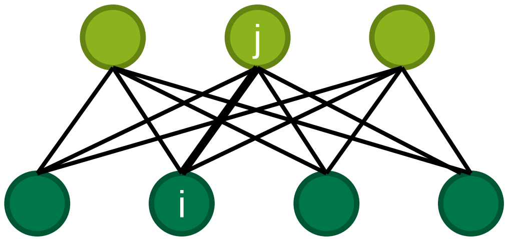
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"}active\text{"}) = \sigma(\text{"}input\text{"})$
- Energy-based Model



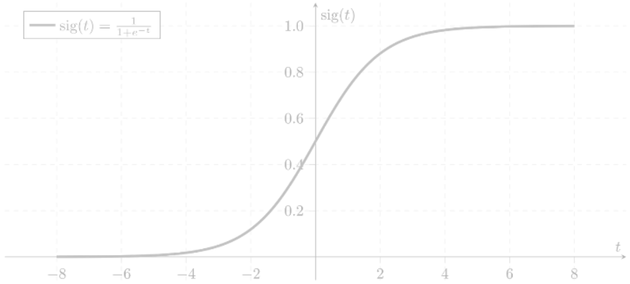


Restricted Boltzmann Machines

Neural Network:



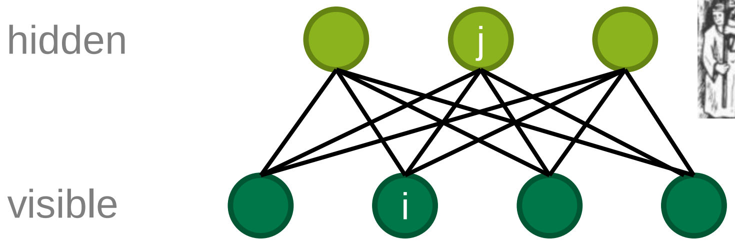
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{" }active\text{ "}) = \sigma(\text{" }input\text{ "})$
- Energy-based Model

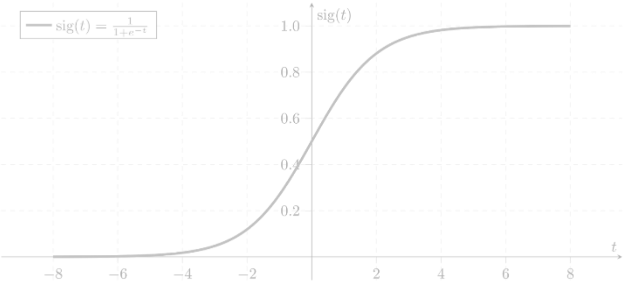


Restricted Boltzmann Machines

Neural Network:

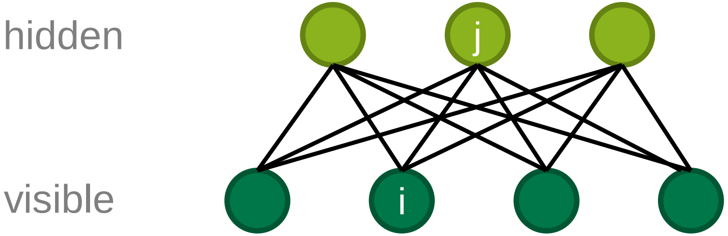
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{" }active\text{ "}) = \sigma(\text{" }input\text{ "})$
- Energy-based Model



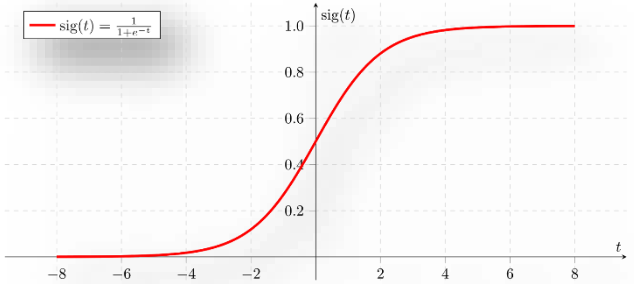


Restricted Boltzmann Machines

Neural Network:

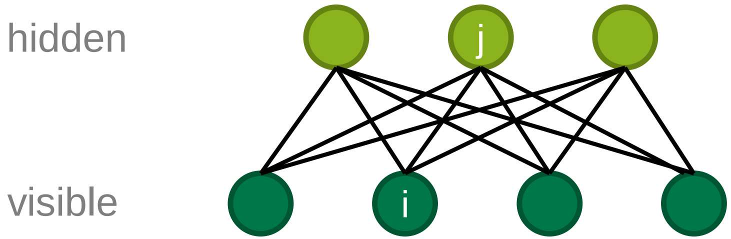


- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{" } active \text{ "}) = \sigma(\text{" } input \text{ "})$
- Energy-based Model

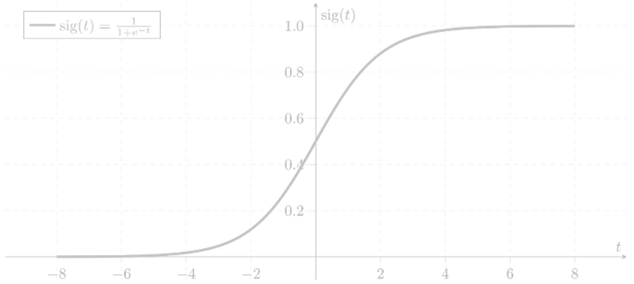


Restricted Boltzmann Machines

Neural Network:

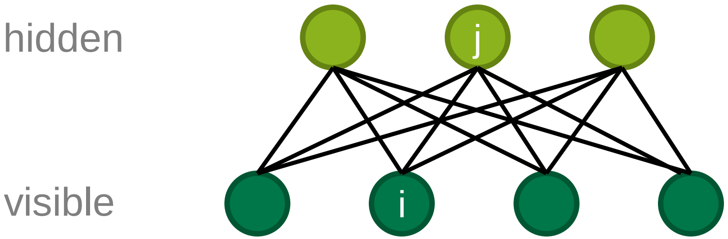


- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p("\,active\,") = \sigma("\,input\,")$
- Energy-based Model

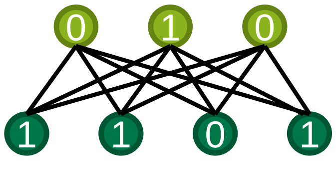


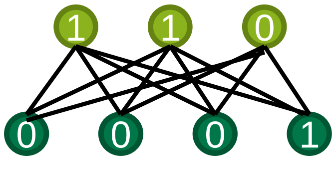
Restricted Boltzmann Machines

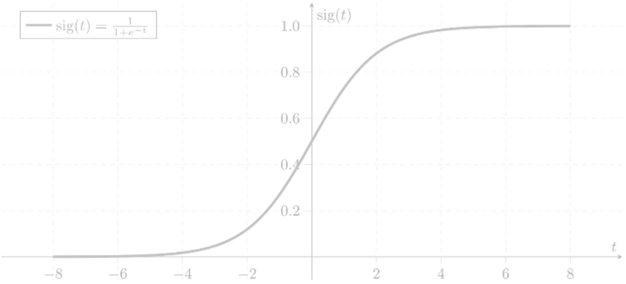
Neural Network:



- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"} active \text{"}) = \sigma(\text{"} input \text{"})$
- Energy-based Model

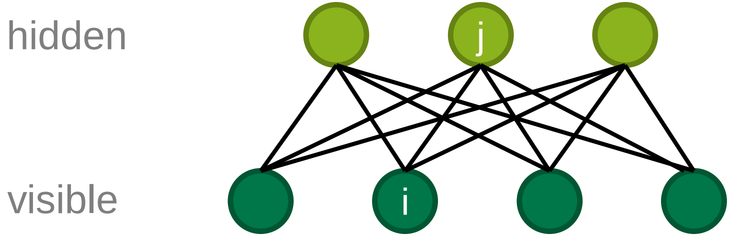




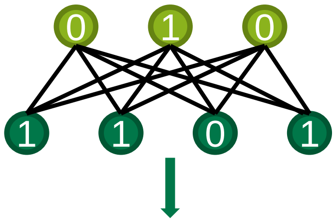


Restricted Boltzmann Machines

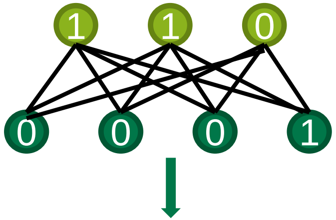
Neural Network:



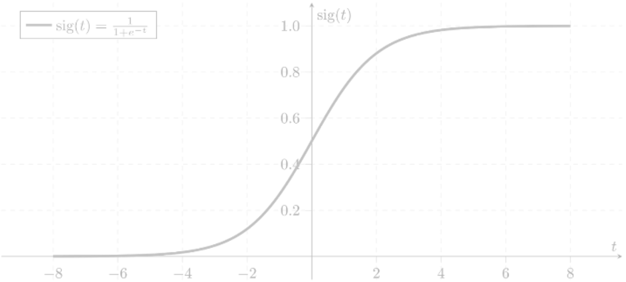
- Binary Units
- Symmetric Connections
- Two bipartite Layers
- Stochastic Activations $p(\text{"}\,active\,\text{"}) = \sigma(\text{"}\,input\,\text{"})$
- Energy-based Model



Energy = 5



Energy = 2

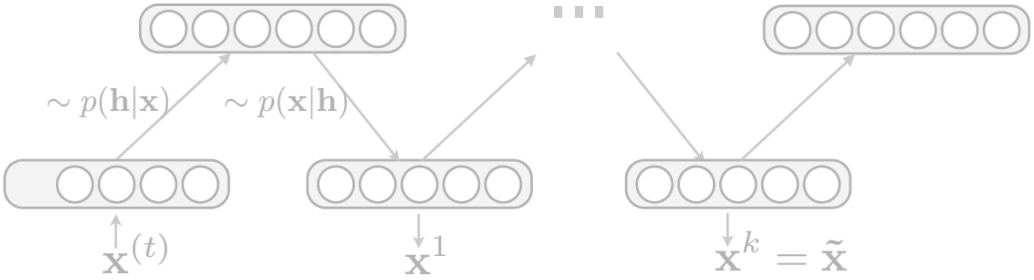


Training RBMs

→ Maximize probability of training samples ⇔ Minimize the Energy :

$$\frac{\partial E(v)}{\partial w} = \ \dots = \left(\mathrm{v\ h^T}\right)_{\mathrm{model}} - \left(\mathrm{v\ h^T}\right)_{\mathrm{data}}$$

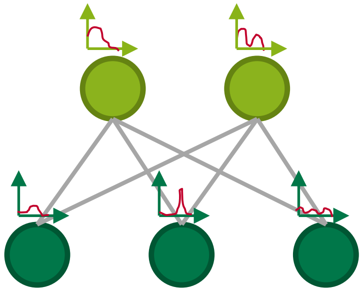
$$\rightarrow \mathrm{w^{new}} = \mathrm{w^{old}} + \ \mu \left((v\ h^T)_{data} - (v\ h^T)_{model} \right)$$



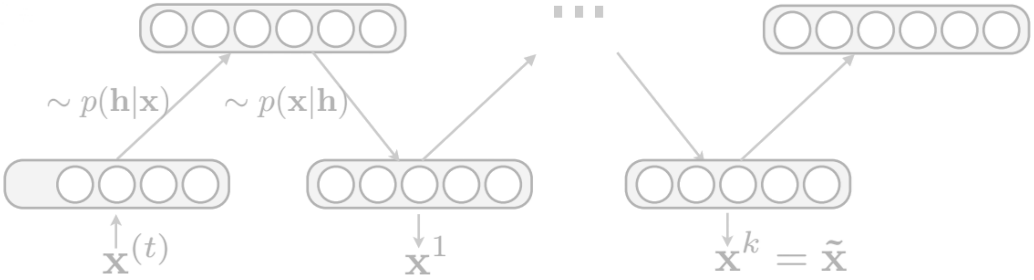
Training RBMs

→ Maximize probability of training samples ⇔ Minimize the Energy :

$$\frac{\partial E(v)}{\partial w} = \; \ldots = \left(\mathrm{v\,h^T}\right)_{\mathbf{model}} - \left(\mathrm{v\,h^T}\right)_{\mathrm{data}}$$


*

$$\rightarrow \mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu \left((v\,h^T)_{data} - (v\,h^T)_{model} \right)$$

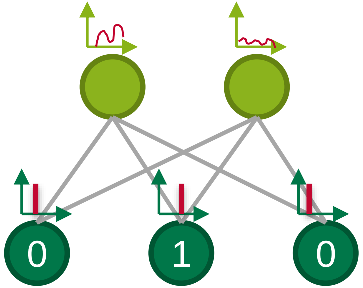


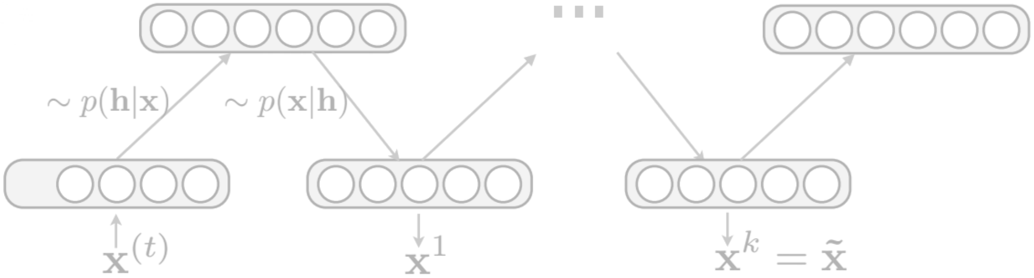
Training RBMs

→ Maximize probability of training samples ⇔ Minimize the Energy :

$$\frac{\partial E(v)}{\partial w} = \ \ldots = (\mathrm{v}\,\mathrm{h}^{\mathrm{T}})_{\mathrm{model}} - (\mathrm{v}\,\mathrm{h}^{\mathrm{T}})_{\mathbf{data}}$$

$$\rightarrow \mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$$



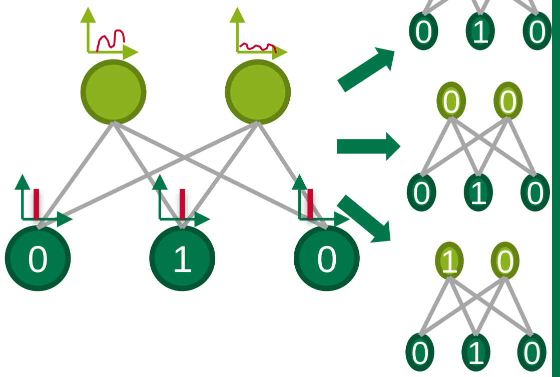


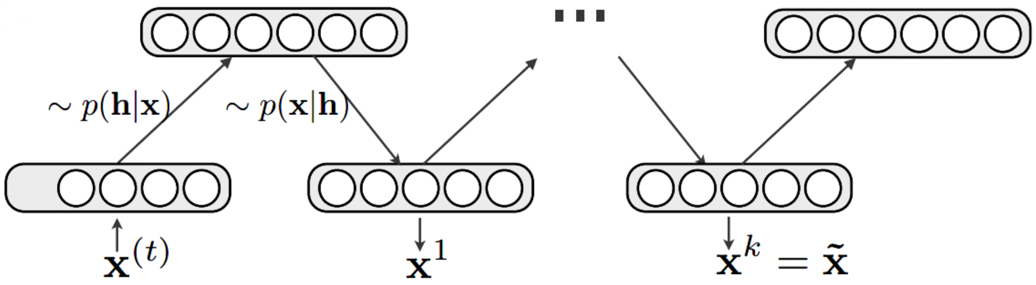
Training RBMs

→ Maximize probability of training samples ⇔ Minimize the Energy :

$$\frac{\partial E(v)}{\partial w} = \ \dots = (\mathrm{v\,h^T})_{\mathrm{model}} - (\mathrm{v\,h^T})_{\mathbf{data}}$$

$$\rightarrow \mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$$



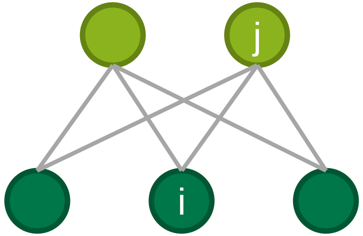


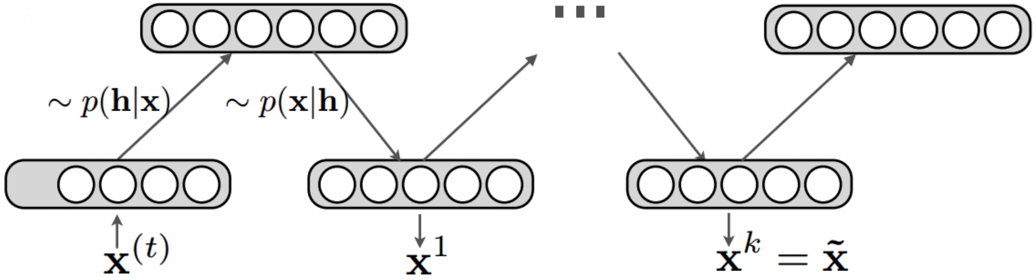
Training RBMs

→ Maximize probability of training samples ⇔ Minimize the Energy :

$$\frac{\partial E(v)}{\partial w} = \; \ldots = (\mathrm{v\,h^T})_{\mathrm{model}} - (\mathrm{v\,h^T})_{\mathrm{data}}$$

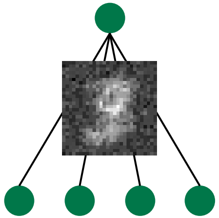
$$\rightarrow \mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$$





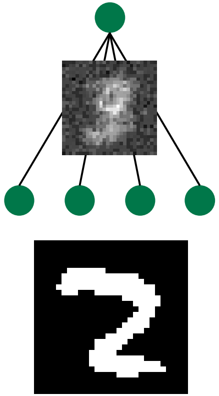
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



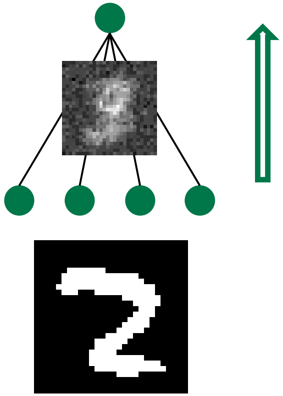
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu \left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



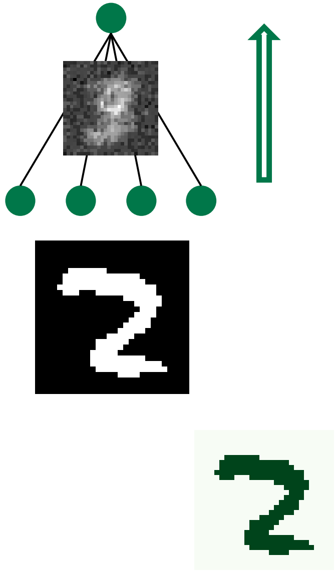
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



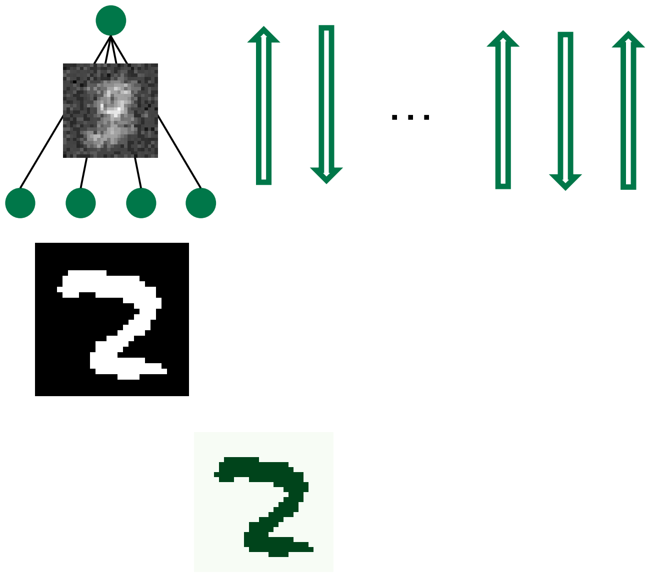
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu \left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



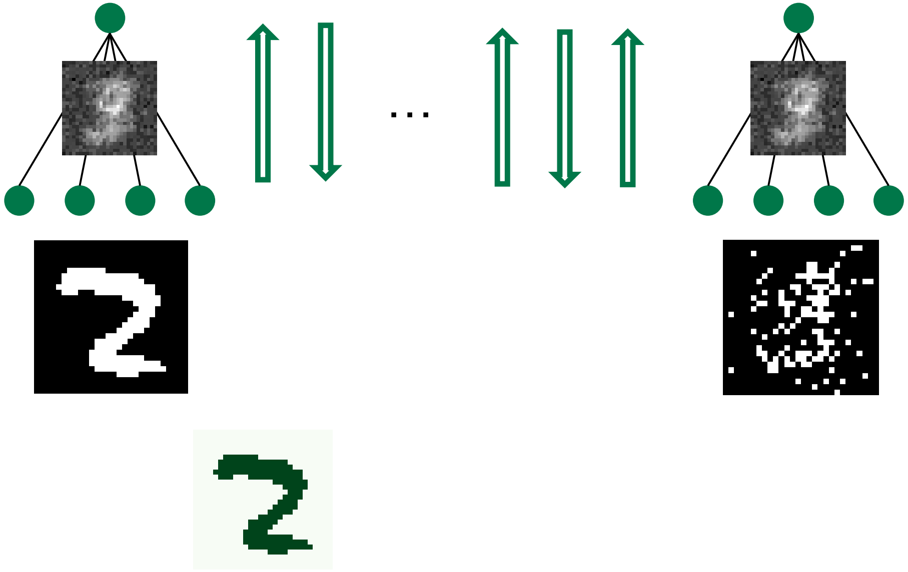
Training RBMs

Contrastive Divergence: $w^{new} = w^{old} + \mu \left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



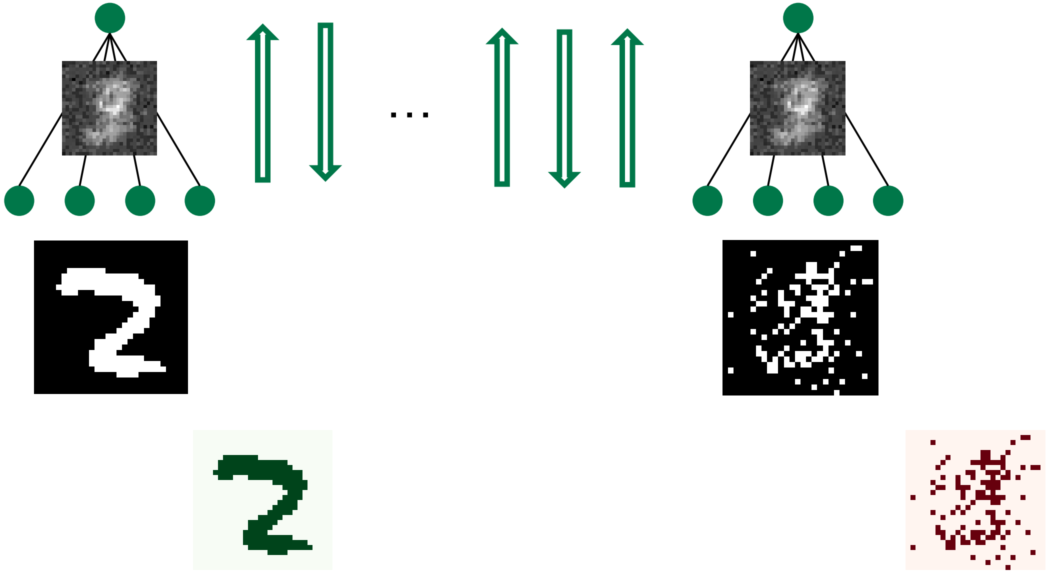
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



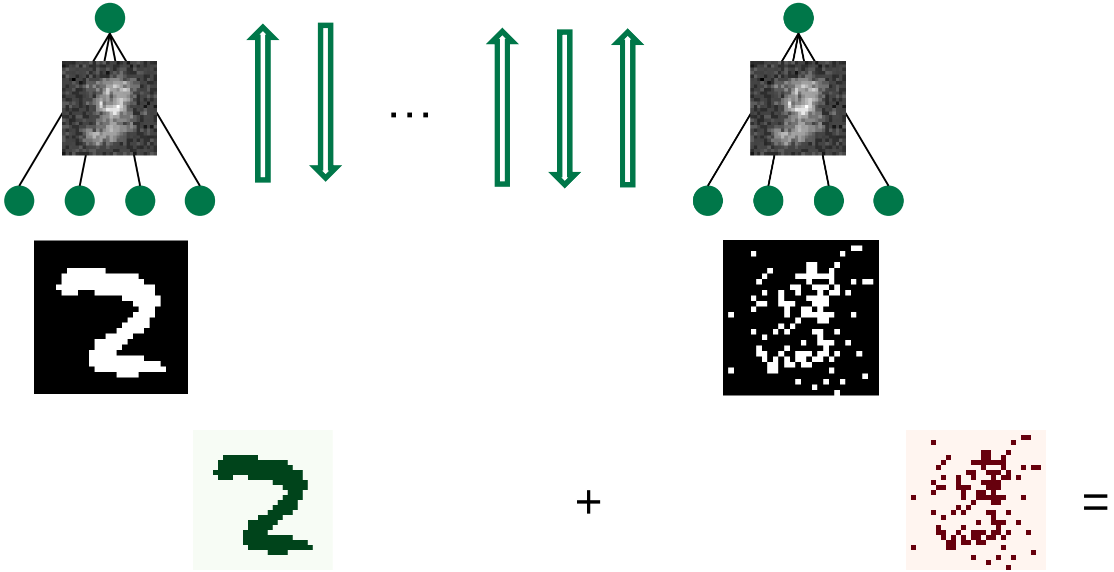
Training RBMs

Contrastive Divergence: $w^{new} = w^{old} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



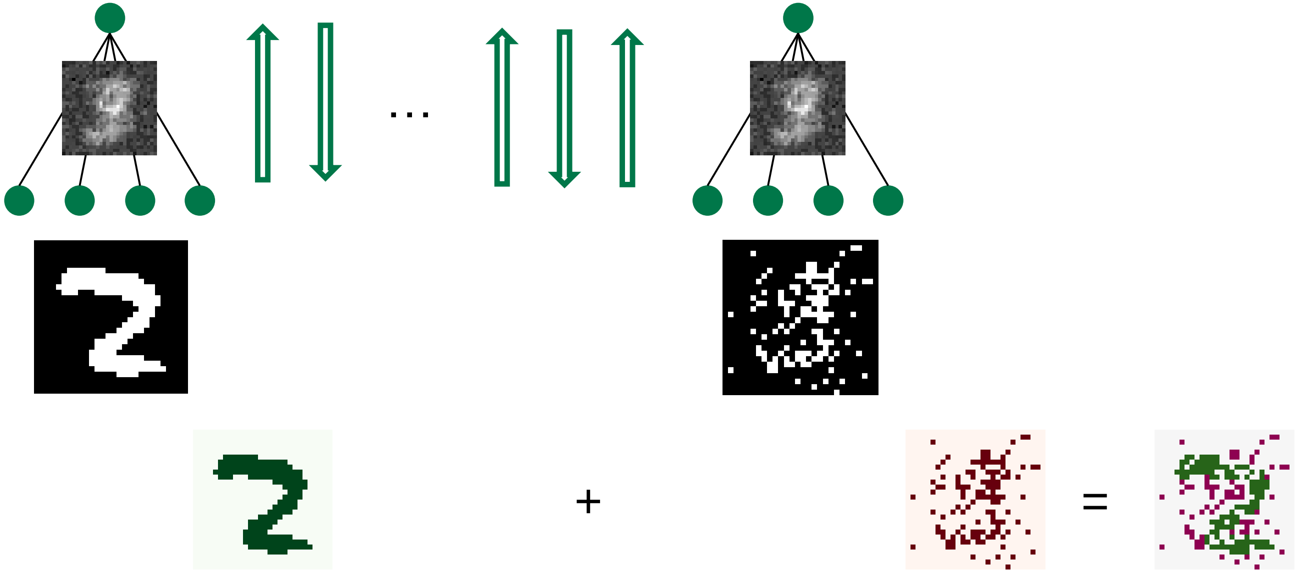
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



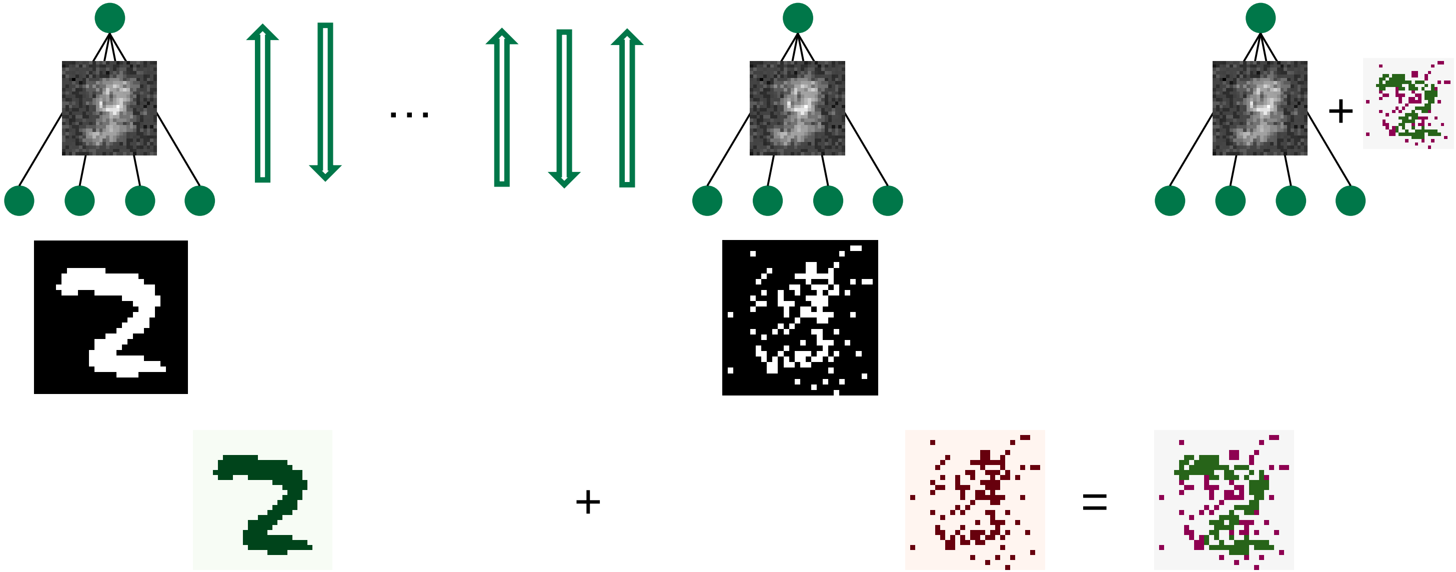
Training RBMs

Contrastive Divergence: $w^{new} = w^{old} + \mu\left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



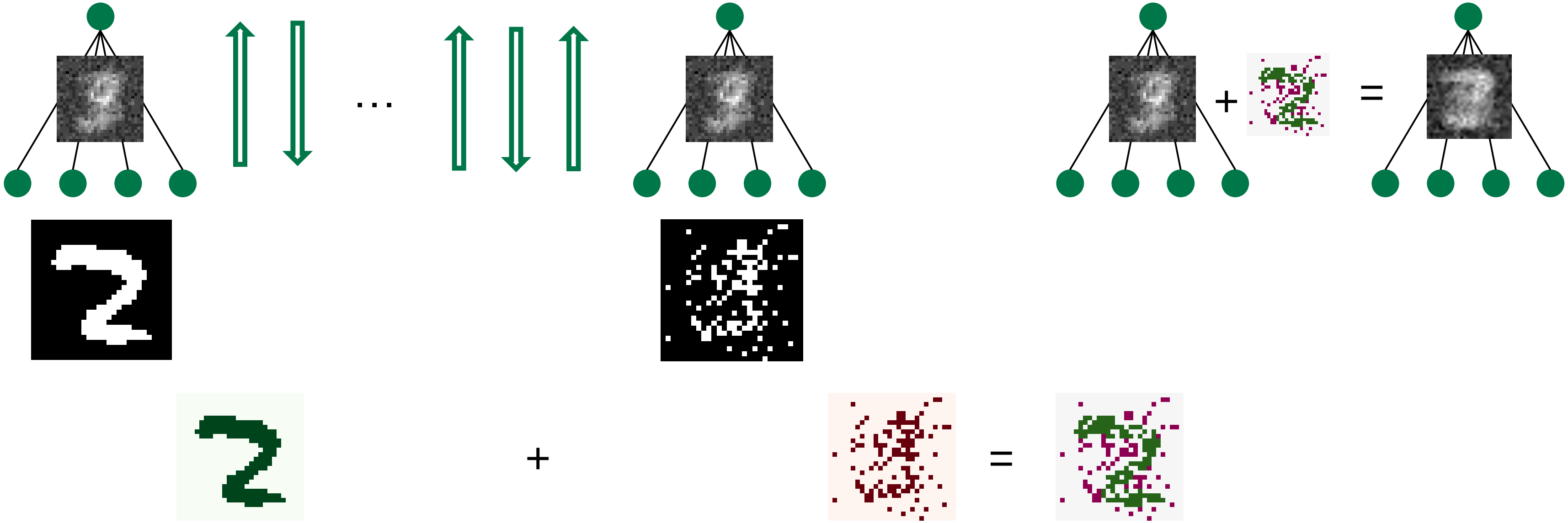
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



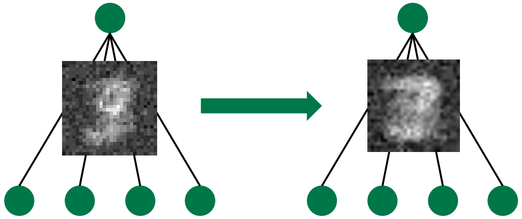
Training RBMs

Contrastive Divergence: $w^{new} = w^{old} + \mu \left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



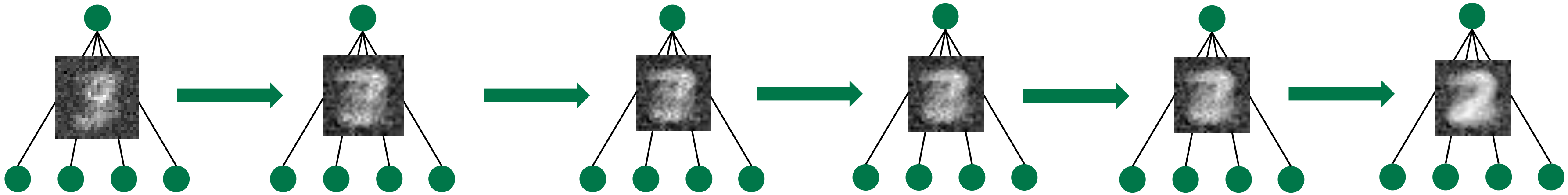
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu \left((v\, h^T)_{data} - (v\, h^T)_{model} \right)$



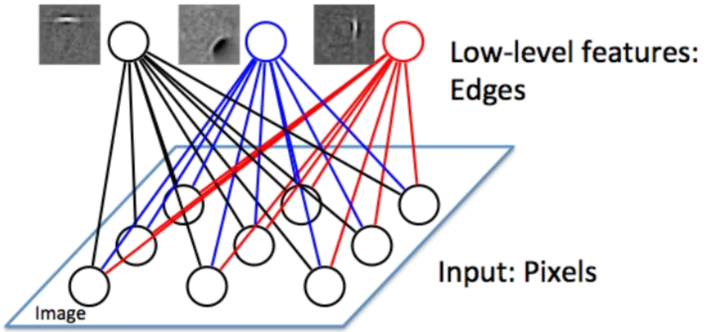
Training RBMs

Contrastive Divergence: $\mathrm{w}^{\mathrm{new}} = \mathrm{w}^{\mathrm{old}} + \mu\,(\,(v\,h^T)_{data} - (v\,h^T)_{model})$



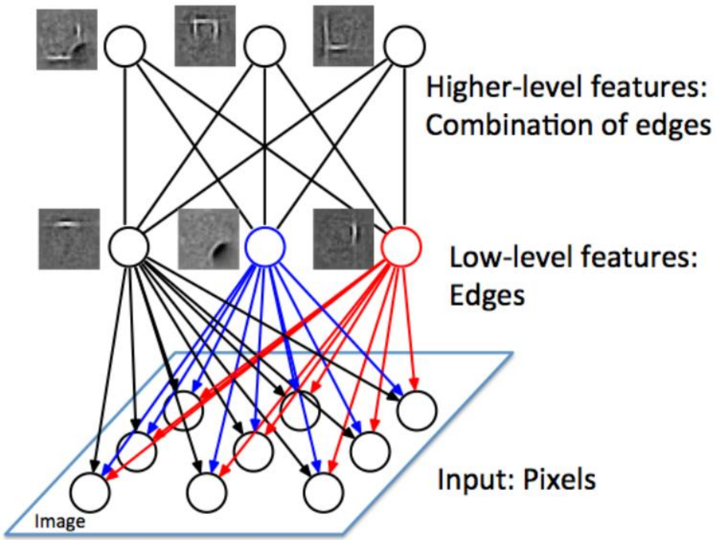
“Diving in Deeper”: Deep Belief Networks

- Adding Hidden Layers increases Representational Power
- DBNs can approximate any distribution over binary vectors



“Diving in Deeper”: Deep Belief Networks

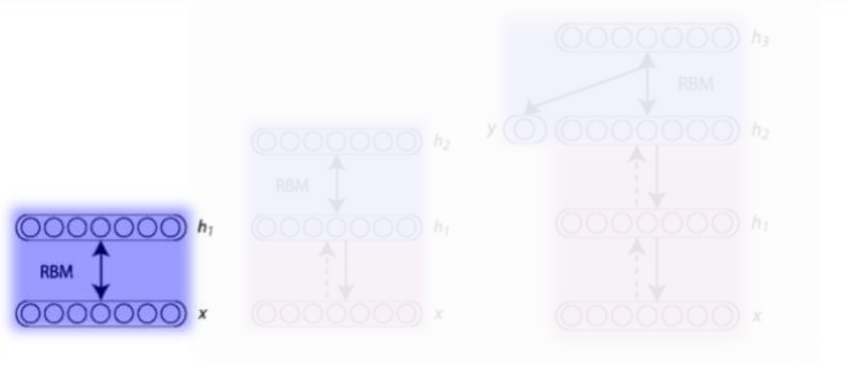
- Adding Hidden Layers increases Representational Power
- DBNs can approximate any distribution over binary vectors



“Diving in Deeper”: Deep Belief Networks

Built up by stacking up RBMs:

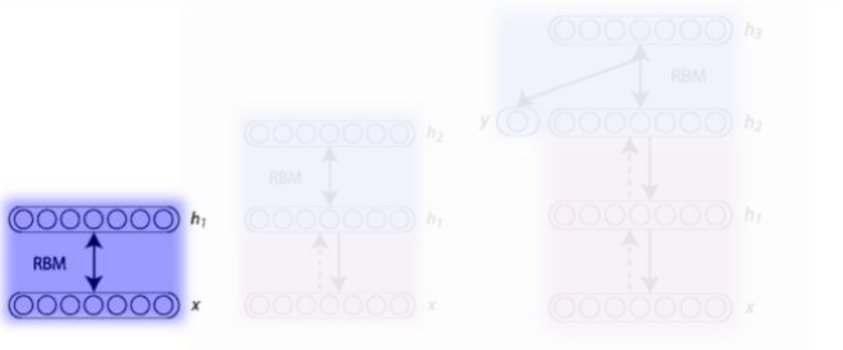
- Fit parameters W_1 of the 1st layer RBM to data (x)
- Freeze W_1 and use samples h_1 as data for the next layer
- Fit parameters W_2 of the 2nd layer RBM to data (h_1)
- Proceed recursively for the next layers



"Diving in Deeper": Deep Belief Networks

Built up by stacking up RBMs:

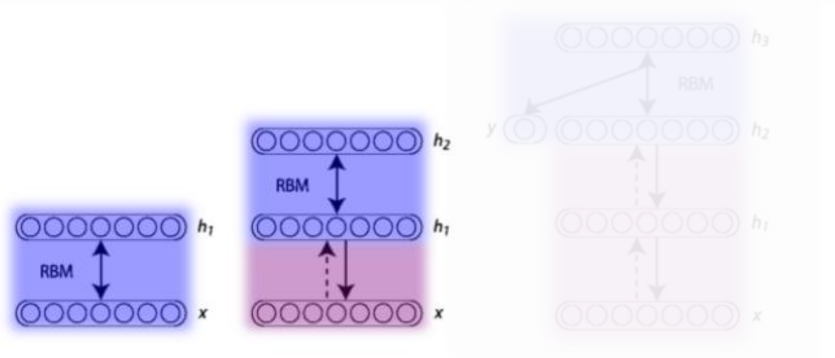
- Fit parameters W_1 of the 1st layer RBM to data (x)
- Freeze W_1 and use samples h_1 as data for the next layer
- Fit parameters W_2 of the 2nd layer RBM to data (h_1)
- Proceed recursively for the next layers



“Diving in Deeper”: Deep Belief Networks

Built up by stacking up RBMs:

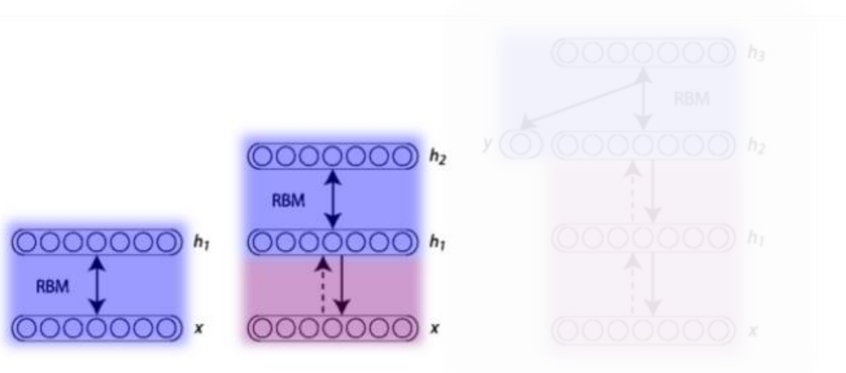
- Fit parameters W_1 of the 1st layer RBM to data (x)
- Freeze W_1 and use samples h_1 as data for the next layer
- Fit parameters W_2 of the 2nd layer RBM to data (h_1)
- Proceed recursively for the next layers



"Diving in Deeper": Deep Belief Networks

Built up by stacking up RBMs:

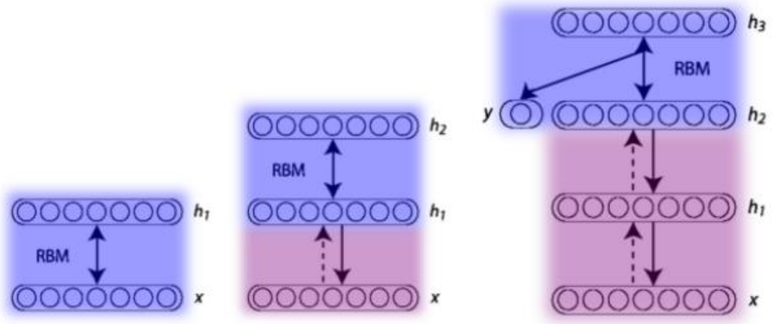
- Fit parameters W_1 of the 1st layer RBM to data (x)
- Freeze W_1 and use samples h_1 as data for the next layer
- Fit parameters W_2 of the 2nd layer RBM to data (h_1)
- Proceed recursively for the next layers



"Diving in Deeper": Deep Belief Networks

Built up by stacking up RBMs:

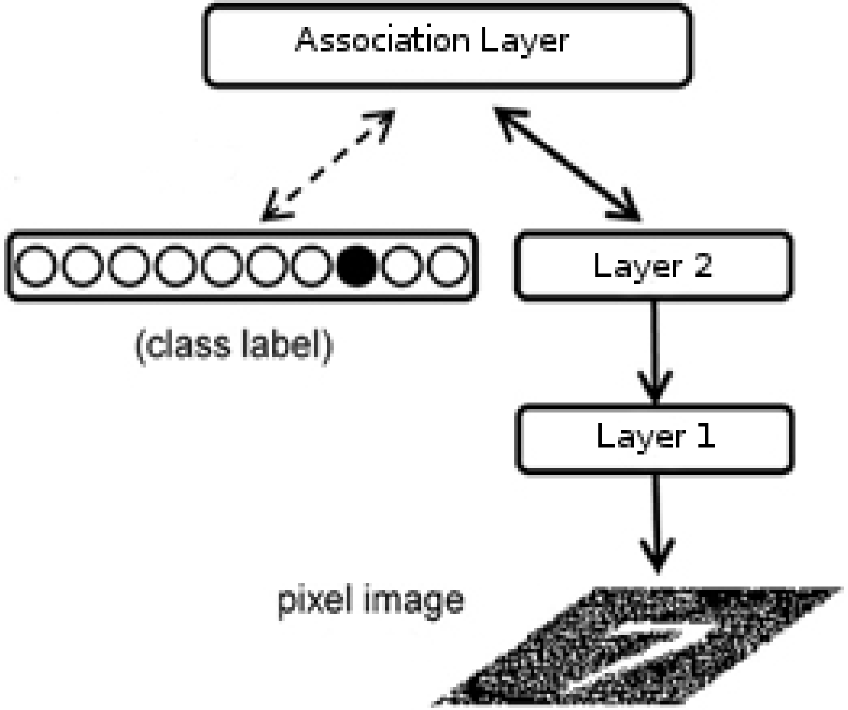
- Fit parameters W_1 of the 1st layer RBM to data (x)
- Freeze W_1 and use samples h_1 as data for the next layer
- Fit parameters W_2 of the 2nd layer RBM to data (h_1)
- Proceed recursively for the next layers



Classification with DBNs

- Top Layer : Joint density for labels and images

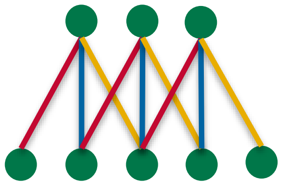
- Bottom Layers : Feature Extraction

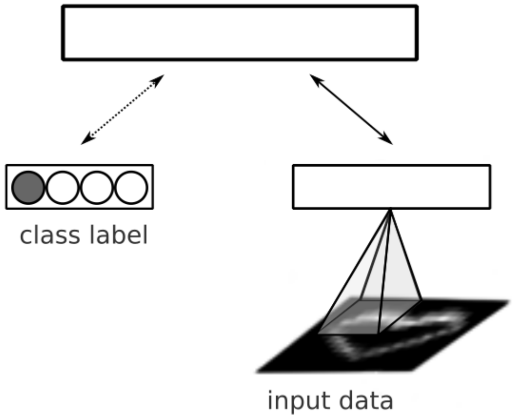


Convolutional DBNs

Using a convolutional architecture:

- Partially connected
- Shared weights

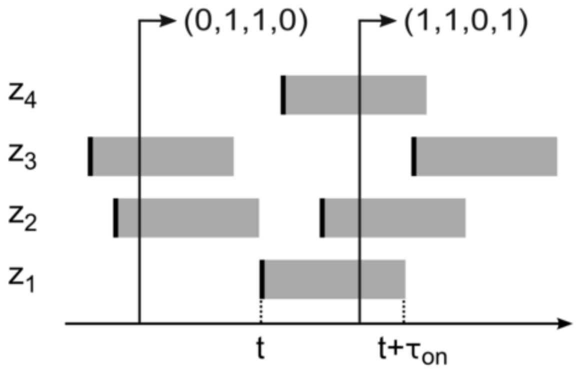




Sampling in Spiking Neural Networks

- State of Neuron defined by its Firing

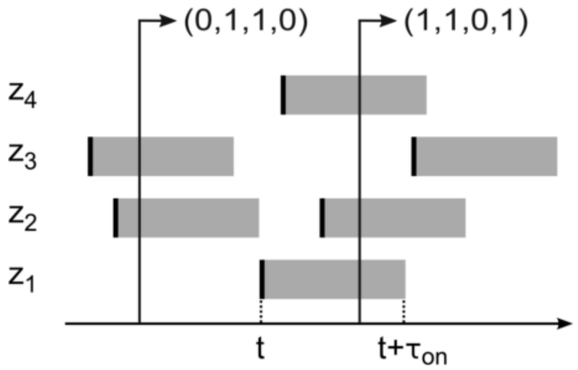
- State of Network defined by Neurons

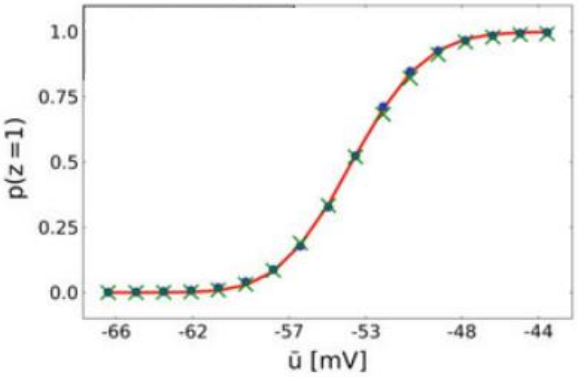


Sampling in Spiking Neural Networks

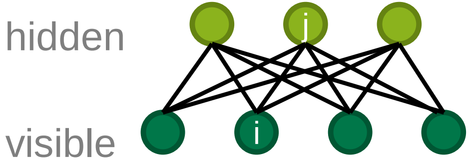
- State of Neuron defined by its Firing
  - Firing probability: $p(\mathrm{x}{=}1) \propto \sigma(Wz)$

- State of Network defined by Neurons





Event-driven Contrastive Divergence

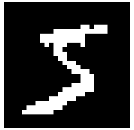


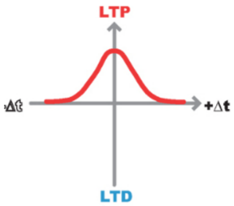
Contrastive Divergence:

- For Spiking Neural Networks
- With Spike-time Dependent (Synaptic) Plasticity (STDP)

Data burn-in

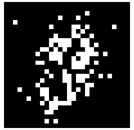
Data distribution

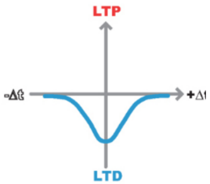




Model burn-in

Model distribution



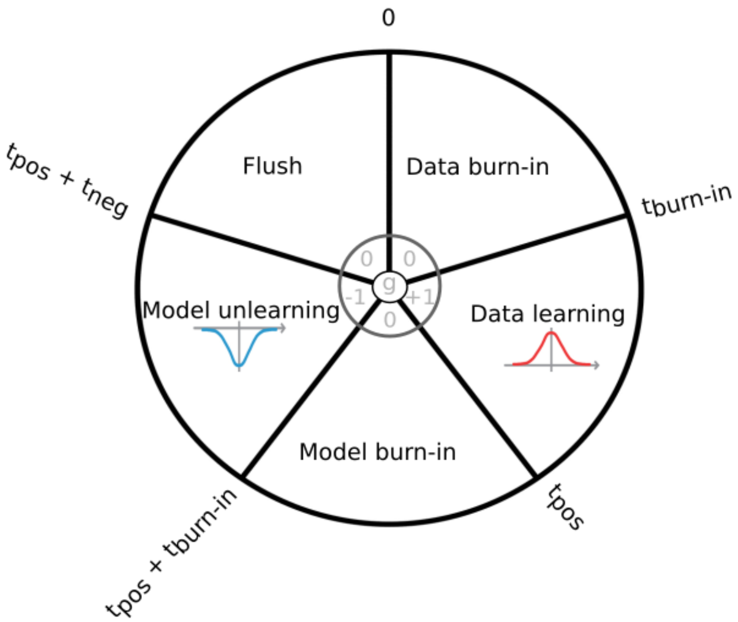


Spiking Convolutional DBN

- Layerwise Training with eCD
- Weight sharing with weight synchronization
- Inhibitory lateral Connections
- Forward Connections between RBMs

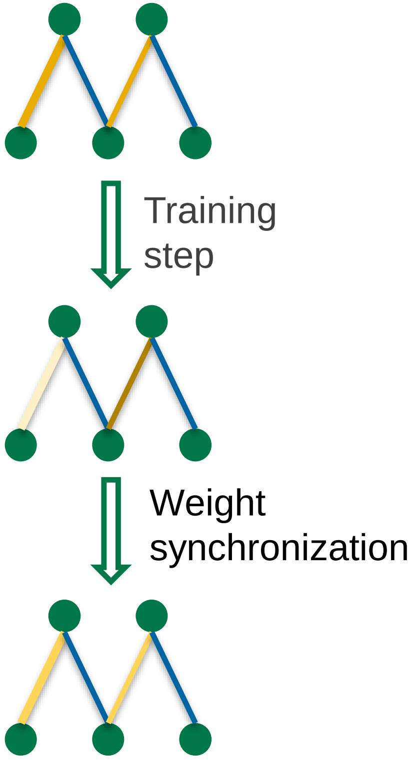
Spiking Convolutional DBN

- Layerwise Training with eCD
- Weight sharing with weight synchronization
- Inhibitory lateral Connections
- Forward Connections between RBMs



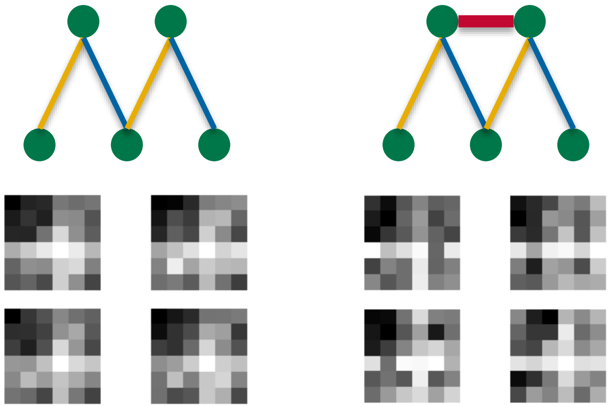
Spiking Convolutional DBN

- Layerwise Training with eCD
- Weight sharing with weight synchronization
- Inhibitory lateral Connections
- Forward Connections between RBMs



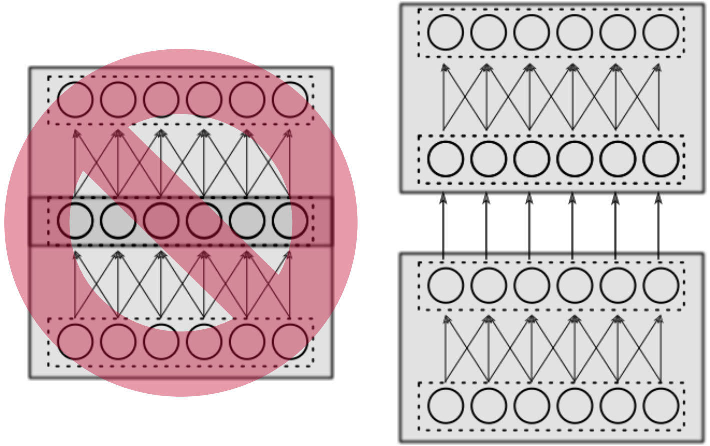
Spiking Convolutional DBN

- Layerwise Training with eCD
- Weight sharing with weight synchronization
- Inhibitory lateral Connections
- Forward Connections between RBMs



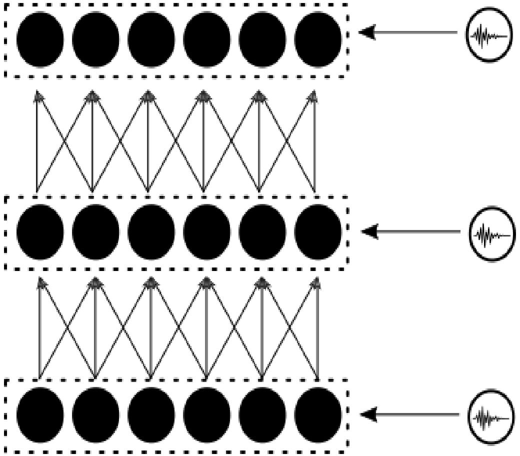
Spiking Convolutional DBN

- Layerwise Training with eCD
- Weight sharing with weight synchronization
- Inhibitory lateral Connections
- Forward Connections between RBMs



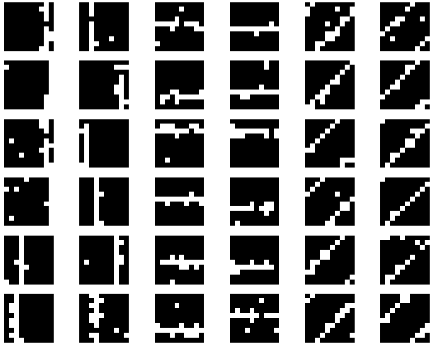
Conversion

- Train a DBN
- Replace Binary Neurons with Spiking Neurons
- Use Synaptic Connections
- Scale Synaptic Weights
- Add external Poisson-Noise



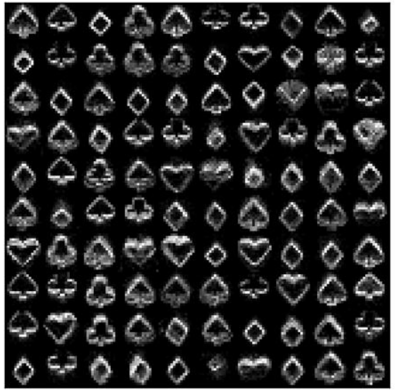
Experiments

Datasets

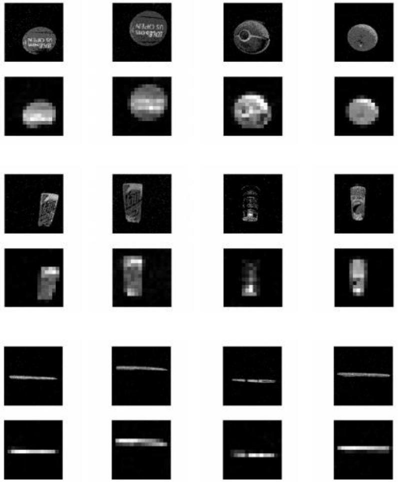


Stripes

10 x 10 Pixel



Poker
(Event-based)
16 x 16 Pixel

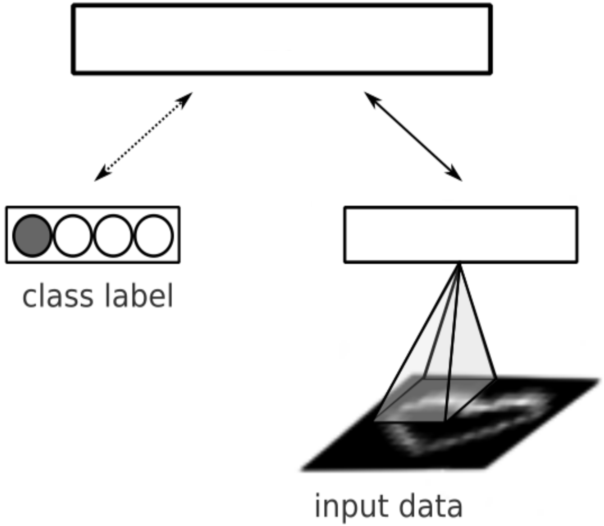


Ball-Can-Pen
(Event-based)
16 x 16 Pixel

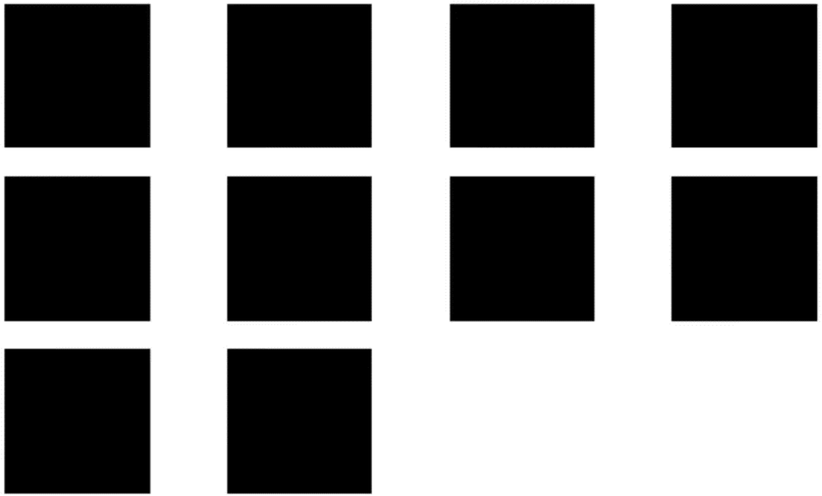
eCD - Results

Accuracy:

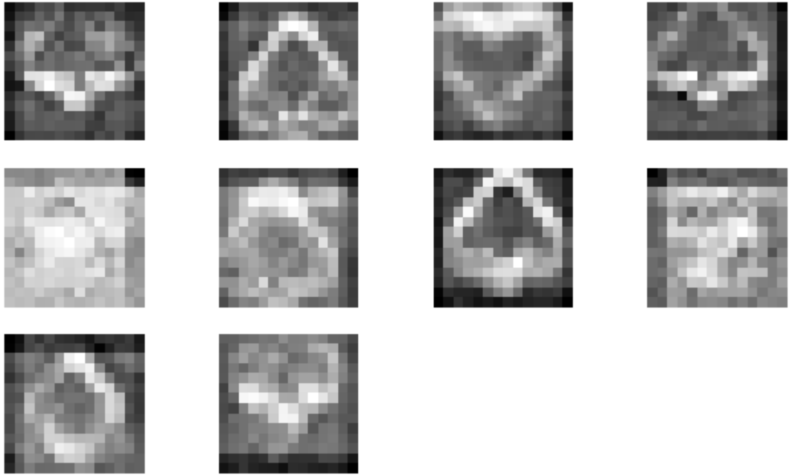
- Stripes: 1.00
- Poker: 0.94
- Ball-Can-Pen: 0.90



eCD - Weights

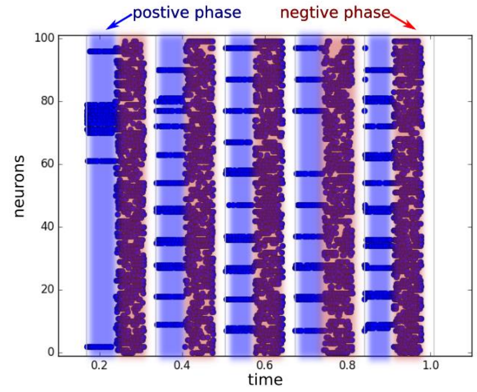


eCD - Weights

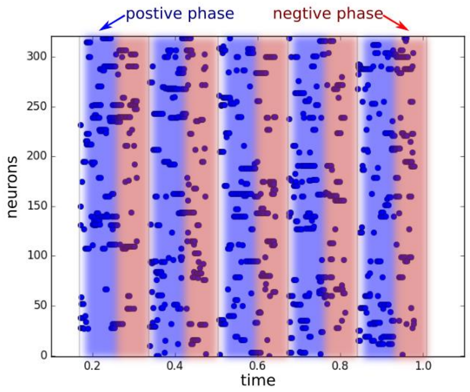


eCD - Spikes

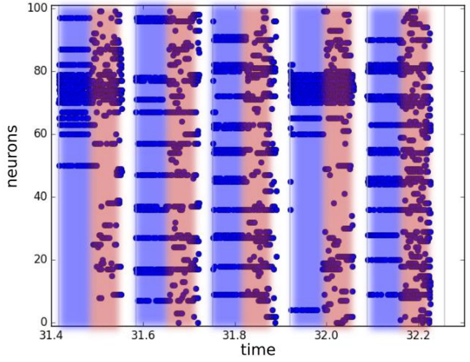
visible

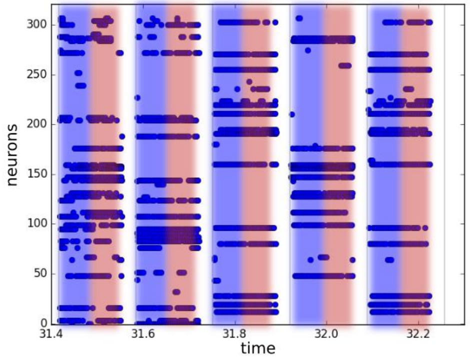


hidden



Training

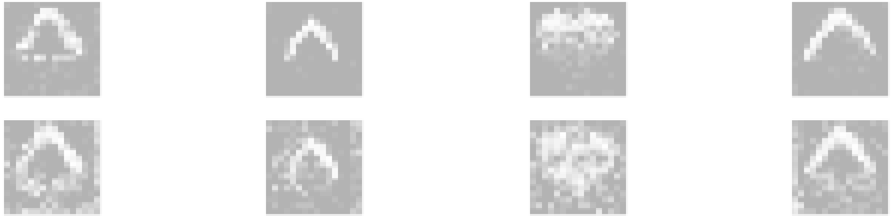




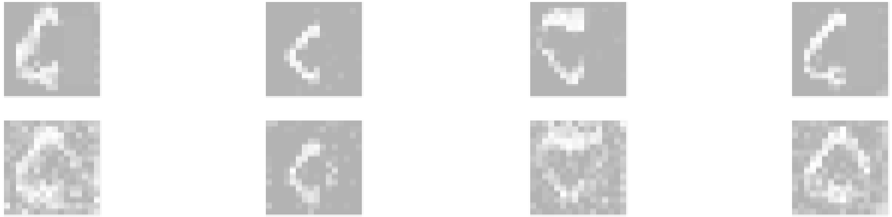
eCD - Reconstruction

Reconstruction

Bottom half missing

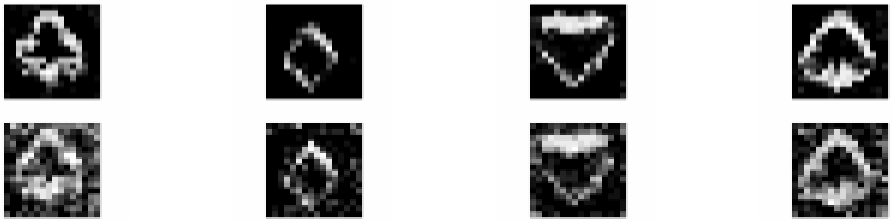


Right half missing

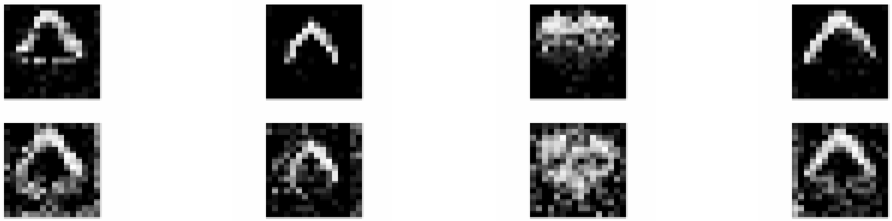


eCD - Reconstruction

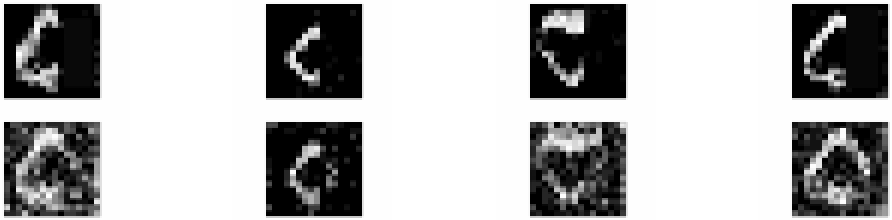
Reconstruction



Bottom half missing



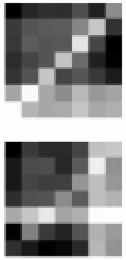
Right half missing



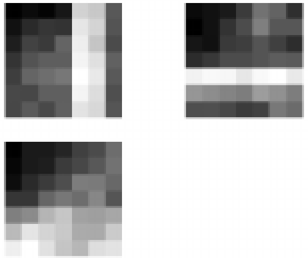
Comparison

Artificial DBN and Spiking DBN trained with 100 Samples

Weights



Converted DBN



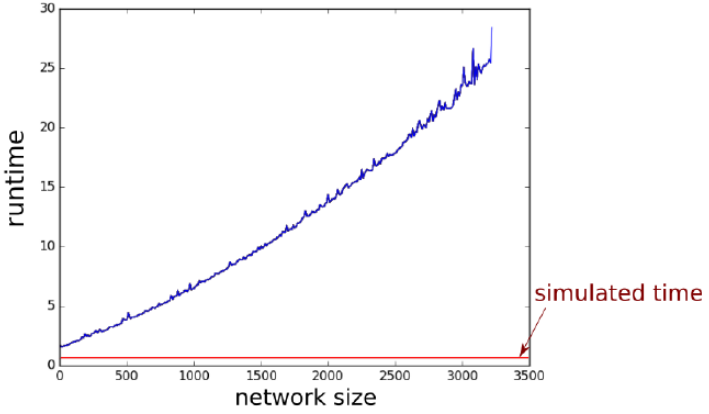
Event-based DBN

Discussion

→ Train a spiking convolutional DBN

+ Unsupervised Learning
+ Spiking Neural Network
+ Event-based
+ Biological Plausibility

- Computational Resources



Thanks !

“We have truly autonomous cars when you tell it to drive to the office, and it decides to drive to the beach.”

“Geoff Hinton doesn't disagree with you, he contrastively diverges.”

“Geoff Hinton discovered how the brain really works. Once a year for the last 25 years.”

References

Lecun, Yann; Predictive Learning; NIPS 2016 , https://drive.google.com/file/d/0BxKBnD5y2M8NREZod0tVdW5FLTQ/view

Kokkinos, Iasonas; Introduction to Deep Learning; http://cvn.ecp.fr/personnel/iasonas/course/DL5.pdf

Larochelle, Hugo; Neural networks: RBM - CD; http://info.usherbrooke.ca/hlarochelle/neural_networks/content.html

Lamblin, Pascal; Deep belief networks; http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/DeepBeliefNetworks

M. Zorzi, A. Testolin, and I. Stoianov. Modeling language and cognition with deep unsupervised learning: a tutorial overview.

L. Buesing, J. Bill, B. Nessler, and W. Maass. Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons.

M. A. Petrovici. Form Versus Function: Theory and Models for Neuronal Substrates. 2016.

K. A. Buchanan and J. R. Mellor. The activity requirements for spike timing-dependent plasticity in the hippocampus.

I. Higgins. Early Visual Concept Learning with Unsupervised Deep Learning. 2016

* Actually Bernoulli

Appendix

$$C_m \frac{\partial u}{\partial t} = g_l(E_l - u(t)) + I^{syn} + I^{ext}$$

Resting potential	-65 mV
Membrane capacity	1.0 nF
Membrane time constant	20.0 ms
Refractory period	10.0 ms
Offset current	1.0 nA
Reset potential	-53.0 mV
Spike threshold	-52.0 mV
Inhibitory reversal potential	90.0 mV
Excitatory reversal potential	-0.0 mV

Appendix

$$STDP(v_i(t), h_j(t)) = v_i(t)A_{h_j}(t) + h_j(t)A_{v_i}(t),$$

$$A_{h_j}(t) = A\int_{-\infty}^{t} W(t-s)h_j(s)ds,$$

$$A_{v_i}(t) = A\int_{-\infty}^{t} W(t-s)v_i(s)ds.$$

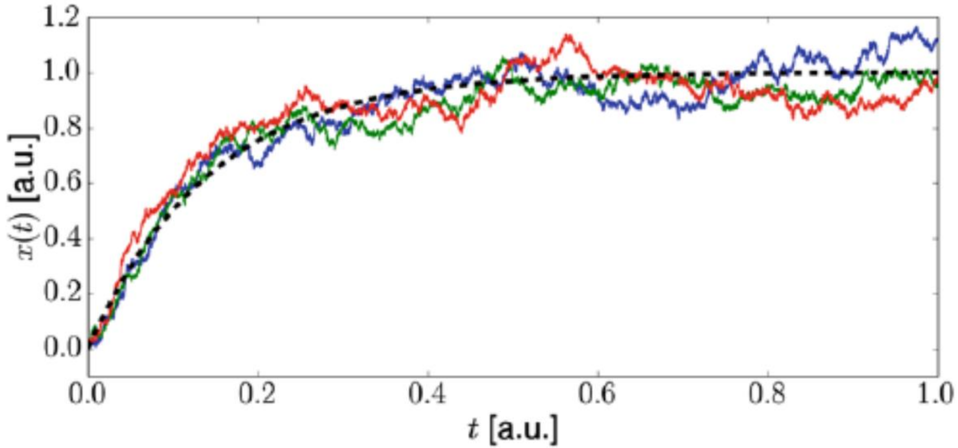
$$W(x) = exp(\tfrac{x}{\tau}).$$

$$A_v = A_v \exp(\frac{-\Delta t}{\tau}) + a_\delta,$$

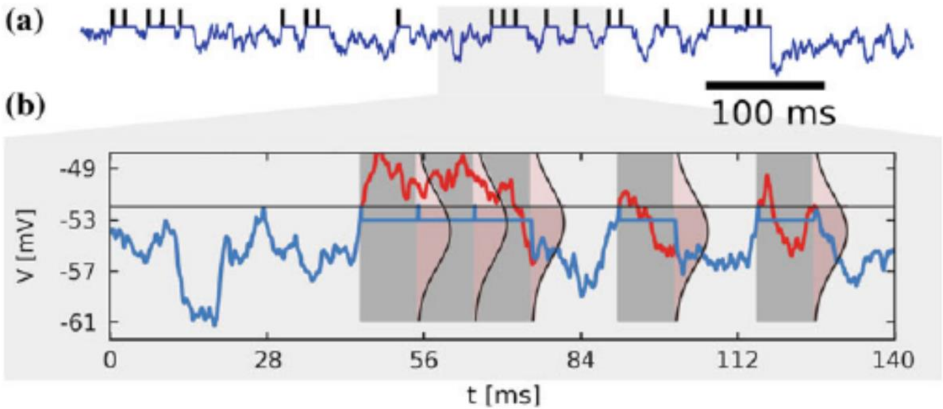
$$A_h = A_h \exp(\frac{-\Delta t}{\tau}),$$

$$\delta w = \mu g(t) A_v,$$

Appendix

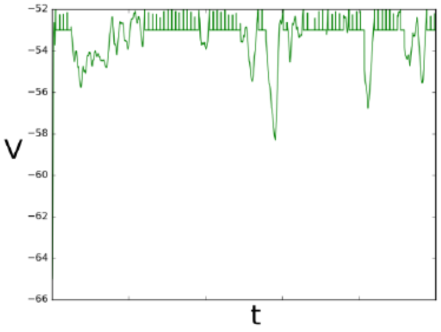


1.2
1.0
0.8
0.6
0.4
0.2
0.0
$x(t)$ [a.u.]
0.0
0.2
0.4
0.6
0.8
1.0
t [a.u.]

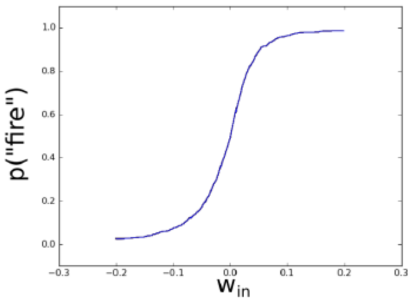


(a)
(b)
100 ms
-49
-53
-57
-61
V [mV]
0
28
56
84
112
140
t [ms]

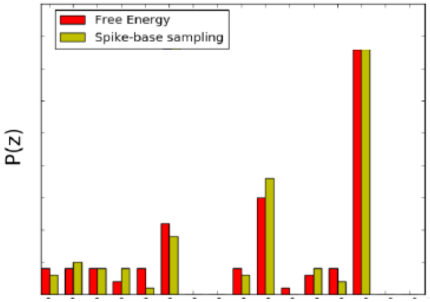
Appendix



−52
−54
−56
−58
−60
−62
−64
−66
V
t

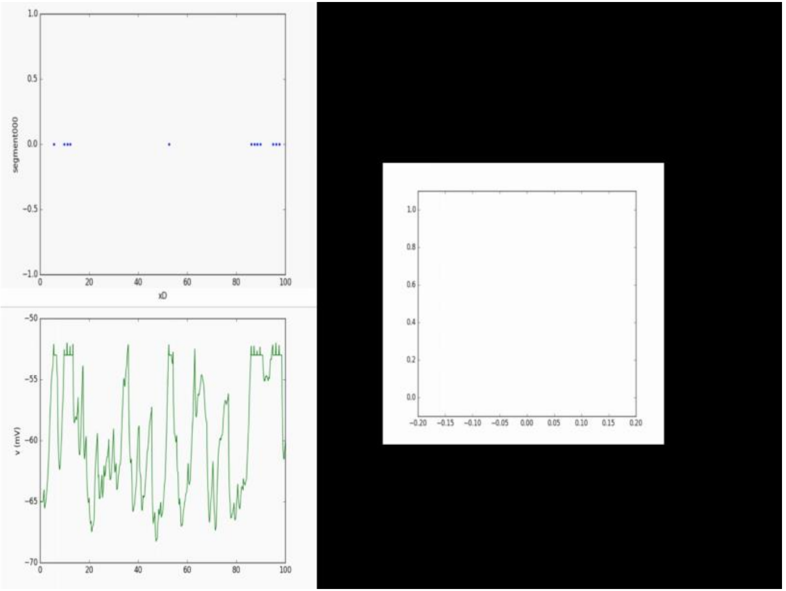


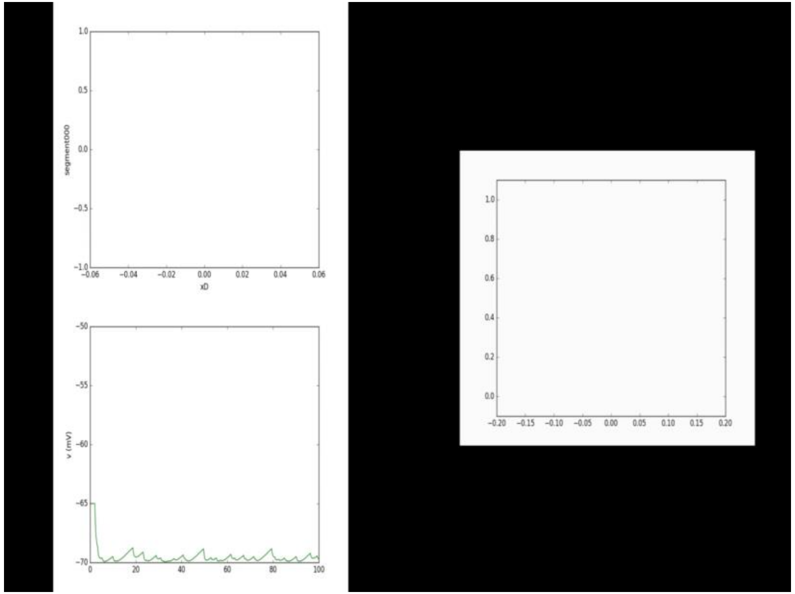
1.0
0.8
0.6
0.4
0.2
0.0
p("fire")
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
w_{in}



Free Energy
Spike-base sampling
P(z)

Appendix

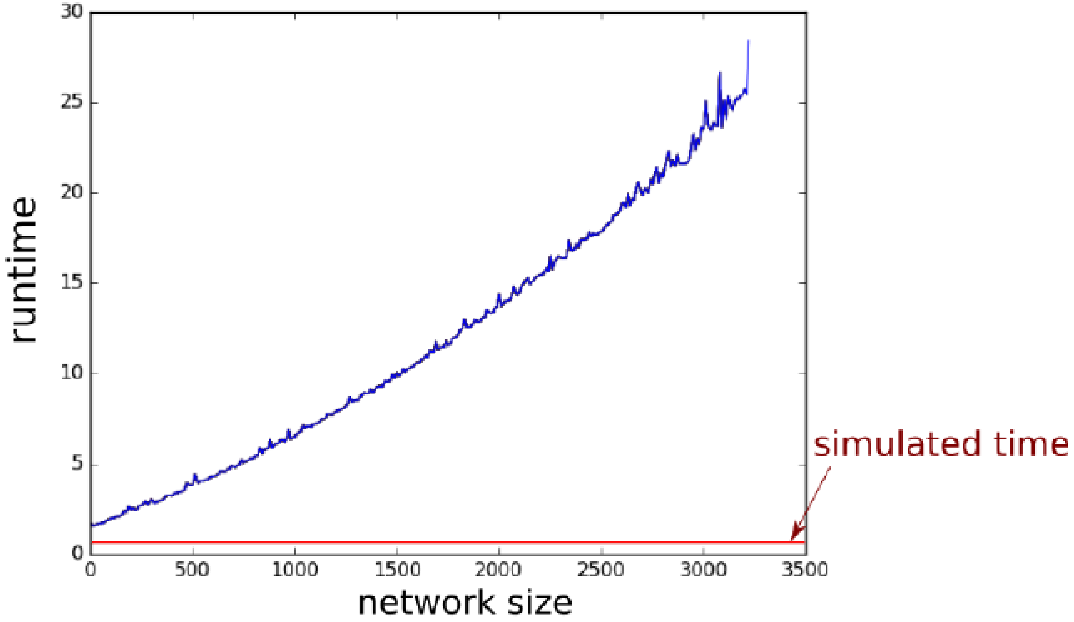




Appendix

	Spiking CNN		CUBA LIF DBN	
Simulated time	Classification Accuracy	Runtime	Classification Accuracy	Runtime
50 ms	0.69	7.8 s	0.81	8.1 s
100 ms	0.77	9.2 s	0.89	10.5 s
200 ms	0.76	13.1 s	0.89	14.6 s
300 ms	0.75	15.1 s	0.91	18.5 s
500 ms	0.83	24.2 s	0.93	30.6 s

Appendix


30
25
20
15
10
5
0
runtime
0
500
1000
1500
2000
2500
3000
3500
network size
simulated time

Appendix

$t_{burn-in}$	14 ms
t_{learn}	56 ms
t_{flush}	28 ms
Learn-rate	1.0
Weight-decay	0.001
Weight synchronization after n samples	1

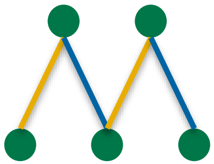
Appendix

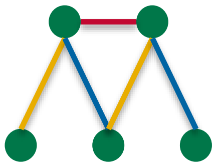
	Stripes	Poker	Ball-Pen-Can
Input	100	256	256
#1. Layer Params	20 x 7 x 7 = 980	10 x 14 x 14 = 1960	20 x 14 x 14 = 3920
1. Layer	20 x 4 x 4 = 320	10 x 3 x 3 = 90	20 x 3 x 3 = 180
#2. Layer Params	(320 + 3) x 20 = 6460	(90 + 4) x 10 = 940	(180 + 4) x 10 = 1840
2. Layer	20	10	10
Labels	3	4	4

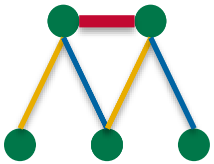
Appendix

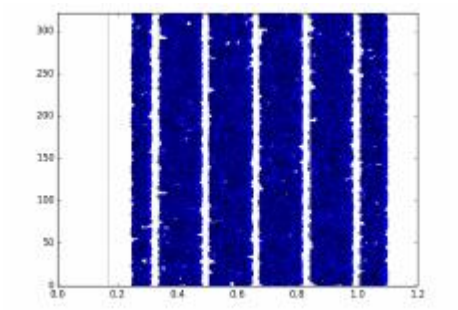
	Stripes	Poker	Ball-Pen-Can
Neurons	443	360	450
Synapses	22140	18580	37120
Parameters	7440	2900	5760

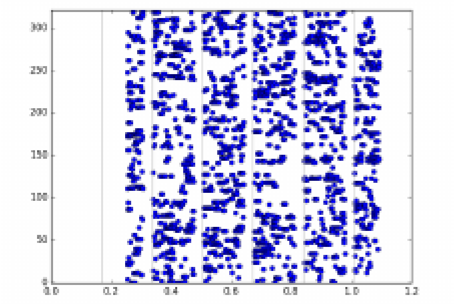
Appendix

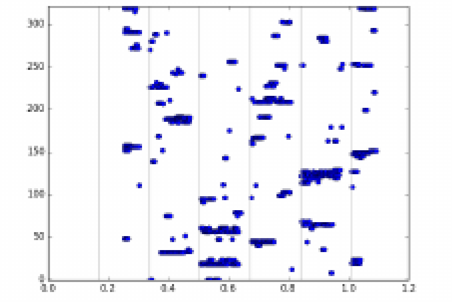




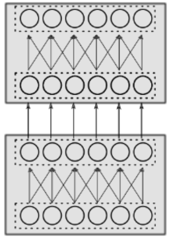


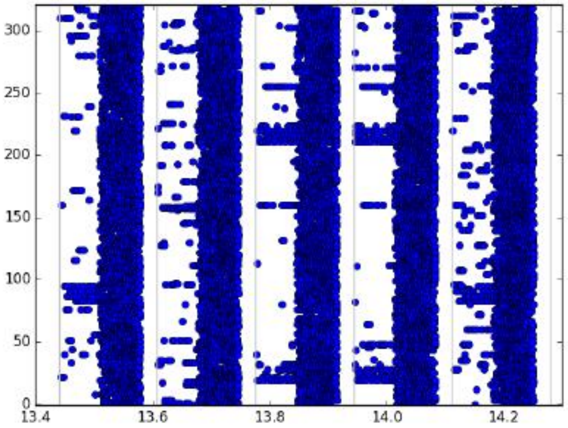




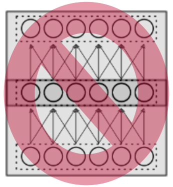


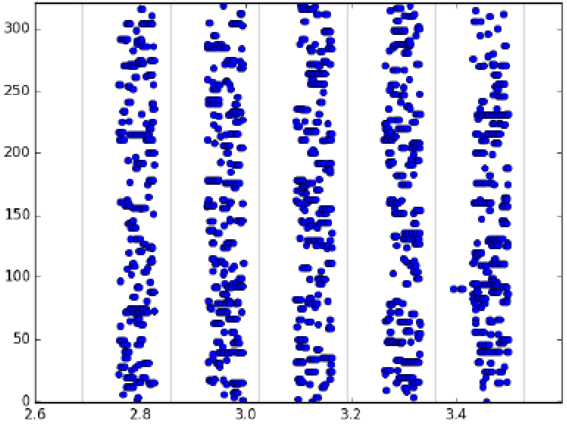
Appendix





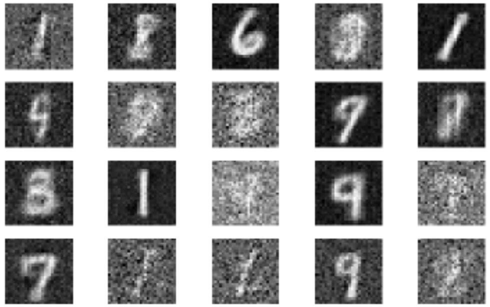
300
250
200
150
100
50
0
13.4
13.6
13.8
14.0
14.2

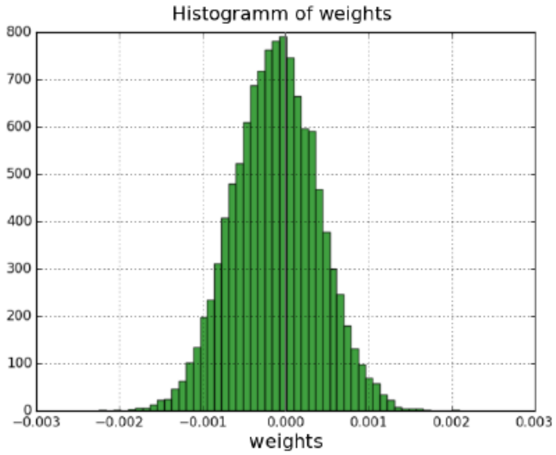




300
250
200
150
100
50
0
2.6
2.8
3.0
3.2
3.4

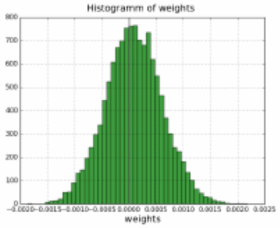
Appendix



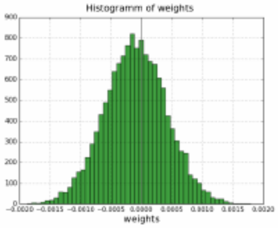


Histogramm of weights
800
700
600
500
400
300
200
100
0
−0.003
−0.002
−0.001
0.000
0.001
0.002
0.003
weights

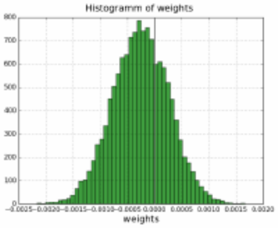
Histogramm of weights
weights



Histogramm of weights
weights



Histogramm of weights
weights



Histogramm of weights
weights

Appendix

Yann LeCun @ylecun · 54 Min.

Geoff Hinton said we were the lunatic fringe & became the lunatic core.
Hope my multilayered cake satisfied the most gourmet LeCunatics.

Naomi Saphra @nsaphra

Lecture hall packed with Edinburgh's LeCunatics waiting to hear @ylecun expound on his cake recipe.

Original (Englisch) übersetzen

2