Input normalization and synaptic scaling – two sides of the same coin

Computational and Biological Learning
UNIVERSITY OF CAMBRIDGE
FIAS Frankfurt Institute for Advanced Studies

Christian Keck[1], Cristina Savin[1,2], Jörg Lücke[1]
[1]FIAS, Universität Frankfurt, Germany; [2]CBL, University of Cambridge, UK

DFG
Bernstein Focus: Neurotechnology Frankfurt

Introduction

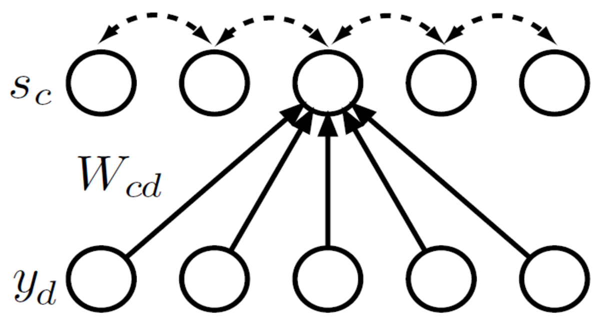
In neural network models:
- input normalization commonly used e.g. for contrast invariance
- weight normalization needed to stabilize Hebbian learning

In neural circuits:
- input normalization implemented through feedforward inhibition [1]
- weight normalization by homeostatic synaptic scaling [2]

Questions:
- could they be more closely related than the separate investigations suggest?
- what functional purpose may this relation have?

Neural network model

2-layer neural network



$$s_c = \frac{\exp(I_c)}{\sum_{c'}\exp(I_{c'})},\; I_c = \sum_d W_{cd}\,y_d \quad \text{softmax}$$

$$\Delta W_{cd} = \epsilon\,(s_c\,y_d - s_c\,W_{cd}) \quad \text{learning rule}$$

$$y_d = (A-D)\frac{\tilde{y}_d}{\sum_{d'}\tilde{y}_{d'}} + 1 \quad \text{input normalization}$$

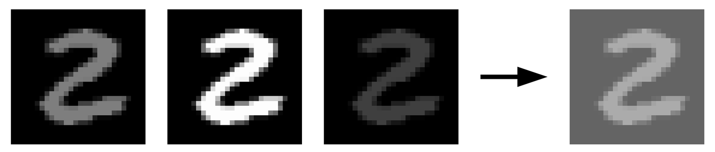


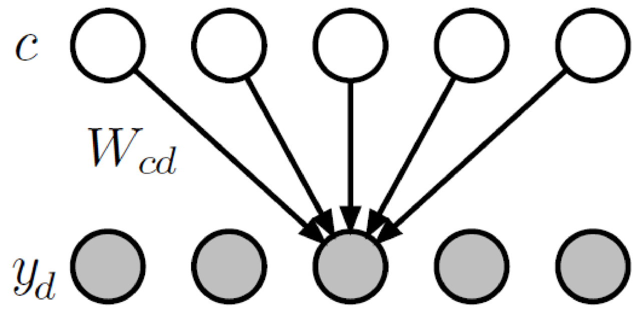
Illustration: input normalization

Network properties:
- functionality: clustering
- lateral competition (softmax)

Generative model

Mixture model with Poisson noise

D dimensions $1\dots d$
C classes $1\dots c$



$$p(c) = \frac{1}{C} \quad \text{prior}$$

$$\sum_d W_{cd} = A \quad \text{normalization constraint}$$

$$p(\vec{y}\,|\,c, W) = \prod_{d=1}^{D} \text{Poisson}(y_d;\, W_{cd})$$

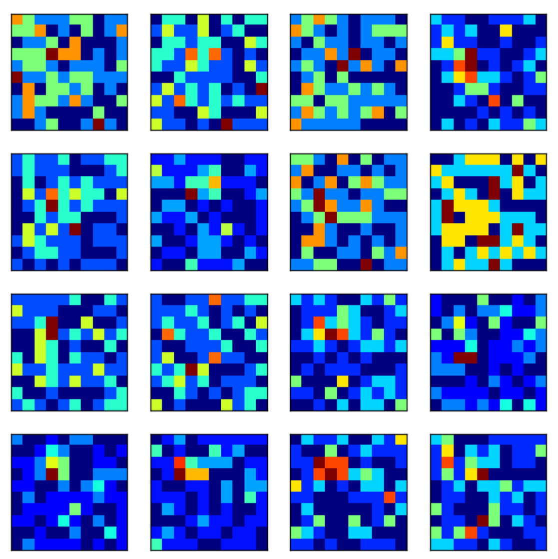
E-step: $$p(c|\vec{y}^{(n)}, W) = \frac{\exp(I_c)}{\sum_{c'}\exp(I_{c'})}, \text{ where } I_c = \sum_d y_d \ln(W_{cd})$$

M-step: $$W_{cd} = A\frac{\sum_n p(c|\vec{y}^{(n)}, W^{(old)})\,y_d^{(n)}}{\sum_d \sum_n p(c|\vec{y}^{(n)}, W^{(old)})\,y_d^{(n)}} \quad \text{update rule}$$

Results of numerical simulations

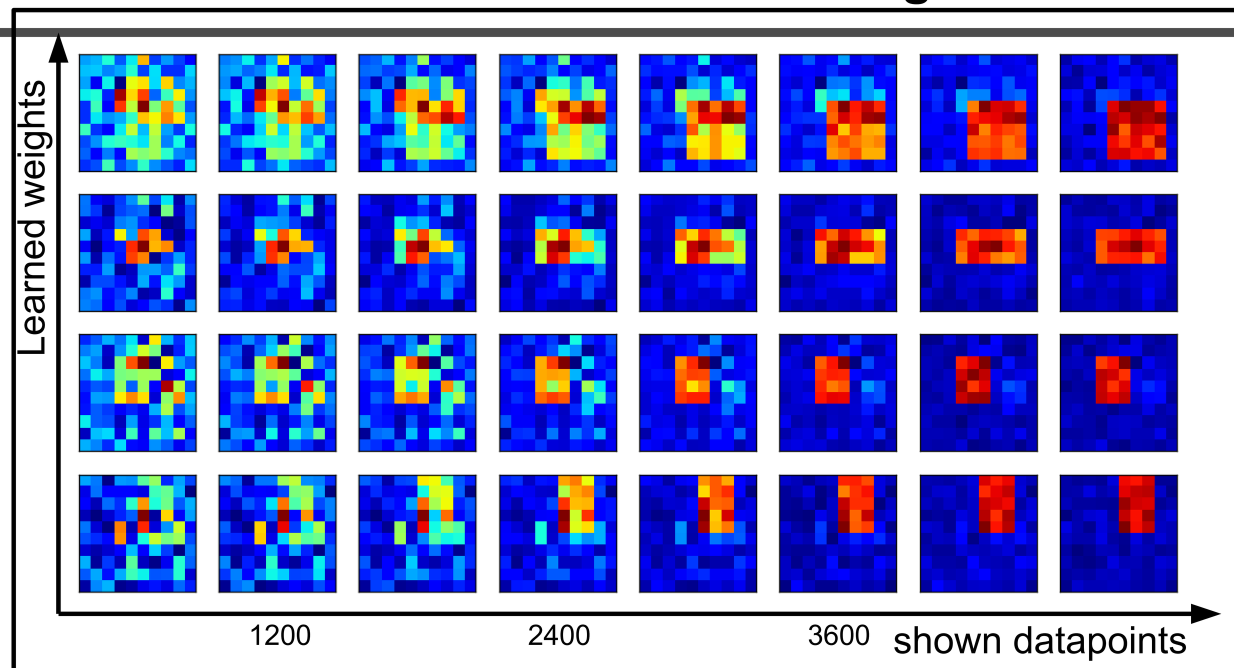
Artificial data

Data

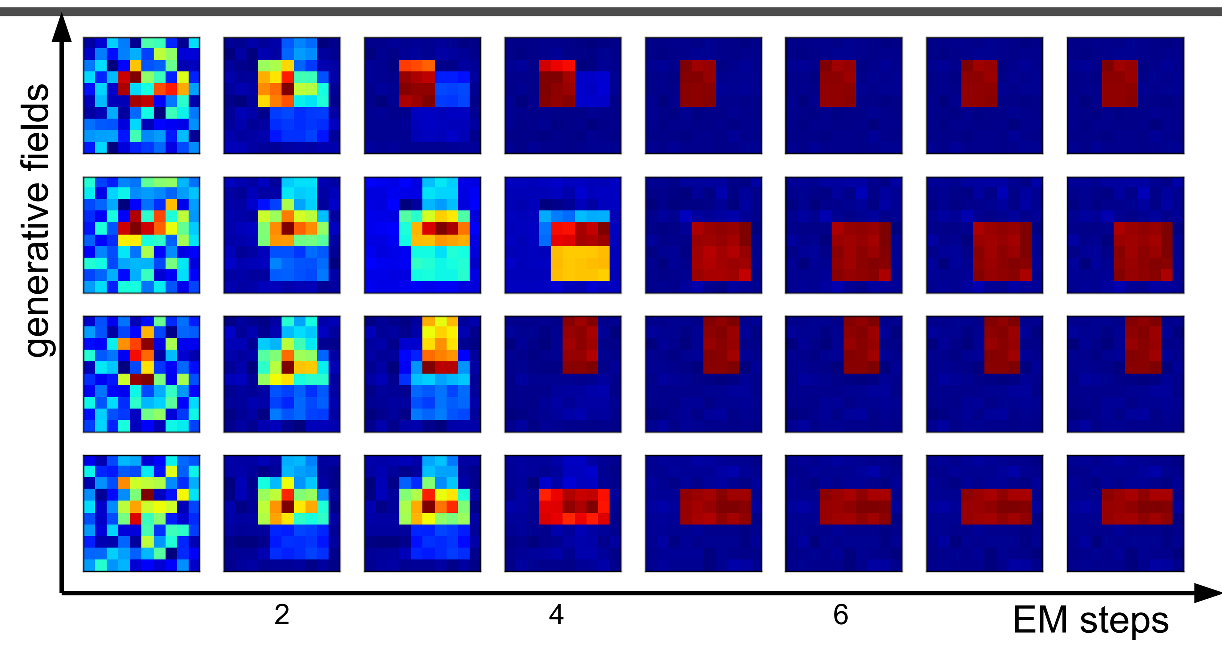


Examples of generated artificial data with strong Poisson noise (for small A)

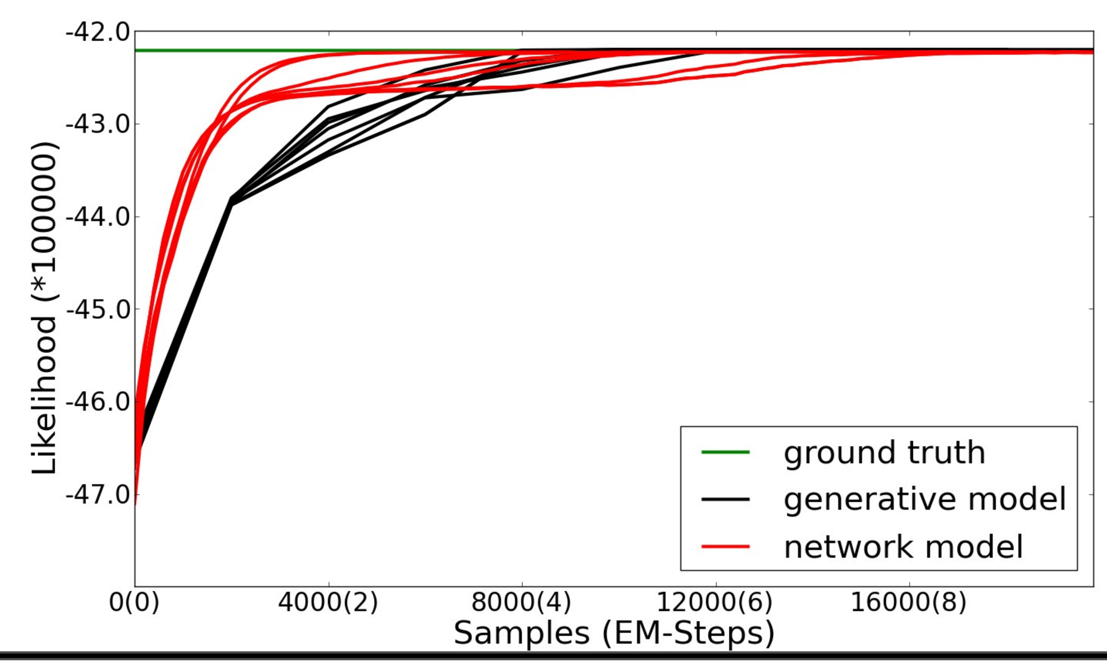
Neural network: learned weights



Mixture model: learned generative fields

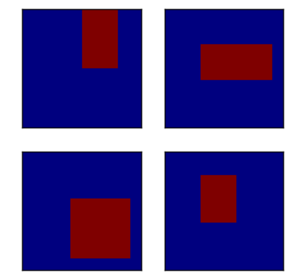


Likelihood comparison

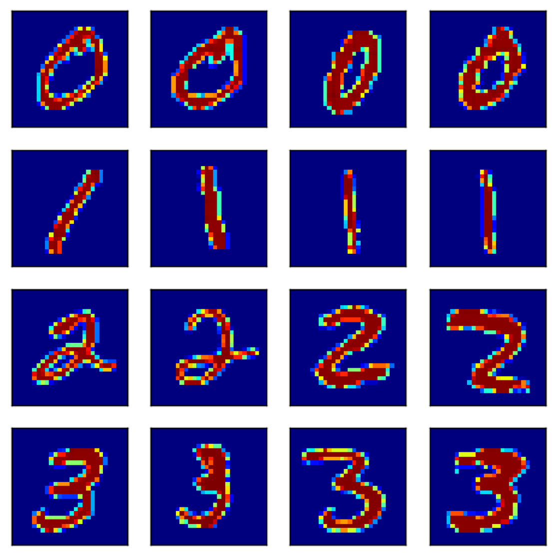


Causes

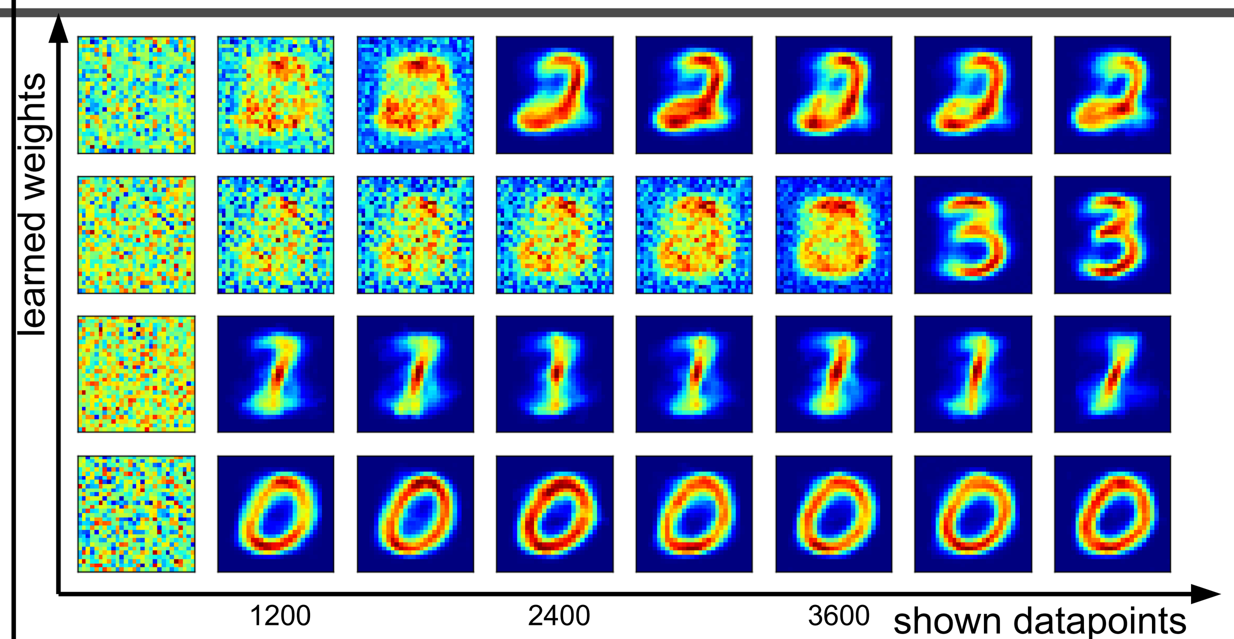
Generated data:
- rectangles
- overlap

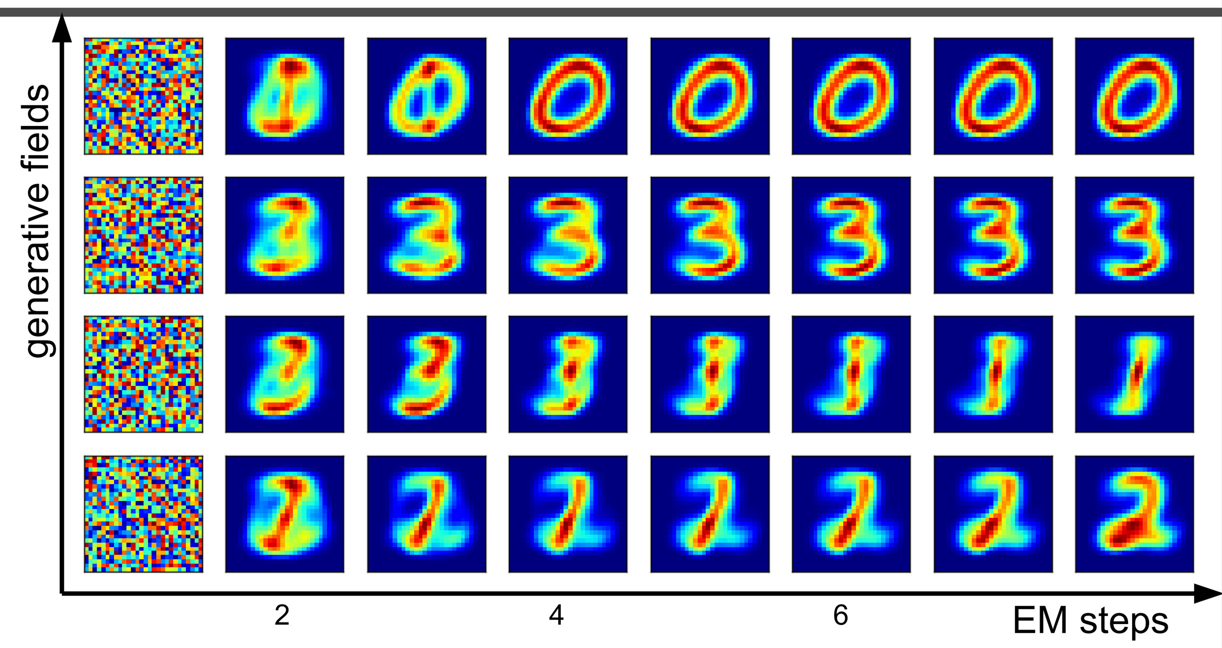


MNIST digits



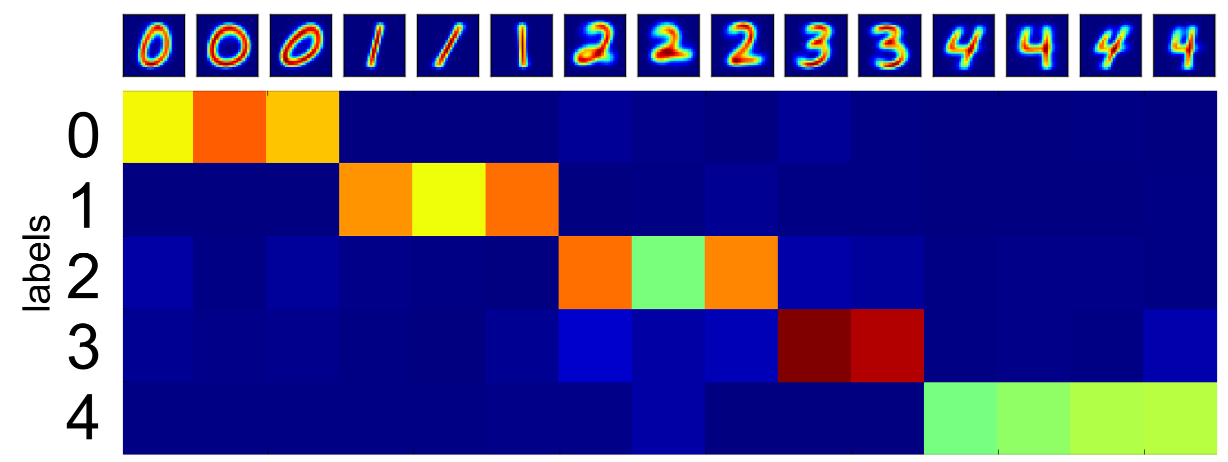
Examples of handwritten digits from MNIST database





Labelling generative fields for classification

- we label learned generative fields using few data points (less than 10%)
- an overcomplete setup allows for learning different styles of digits
- performance increases with overcompleteness (97% correct for 5x overcomplete)



Probabilistic labels: distribution of responses of fields given a label

Discussion

- neural network with homeostatic synaptic scaling and feedforward inhibition learns optimal parameters in mixture model
- synaptic scaling mirrors the normalization of input patterns (the weights „follow“ the input)
- simplified learning on constraint space → input normalization and synaptic scaling could generally facilitate learning in neural circuits

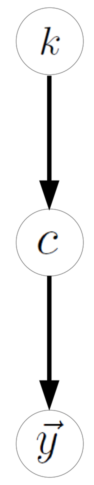
References

[1] Pouille at al, Input normalization by global feedforward inhibition expands cortical dynamic range, Nat.Neurosc. 12, 2009

[2] Turrigiano and Nelson, Homeostatic plasticity in the developing nervous system, Nat Rev Neurosc 5, 2004

We gratefully acknowledge funding by grants BMBF 01GQ0840 (BFNT Frankfurt) and DFG LU 1196/4-1.

Classification details



Interpreted as a graphical model:
- datapoint $\vec{y}$ is generated by cause c
- cause c is generated by label k

$$B_{ck} = \frac{1}{M}\sum_{m=1}^{M} p(c\,|\,\vec{y}^{(m)}, \Theta) \quad \text{where } \vec{y}^{(m)} \text{ is sampled from } p(\vec{y}\,|\,k)$$

$$p(k\,|\,\vec{y}, \Theta) = \frac{\sum_c B_{ck}\,p(\vec{y}\,|\,c, \Theta)}{\sum_{k'}\sum_{c'} B_{c'k'}\,p(\vec{y}\,|\,c', \Theta)}$$