Learning with incomplete information on the Committee Machine

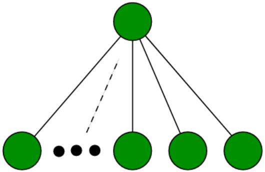
Urs Bergmann

FIAS
FRANKFURT INSTITUTE
FOR ADVANCED STUDIES

Heidelberg, Δ Meeting, December 2007

1. Introduction
   - Network Architectures
2. Learning
   - Introduction
   - Generalization Error
   - Exemplified Analysis
3. Learning with Incomplete Information
   - Introduction
   - Coarse Graining
4. Results
   - Perceptron
   - Committee Machine
5. Conclusion

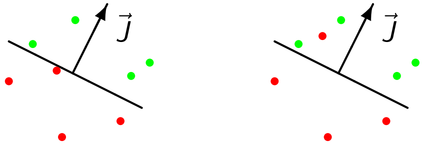
Perceptron



The ‘elementary particle’ of perception.

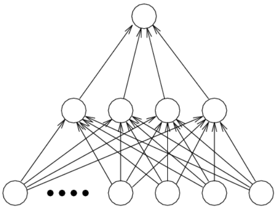
- N dimensional Ising input $\vec{\xi} \in \{-1, 1\}^N$
- adaptive weightvector $\vec{J}$
- Network response: $s = \mathrm{g}(\vec{J} \cdot \vec{\xi}/\sqrt{N})$
  where g is a sigmoidal function.

Why more complex?



- The illustration shows a linear separable function (left) and a non-linear separable function (right).
- A single layered network $\mathrm{g}(\vec{J}\vec{\xi}) = \mathrm{sign}\left(\vec{J}\vec{\xi}\right)$ can only implement linear separable functions.
- $\rightarrow$ more layers needed.

Committee Machine[1]



- N dimensional Ising input $\vec{\xi} \in \{-1, 1\}^N$
- K weightvectors $\vec{J}_k$ for all 'hidden' units
- K weights w_k at output unit
- Output of the network:

$$s(\vec{\xi}) = \mathrm{f}\left(\frac{1}{\sqrt{K}} \sum_{k=1}^{K} w_k \, \mathrm{g}(\vec{J}_k \vec{\xi} / \sqrt{N})\right)$$

[1] [*Saad*95]

Committee Machine (2)

- for K unrestricted, the network is an *universal approximator* [*Cybenko*89]
- soft-committee machine
  - $\mathrm{f} = \mathrm{id}$
  - output weights $w_k = 1$
  - Def. field: $x_k = \vec{J}_k \vec{\xi} / \sqrt{N}$
  - $\rightarrow s(\vec{\xi}) = \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \mathrm{g}(x_k)$

Unsupervised and Supervised Learning

- Unsupervised learning
  - no answers of a goal function given
  - information only in the correlations of the input
  - $\longrightarrow$ categorization of the input patterns

Unsupervised and Supervised Learning

- Unsupervised learning
  - no answers of a goal function given
  - information only in the correlations of the input
  - $\rightarrow$ categorization of the input patterns
- Supervised learning
  - learning based on examples / correct answers
  - here: consider a teacher network with output $t(\vec{\xi})$ and M hidden units
  - for the Committee Machine we have 3 different cases:
    - $K = M$ exactly learnable
    - $K < M$ unlearnable
    - $K > M$ overlearnable

Reinforcement Learning[2]

- instead of examples, only a reward signal is given
- this signal may be unspecific in time
- this unspecificity causes a credit assignment problem
- more plausible biologically
- RL 'in between' unsupervised and supervised learning

[2] [*Hertz*91, *Sutton*98]

Hebbian Type Learning

- General Hebbian learning rule:

$$\vec{J}^{\mu} = \vec{J}^{\mu-1} + \frac{1}{N} I(s^{\mu}, t^{\mu}) \vec{\xi}^{\mu} t^{\mu}$$

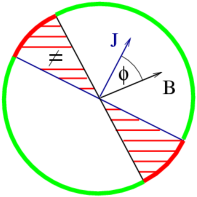
- Supervised Hebbian learning: $I(s^{\mu}, t^{\mu}) = 1$
- Rosenblatts Perceptron learning: $I(s^{\mu}, t^{\mu}) = \Theta(-s^{\mu} t^{\mu})$

Generalization Error

- **Def.**: $\varepsilon^{\mu}(\vec{J}) = \frac{1}{4}\left[s^{\mu} - t^{\mu}\right]^2$, t denoting the goal function.
- theoretical analysis of the network: average error over probability density $P(\xi)$ of input patterns.
- $\varepsilon_g = \langle \varepsilon(\vec{J}, \vec{\xi}) \rangle_{P(\vec{\xi})}$
- in the following: isotropic density $P(\vec{\xi})$
    - $\langle \xi_i \rangle = 0, \quad \langle \xi_i \xi_j \rangle = \delta_{ij}$

One-Layered Geometrical Solution

- Project to the $(\vec{J}, \vec{B})$ plane:



- Error probability: $\varepsilon_g = \phi/\pi = \frac{1}{\pi}\arccos(\frac{\vec{J}.\vec{B}}{J.B})$
- for $Q = \vec{J}\cdot\vec{J},\ R = \vec{J}\cdot\vec{B},\ T = \vec{B}\cdot\vec{B}$ gilt:

$$\varepsilon_g = \frac{1}{\pi}\arccos(\frac{R}{\sqrt{QT}})$$

Committee Generalization Error

- Plug in the definition of the soft-committee machine:
$$\varepsilon(\mathbf{x},\mathbf{y}) = \frac{1}{4}\left(\frac{1}{\sqrt{K}}\sum_{k=1}^{K}\mathrm{g}(x_k) - \frac{1}{\sqrt{M}}\sum_{m=1}^{M}\mathrm{g}(y_m)\right)^2$$
- Generalization Error: $\varepsilon_g(\vec{J}) = \langle \epsilon(\vec{J},\vec{\xi})\rangle_{\{\vec{\xi}\}}$
- Integration yields:
  - $\mathrm{g}(x) = \mathrm{sign}(x)$:

$$\begin{aligned}\varepsilon_g = \quad & \frac{K}{4} - \frac{1}{2\pi K} && \sum_i^K\sum_j^K \arccos\left(\frac{Q_{ij}}{\sqrt{Q_{ii}Q_{jj}}}\right)\\ & +\frac{M}{4} - \frac{1}{2\pi M} && \sum_m^K\sum_n^K \arccos\left(\frac{T_{mn}}{\sqrt{T_{mm}T_{nn}}}\right)\\ & -\frac{\sqrt{KM}}{2} + \frac{1}{\pi\sqrt{KM}} && \sum_i^K\sum_n^M \arccos\left(\frac{R_{in}}{\sqrt{Q_{ii}T_{nn}}}\right). \end{aligned} \tag{1}$$

Supervised Hebbian Learning

- Learing rule: $\vec{J}^{\mu} = \vec{J}^{\mu-1} + \frac{1}{N}\vec{\xi}^{\mu} t^{\mu}$
- Thermodynamic limit yields gaussian variables x and y.
- Therefore using a continuous time limit, we can rewrite the equation in the order parameters:

$$dR/d\alpha = \sqrt{\frac{2}{\pi}}, \quad dQ/d\alpha = 2\sqrt{\frac{2}{\pi}}R(\alpha) + 1$$

- $\varepsilon_g = \frac{1}{\pi}\arccos\left[(1 + \frac{\pi}{2\alpha})^{-1/2}\right]$
- Asymptotics ($\alpha \to \infty$): $\varepsilon_g \approx \frac{1}{\sqrt{2\pi}}\alpha^{-1/2}$

Rosenblatt Perceptron Algorithm

- Learning rule: $\vec{J}^{\mu} = \vec{J}^{\mu-1} + \frac{1}{N}\Theta(-s^{\mu}t^{\mu})\vec{\xi}^{\mu}t^{\mu}$
- with $\rho(\alpha) = \frac{R(\alpha)}{\sqrt{Q(\alpha)}}$ yields:

$$\begin{aligned} \frac{d\rho}{d\alpha} &= \sqrt{\frac{1}{2\pi}}\left(\frac{1-\rho^2}{\sqrt{Q}} - \frac{\rho}{Q}\arccos(\rho)\right) \\ \frac{dQ}{d\alpha} &= \sqrt{\frac{2}{\pi}}(\rho-1)\sqrt{Q} + \frac{1}{\pi}\arccos(\rho) \end{aligned} \tag{2}$$

- Asymptotics ($\alpha \to \infty$): $\varepsilon_g \approx \frac{1}{\pi}(\frac{2}{3})^{1/3}\alpha^{-1/3}$

Associative Reinforcement Learning[3]

- so far only immediate supervision considered
- now only graded feedback after L steps:
  - AE ("Average Error"):

$$e_q = \frac{1}{2L} \sum_{l=1}^{L} \left| t^{q,l} - s^{q,l} \right|$$

  - HI ("Hidden Instance"):

$$e_q = \frac{1}{4L^2} \left(\sum_{l=1}^{L} (t^{q,l} - s^{q,l}) \right)^2$$

[3] [*Kuhn*, *Stamatescu*, 2007]

Associative Reinforcement Learning (2)[4]

- the learning algorithm consists of two phases:
  1. L times unsupervised Hebbian learning:
  $$\vec{J}^{q,l+1} = \vec{J}^{q,l} + \frac{a_1}{\sqrt{N}} s^{q,l} \vec{\xi}^{q,l}$$
  2. finally an unspecific reinforcement step:
  $$\vec{J}^{q+1,L} = \vec{J}^{q,L+1} - \frac{a_2}{\sqrt{N}} e_q \sum_{l=1}^{L} s^{q,l} \vec{\xi}^{q,l}$$

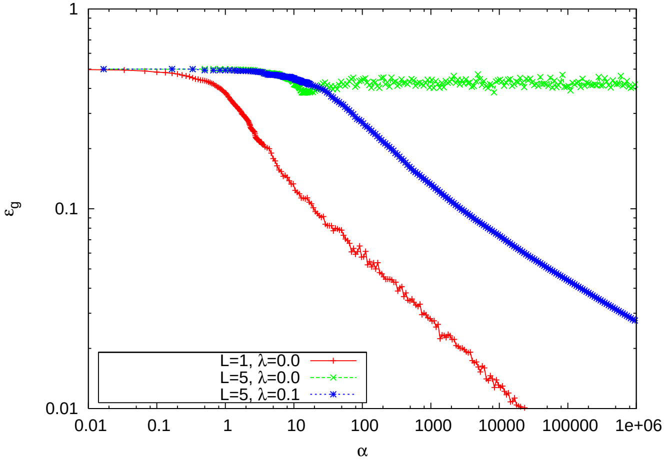
[4] [*Kuhn*, *Stamatescu*, 2007]

Associative Reinforcement Learning (3)

Mechanisms needed for the algorithm have been observed in the brain:

- hippocampal replay of activity sequences during awake state [*Foster*06] as well as in sleep [*Nadasdy*99]
- replays have also been observed in cortex [*Euston*07]
- neuromodulators control the polarity of plasticity [*Seol*07]

Simulation Results



Coarse Graining – Motivation

Goal of the following analysis:

- get rid of random fluctuations
- reduce degrees of freedom
- speed up computing time
- $\longrightarrow$ gain knowledge about the learning behavior

Coarse Graining

- Combine the two phases in one coarse grained step:
$$J_{ki}^{(q+1,1)} = J_{ki}^{(q,1)} + \frac{g_q}{\sqrt{N}} \sum_{l=1}^{L} \mathrm{g}(x_k^{(q,l)})\xi_i^{(q,l)}$$
- $g_q = \lambda - e_q$, $\lambda = a_1/a_2$
- Thermodynamic limit and continuous time limit yield:

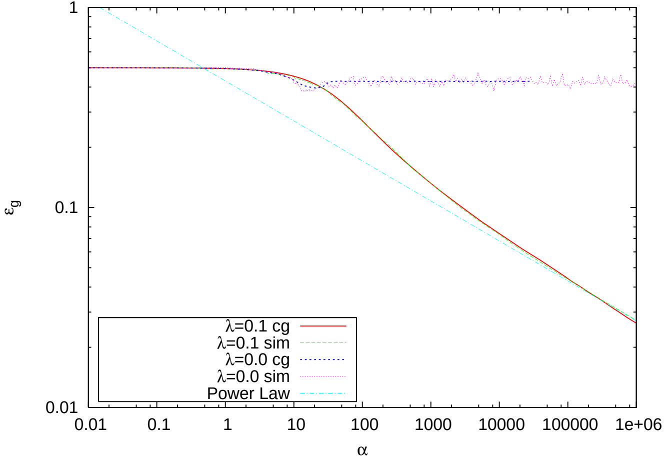
$$\begin{aligned}
\frac{dR_{kk'}}{d\alpha} &= \langle \frac{g_q}{L} \sum_{l}^{L} \mathrm{g}(x_k^{(q,l)}) y_{k'}^{(q,l)} \rangle, \\
\frac{dQ_{kk'}}{d\alpha} &= \langle \frac{g_q^2}{L} \sum_{l}^{L} \mathrm{g}(x_k^{(q,l)}) \mathrm{g}(x_{k'}^{(q,l)}) \rangle \\
&+ \langle \frac{g_q}{L} \sum_{l}^{L} \left[\mathrm{g}(x_k^{(q,l)}) x_{k'}^{(q,l)} + \mathrm{g}(x_{k'}^{(q,l)}) x_k^{(q,l)} \right] \rangle
\end{aligned}$$

Methods

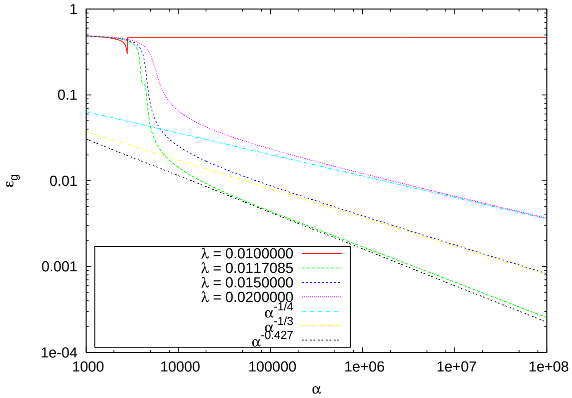
Reduction from NK degrees of freedom to $K(K+1)/2 + KM$ degrees, but

- Integrals cannot be expressed in closed form
- $\rightarrow$ Monte-Carlo Simulation to solve r.h.s.
- Runge-Kutta method to solve the DE

Simulation and Coarse Graining

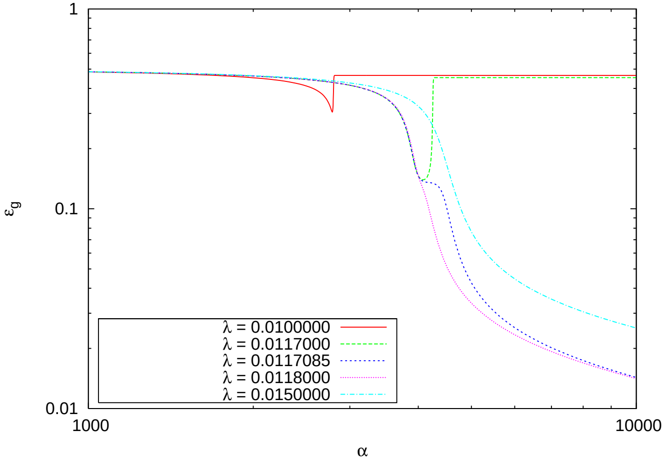


K1 Asymptotics

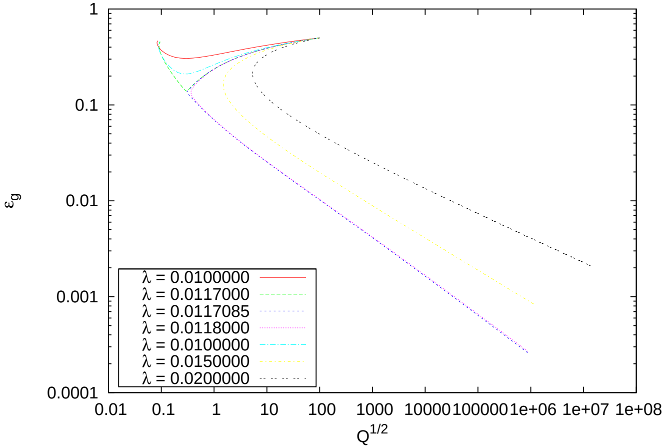


- Figure: $L = 10$ and $\sqrt{Q_0} = 100$
- Above λ_c perfect generalization
- Coarse graining yields $\alpha^{-1/2\lambda L^2}$ asymptotics

K1 Saddlepoint Plateau



K1 Phase Trajectories



K1 Flow

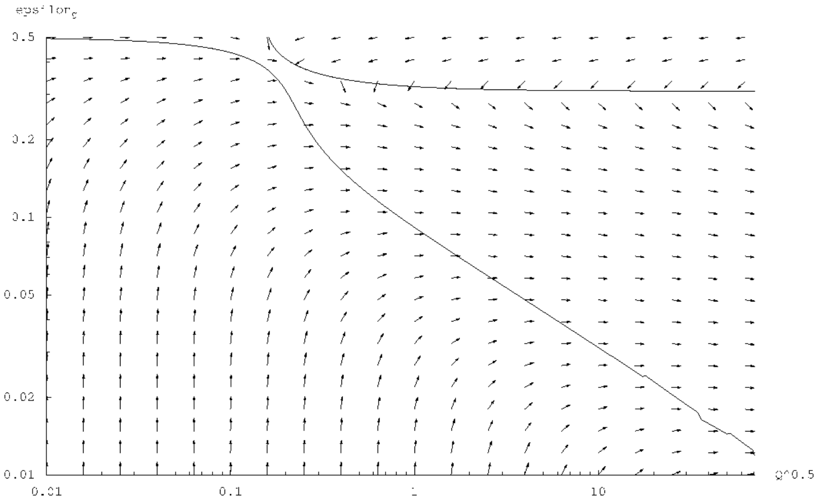


Figure: Flow for $\lambda = 0.3$ and $K = M = 1$.

K1 Flow

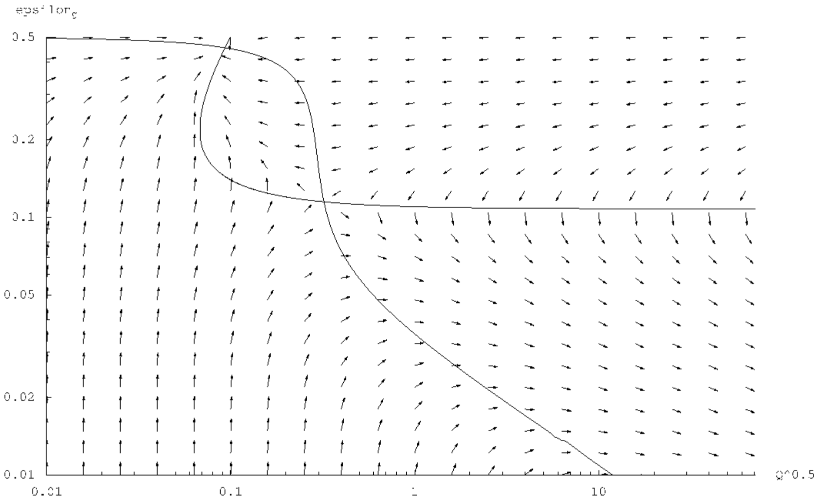
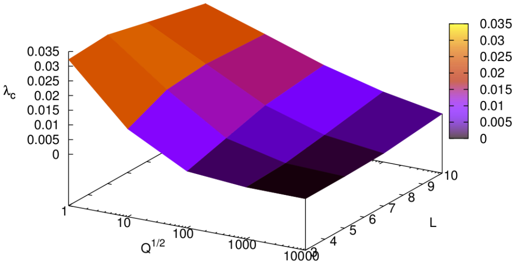


Figure: Flow for $\lambda = 0.1$ and $K = M = 1$.

λ_c Dependence



	$\sqrt{Q_0} = 1$	$\sqrt{Q_0} = 10$	$\sqrt{Q_0} = 100$	$\sqrt{Q_0} = 10000$
L=3	$3.231 \cdot 10^{-2}$	$1.242 \cdot 10^{-2}$	$1.831 \cdot 10^{-3}$	$1.833 \cdot 10^{-5}$
L=5	$3.322 \cdot 10^{-2}$	$1.828 \cdot 10^{-2}$	$5.550 \cdot 10^{-3}$	$8.580 \cdot 10^{-5}$
L=7	$2.991 \cdot 10^{-2}$	$1.954 \cdot 10^{-2}$	$9.424 \cdot 10^{-3}$	$4.543 \cdot 10^{-4}$
L=10	$2.510 \cdot 10^{-2}$	$1.848 \cdot 10^{-2}$	$1.171 \cdot 10^{-2}$	$2.708 \cdot 10^{-3}$

Table: Critical values of λ_c for various L and starting conditions $\sqrt{Q_0}$.

Symmetric Plateau

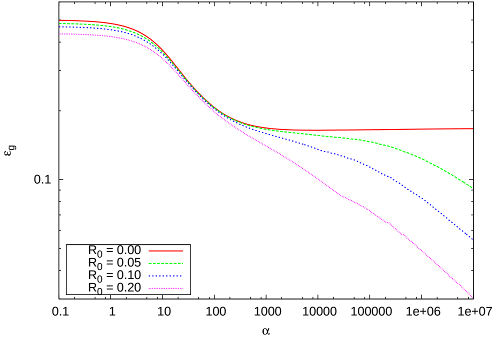


Figure: $K = M = 2$. Less symmetric starting conditions yield smaller plateaus.

Initial conditions:

- $Q_{kk'} = \delta_{kk'}$
- $R_{km} = R_0 \delta_{km}$
- $T_{mm'} = \delta_{mm'}$

Unrealizable Learning

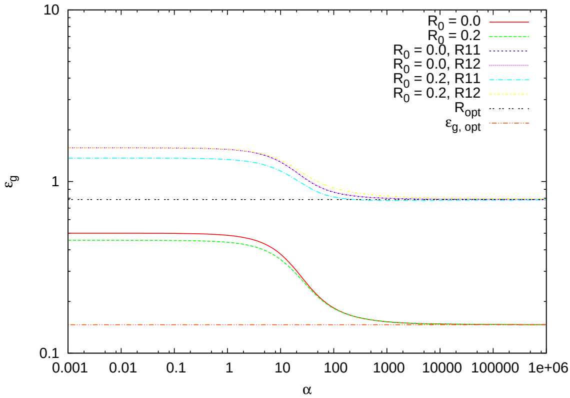


Figure: $K = 1$ and $M = 2$ unrealizable case.

Overrealizable Learning

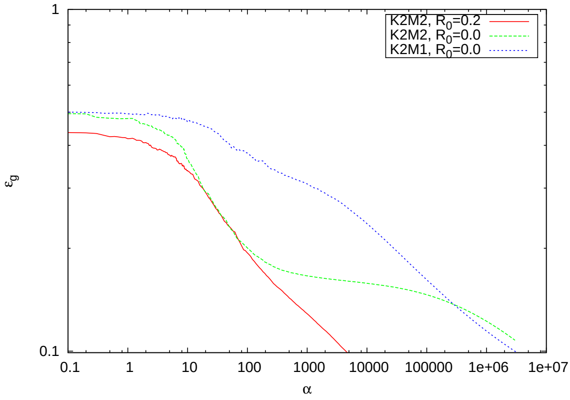


Figure: $K = 2$ and $M = 1$ overrealizable case.

Conclusion

- Standard learning algorithms not sufficient to learn incomplete information
- Non-trivial dynamics:
  - above the bifurcation point λ_c learning always converges
  - below, two fixed points occur
- In the committee machine many fixed points arise, that may disturb learning
- Overrealizable and unrealizable cases converge

Thanks to my collaborators:
Ion-Olimpiu Stamatescu and Reimer Kühn

Thank you for your attention!