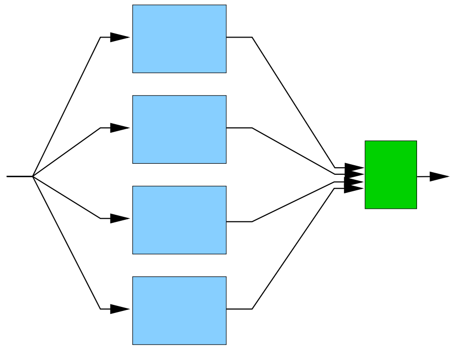
Mixture of Experts

ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization

Committee Machines
Averaging

Specialized Experts
Mixture of Experts
Expectation Maximization

ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization



- Multiple networks
- Output averaging

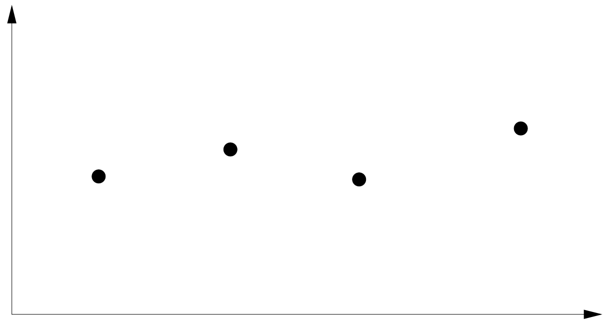
ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization

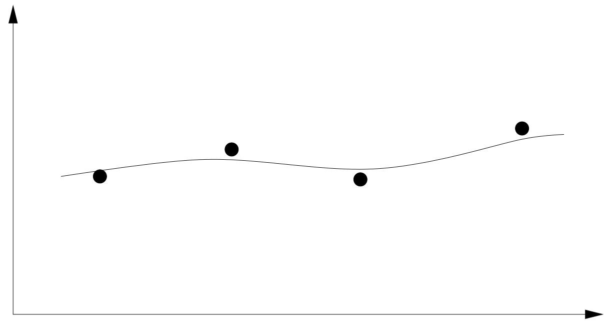
Two ways of utilizing multiple networks

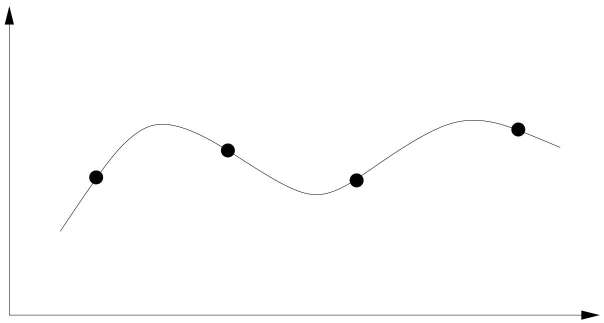
- Smoothen peculiarities of individual nets
- Make the networks specialize

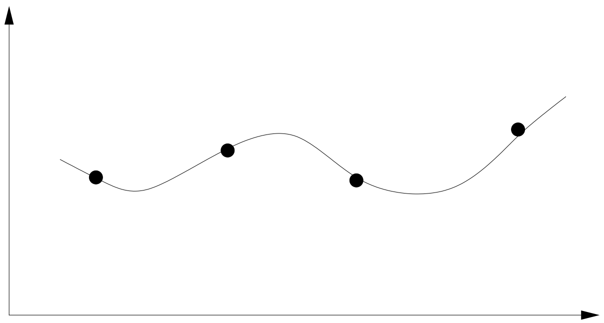
How does over-training affect a network?

- Over-training can be avoided by early stopping
- Results in systematic errors
- Over-trained networks have less error but large variance
- Averaging can reduce this variance

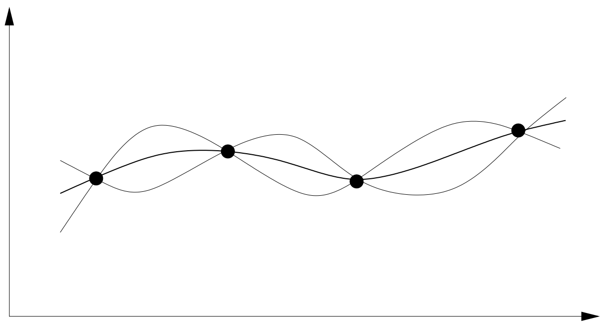








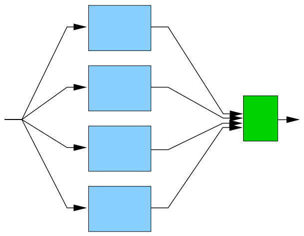
ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization



ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization

Ensemble Averaging

- Train several networks
  - Same topology
  - Same training data
  - Different initial weights
- Train until convergence
- Average any output over all networks



- The networks tend to fall in different local minima
- Averaging smoothens the variations out

ANN fk
Örjan Ekeberg
Committee Machines
Averaging
Specialized Experts
Mixture of Experts
Expectation Maximization

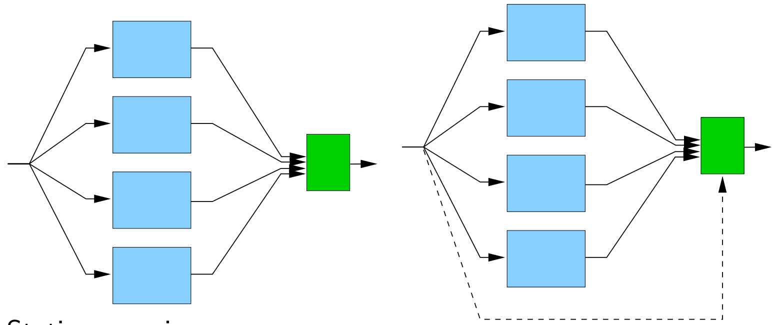
Committee Machines

Averaging

Specialized Experts

Mixture of Experts

Expectation Maximization



Static averaging

Dynamic averaging

World Model

- Data comes from several sources
- Each source generates data with a simple distribution
- Different sources have different probabilities for generating data

Idéa:

- Each network should be an expert of one source
- The gate network chooses which expert to trust

Simple Mixture-of-Experts Network

Expert Network — Single layer, Linear

$$y_k = \vec{w}_k^T \vec{x}$$

Gate Network — Weighted according to *SoftMax*

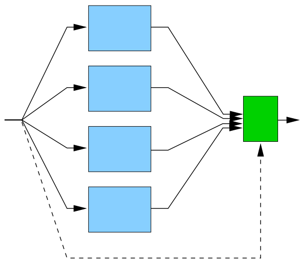
$$y = \sum_k y_k \phi(\vec{a}_k^T \vec{x}) \qquad \text{där } \phi(u_k) = \frac{e^{u_k}}{\sum_i e^{u_i}}$$

Training a Mixtures-of-Experts network

- Gradient Decent
- Expectation Maximization

Gradient Decent

- Maximize Log-Likelihood for observed data
- Function of the weights



- Each expert is updated in proportion to the trust from the gate network
- The gate is updated so that the expert weighting better captures how well the experts are actually doing

Regard the source of the data as unobservable variables

Expectation Maximization

Repeat

1. Estimate the probability for each source having generated each pattern
2. Update the source model parameters to match these estimates

- E-step
  Calculate the probability that a pattern x comes from source u given the source model parameters $\hat{\Theta}$

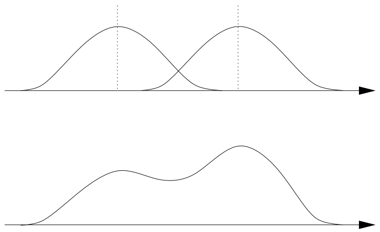
$$P(u|x, \hat{\Theta})$$

- M-step
  Compute new parameters Θ that maximizes expected likelihood

$$\Theta = \underset{\Theta}{\operatorname{argmax}} \sum_u P(u|x, \hat{\Theta}) \log P(x, u|\Theta)$$

Classical EM-problem

- Mix of two normal distributions
- Find the center of both distributions $< \mu_1, \mu_2 >$



$$Q_{i,j} = P(u_i | x_j, < \mu_1, \mu_2 >) = \frac{e^{-(x_j - \mu_i)^2/2\sigma^2}}{\sum_k e^{-(x_j - \mu_k)^2/2\sigma^2}}$$

$$\mu_i = \frac{1}{m} \sum_{j=1}^{m} Q_{i,j} x_j$$