Artificiel Neural Networks 2

Morten Nielsen
Department of Health Technology,
DTU

Outline

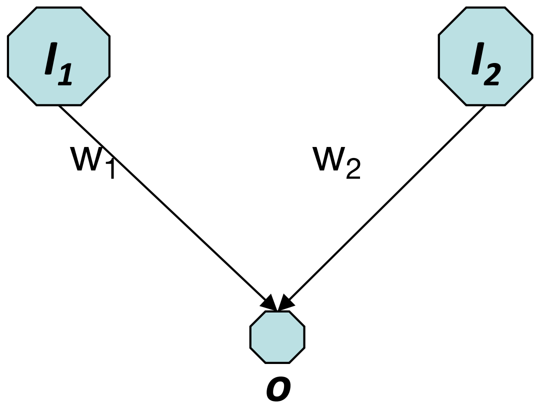
- Optimization procedures
  - Gradient decent (this you already know)
- Network training
  - back propagation
  - cross-validation
  - Over-fitting
  - Examples
  - Deeplearning

Neural network. Error estimate

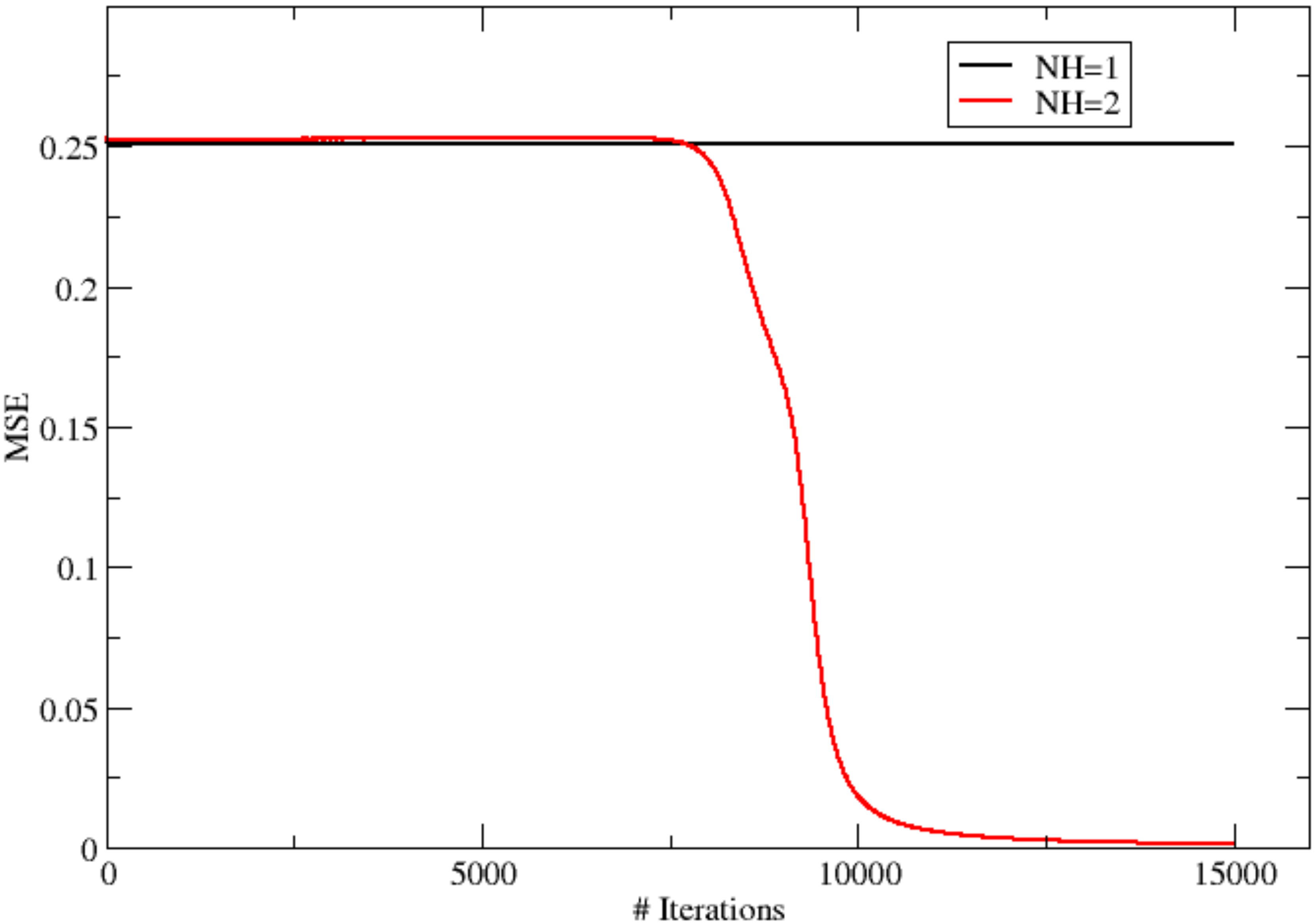
Linear function

$$o = I_1 \cdot w_1 + I_2 \cdot w_2$$

$$E = \tfrac{1}{2} \cdot (o - t)^2$$



Neural networks

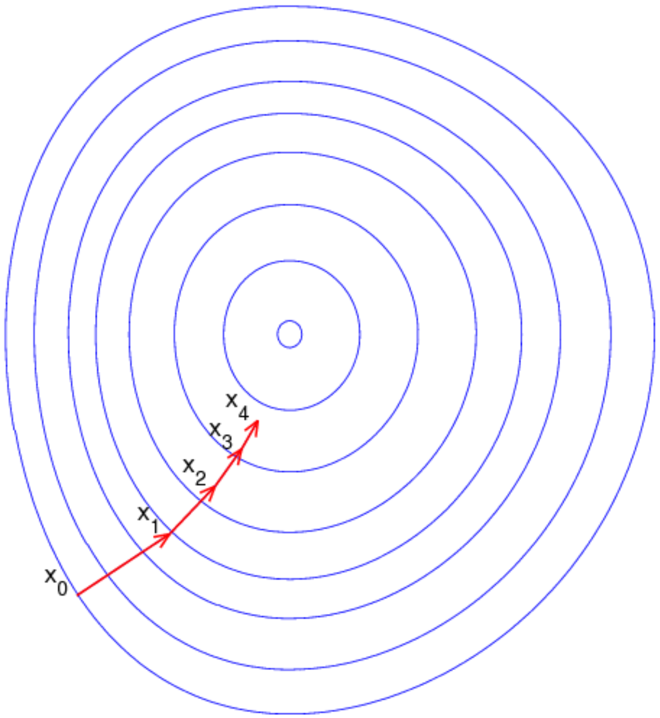


Gradient decent (from wekipedia)

Gradient descent is based on the observation that if the real-valued function F(x) is defined and differentiable in a neighborhood of a point **a**, then F(x) decreases fastest if one goes from **a** in the direction of the negative gradient of F at **a**.
It follows that, if

$$b = a - \varepsilon \cdot \nabla F(a)$$

for $\varepsilon > 0$ a small enough number,
then F(b)<F(a)

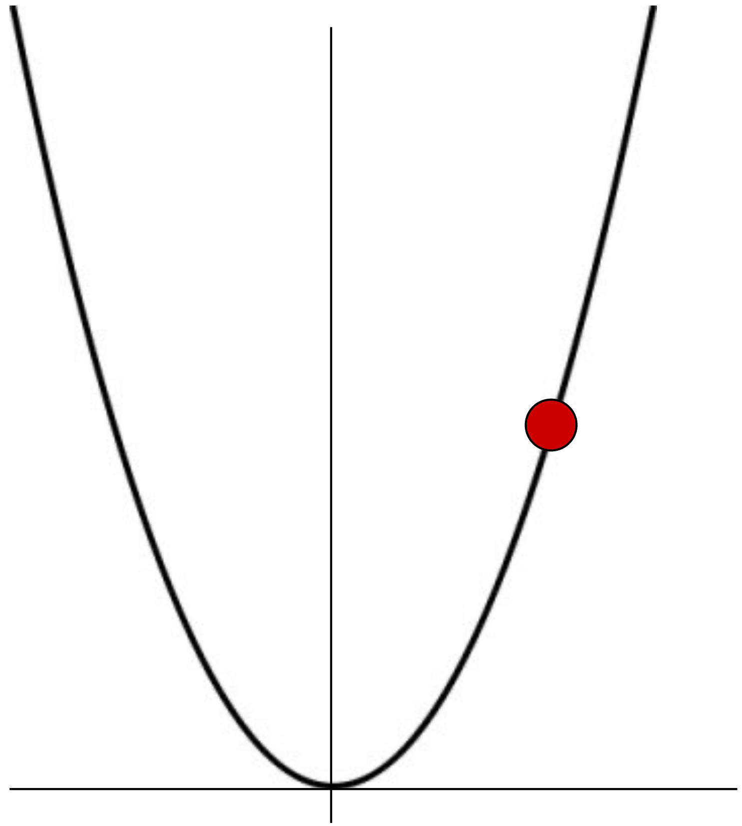


Gradient decent (example)

$$F(x) = x^2$$

$$a = 2$$

$$F(a) = 4$$



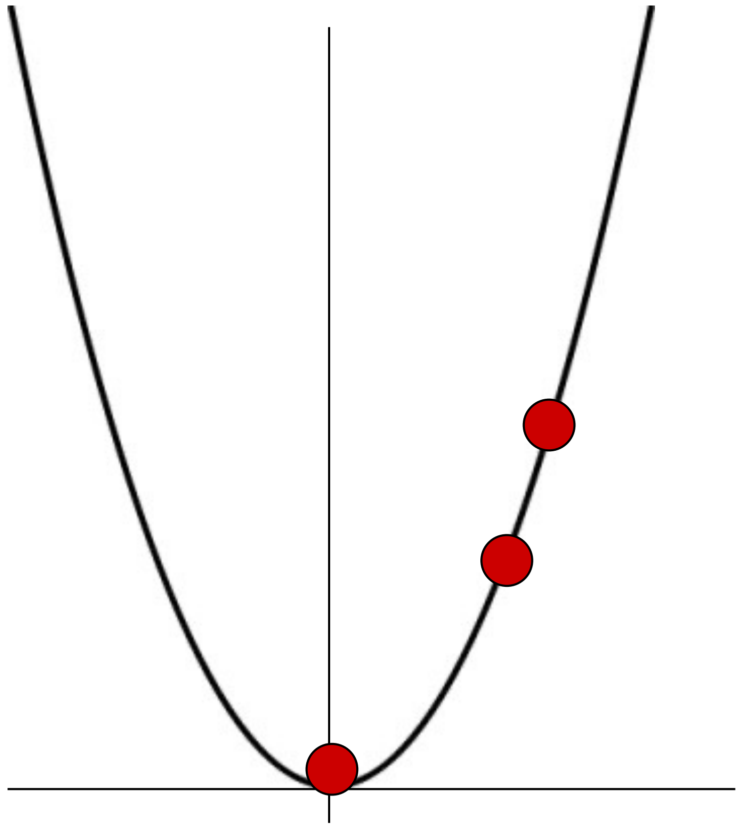
Gradient decent (example)

$$F(x) = x^2$$

$$a = 2$$

$$F(a) = 4$$

$$\frac{\partial F}{\partial x} = 2 \cdot x = 4$$



$$b = a - \varepsilon \cdot \nabla F(a) = 2 - 0.1 \cdot 4 = 1.6$$

Gradient decent. Example

Weights are changed in the opposite direction of the gradient of the error

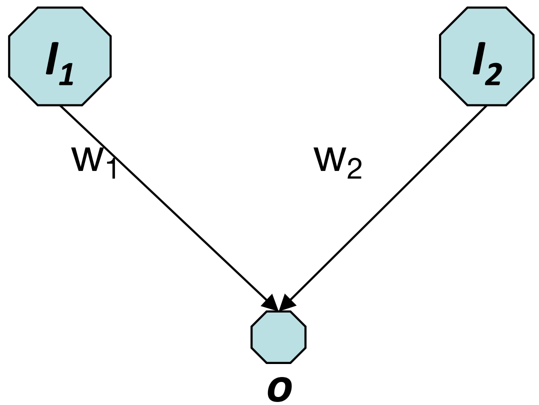
$$w_i^{'} = w_i + \Delta w_i$$

$$E = \tfrac{1}{2} \cdot (O - t)^2$$

$$O = \sum_i w_i \cdot I_i$$

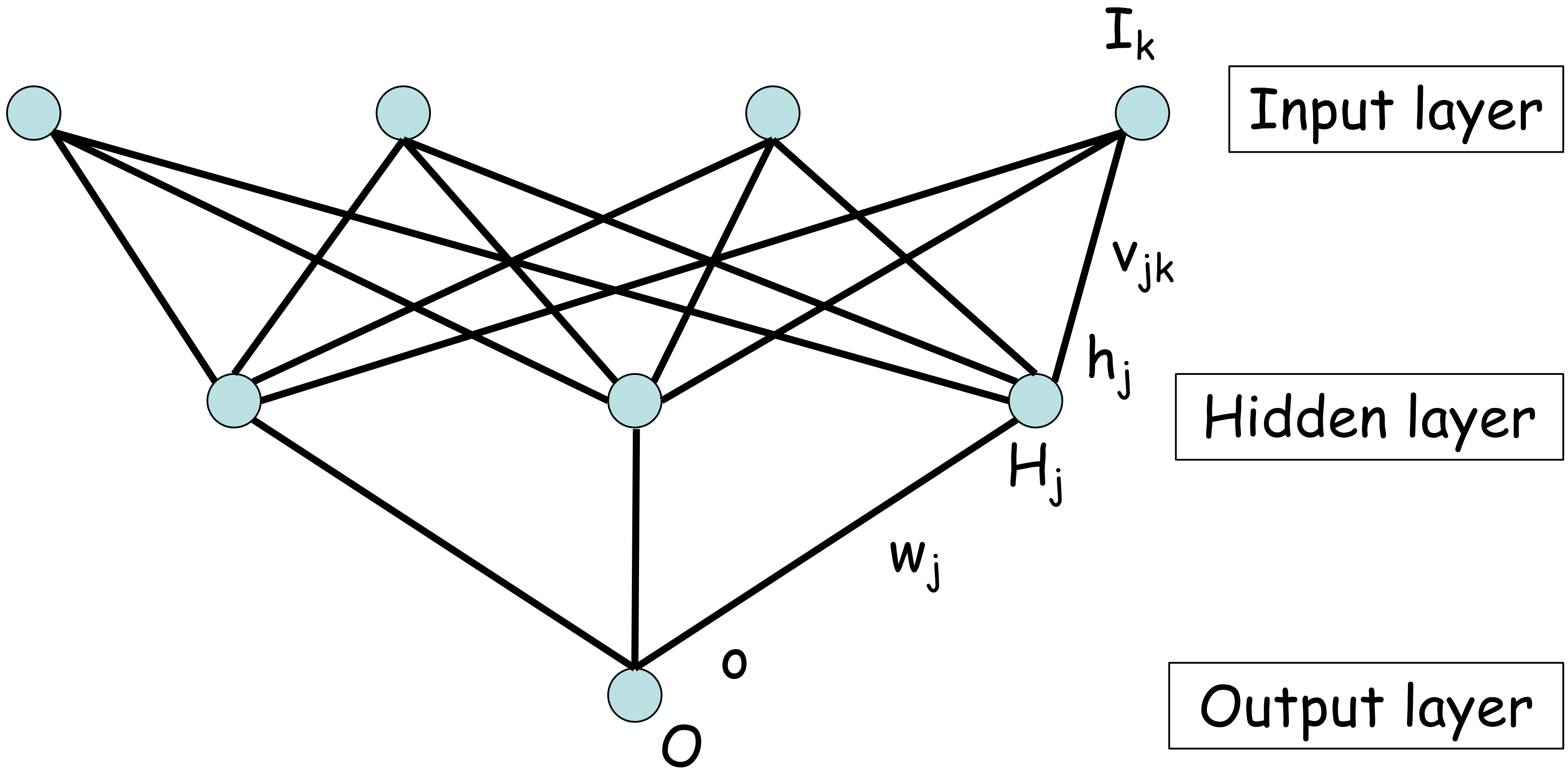
Linear function

$$O = I_1 \cdot w_1 + I_2 \cdot w_2$$



$$\Delta w_i = -\varepsilon \cdot \frac{\partial E}{\partial w_i} = -\varepsilon \cdot \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial w_i} = -\varepsilon \cdot (O - t) \cdot I_i$$

Network architecture



What about the hidden layer?

$$\Delta w_i = -\varepsilon \cdot \frac{\partial E}{\partial w_i}$$

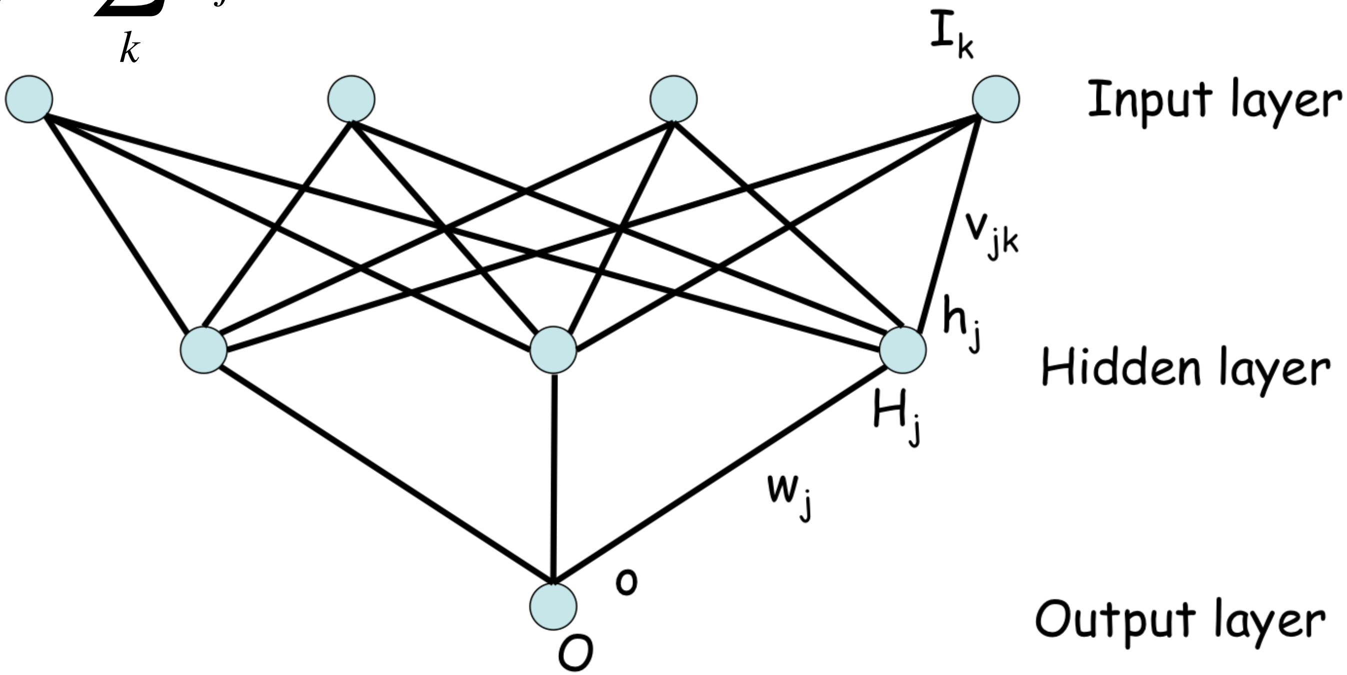
$$E = \tfrac{1}{2} \cdot (O - t)^2$$

$$O = g(o), H = g(h)$$

$$o = \sum_j w_j \cdot H_j$$

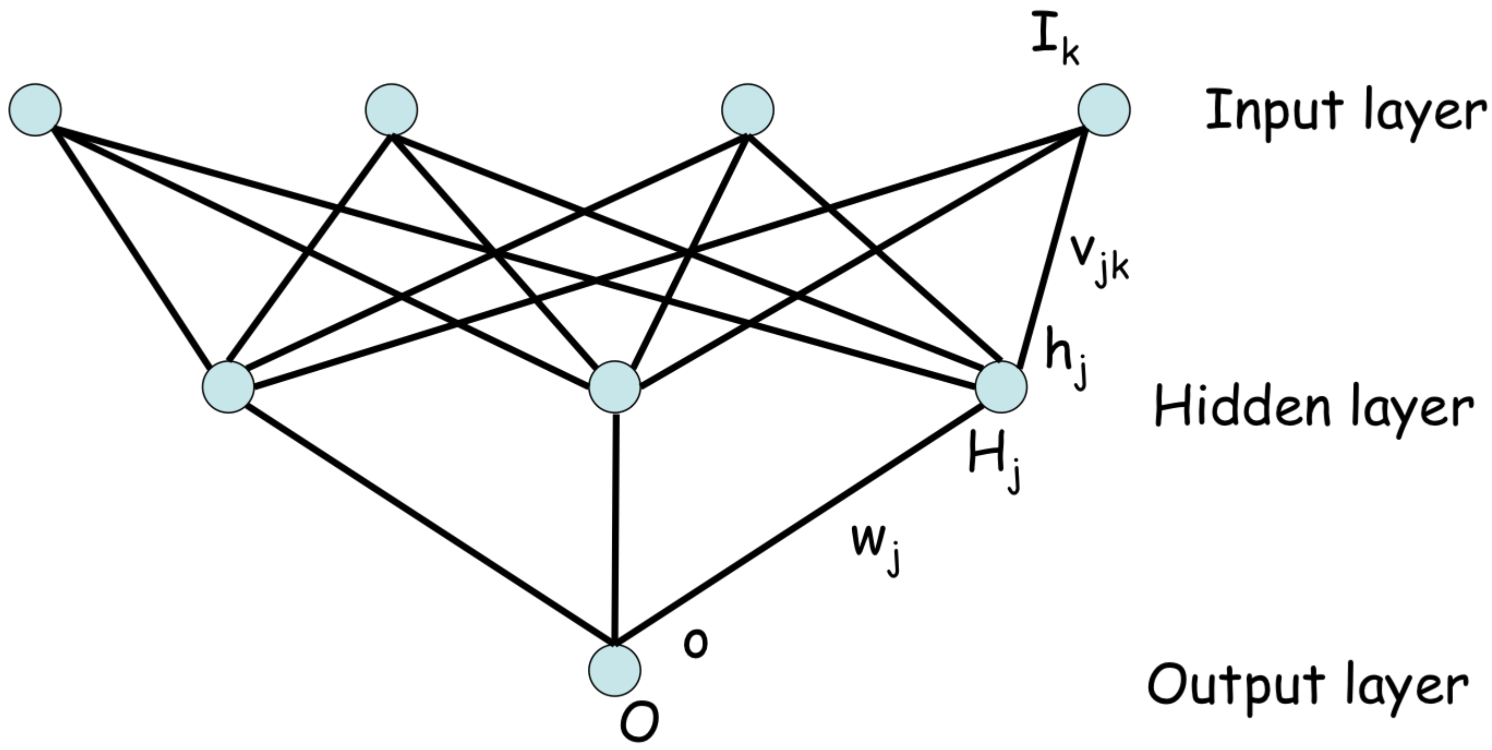
$$g(x) = \frac{1}{1 + e^{-x}}$$

$$h_j = \sum_k v_{jk} \cdot I_k$$



Hidden to output layer

$$\frac{\partial E}{\partial w_j} = \frac{\partial E(O(o(w_j)))}{\partial w_j} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial o} \cdot \frac{\partial o}{\partial w_j}$$



Hidden to output layer

CENTERFORBIOLOGICALSEQUENCEANALYSIS CBS

$$\boxed{\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial o} \cdot \frac{\partial o}{\partial w_j} \qquad =}$$

$$\frac{\partial E}{\partial O} = (O - t)$$

$$\frac{\partial O}{\partial o} = \frac{\partial g}{\partial o} = ?$$

$$\frac{\partial o}{\partial w_j} =$$

$$O = g(o)$$

$$g(x) = \frac{1}{1 + e^{-x}}$$

$$g'(x) = \frac{-1}{(1 + e^{-x})^2} \cdot (-e^{-x})$$

$$= (1 - g(x)) \cdot g(x)$$

Hidden to output layer

$$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial o} \cdot \frac{\partial o}{\partial w_j} \qquad =$$

$$\frac{\partial E}{\partial O} = (O - t)$$

$$\frac{\partial O}{\partial o} = \frac{\partial g}{\partial o} = ?$$

$$\frac{\partial o}{\partial w_j} = \frac{1}{\partial w_j} \sum_l w_l \cdot H_l = H_j$$

$$O = g(o)$$

$$g(x) = \frac{1}{1 + e^{-x}}$$

$$g'(x) = \frac{-1}{(1 + e^{-x})^2} \cdot (-e^{-x})$$

$$= (1 - g(x)) \cdot g(x)$$

Hidden to output layer

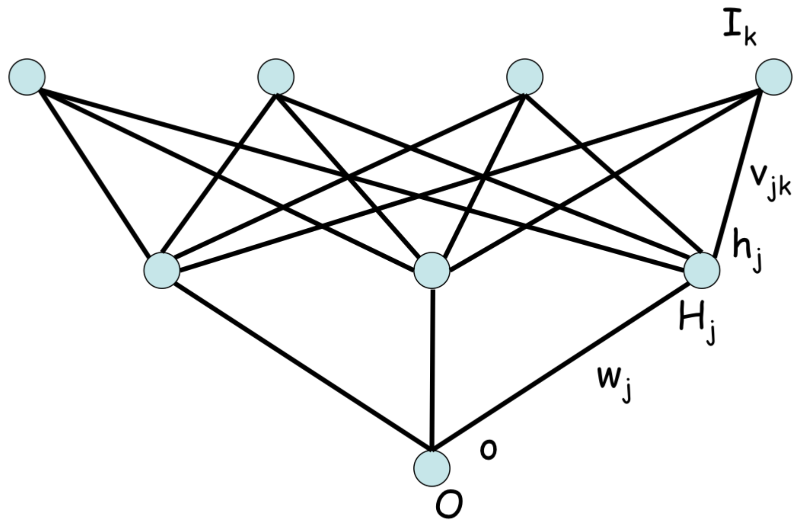
$$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial o} \cdot \frac{\partial o}{\partial w_j} \qquad = (O - t) \cdot g'(o) \cdot H_j$$

$$= (O - t) \cdot (1 - O) \cdot O \cdot H_j$$

$$O = g(o)$$

$$g'(o) = (1 - g(o)) \cdot g(o)$$

$$= (1 - O) \cdot O$$

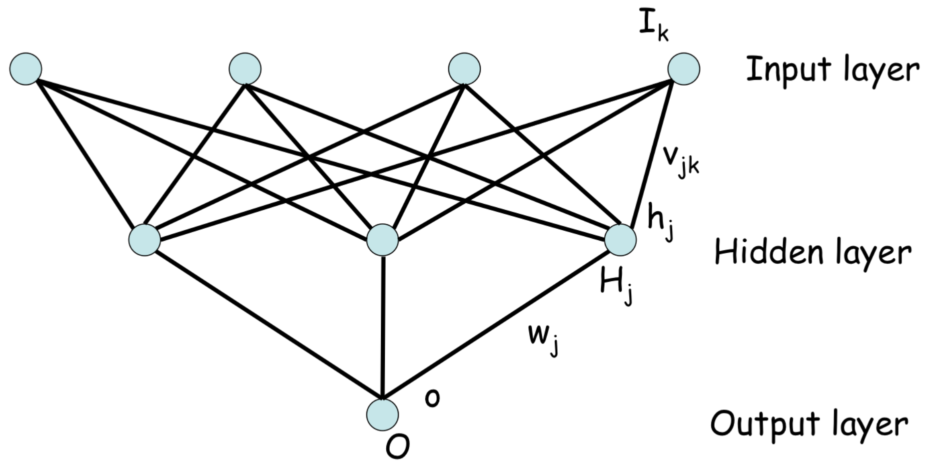


Input to hidden layer

$$\frac{\partial E}{\partial v_{jk}} = \frac{\partial E(O(o(H_j(h_j(v_{jk})))))}{\partial v_{jk}}$$

$$= \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial o} \cdot \frac{\partial o}{\partial H_j} \cdot \frac{\partial H_j}{\partial h_j} \cdot \frac{\partial h_j}{\partial v_{jk}}$$

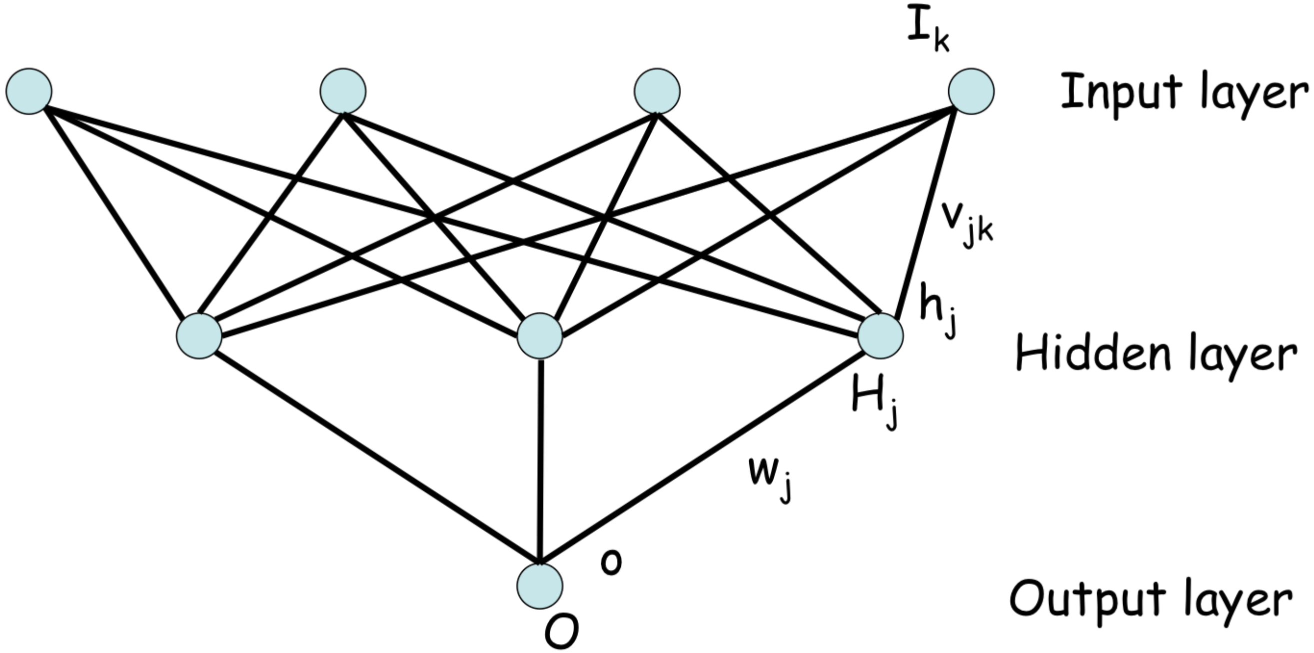
$$= (O - t) \cdot g'(o) \cdot w_j \cdot g'(h_j) \cdot I_k$$



Summary

$$\frac{\partial E}{\partial w_j} = (O - t) \cdot g'(o) \cdot H_j$$

$$\frac{\partial E}{\partial v_{jk}} = (O - t) \cdot g'(o) \cdot w_j \cdot g'(h_j) \cdot I_k$$



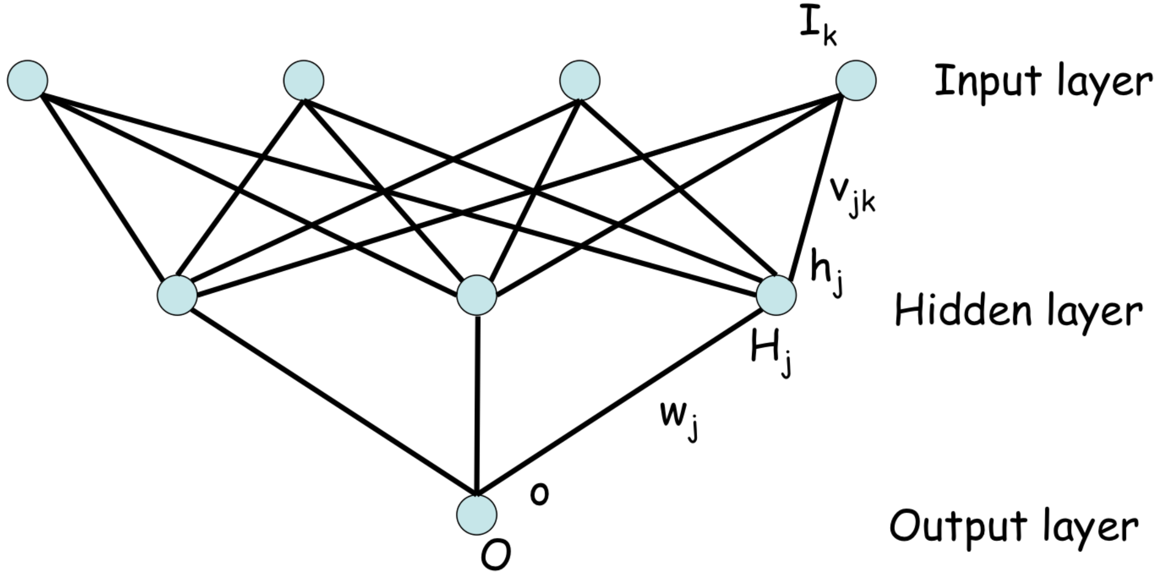
Or

CENTERFORBIOLOGICALSEQUENCEANALYSIS CBS

$$\frac{\partial E}{\partial w_j} = (O - t) \cdot g'(o) \cdot H_j = \delta \cdot H_j$$

$$\frac{\partial E}{\partial v_{jk}} = (O - t) \cdot g'(o) \cdot w_j \cdot g'(h_j) \cdot I_k = \delta \cdot w_j \cdot g'(h_j) \cdot I_k$$

$$\delta = (O - t) \cdot g'(o)$$



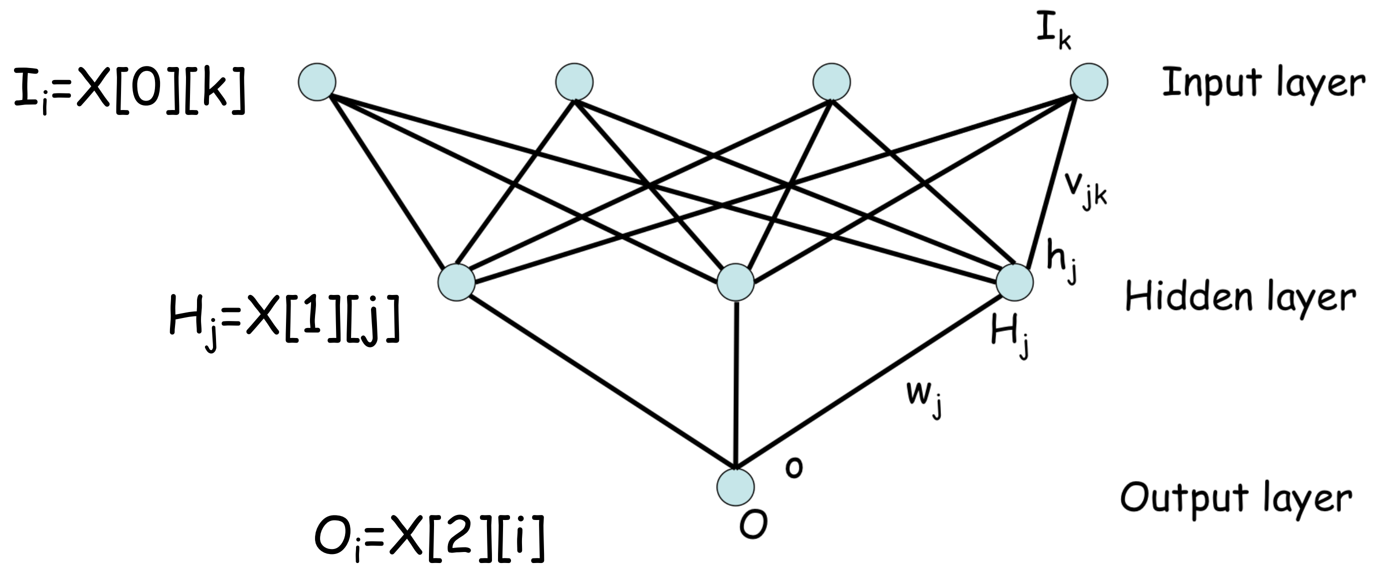
Or

CENTERFO
RBIOLOGI
CALSEQU
ENCEANA
LYSIS **CBS**

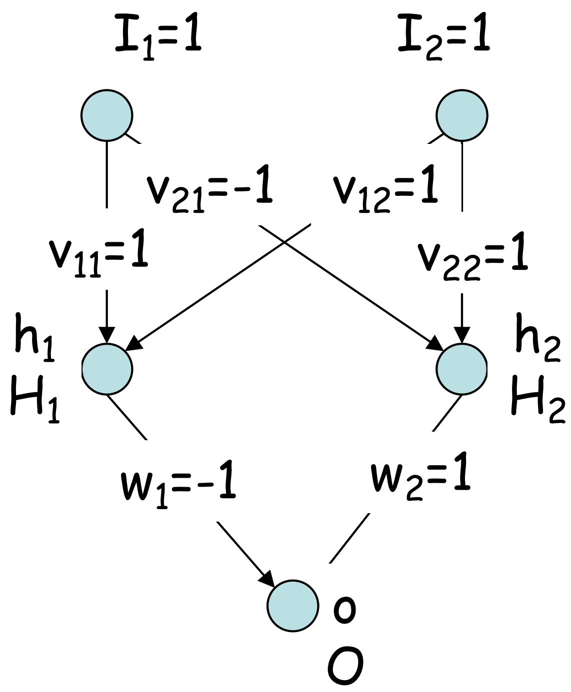
$$\frac{\partial E}{\partial w_j} = \delta \cdot H_j = \delta \cdot x[1][j]$$

$$\frac{\partial E}{\partial v_{jk}} = \delta \cdot w_j \cdot g'(h_j) \cdot I_k = \delta \cdot w_j \cdot x[1][j] \cdot (1 - x[1][j]) \cdot I_k$$

$$\delta = (O - t) \cdot g'(o) = (x[2][i] - t_i) \cdot x[2][i] \cdot (1 - x[2][i])$$



Can you do it your self?



$$\Delta w_j = -\varepsilon \cdot \frac{\partial E}{\partial w_j}; \Delta v_{jk} = -\varepsilon \cdot \frac{\partial E}{\partial v_{jk}}$$

$$\frac{\partial E}{\partial w_j} = (O-t)\cdot g'(o)\cdot H_j$$

$$\frac{\partial E}{\partial v_{jk}} = (O-t)\cdot g'(o)\cdot w_j \cdot g'(h_j)\cdot I_k$$

$$g'(x) = (1-g(x))\cdot g(x)$$

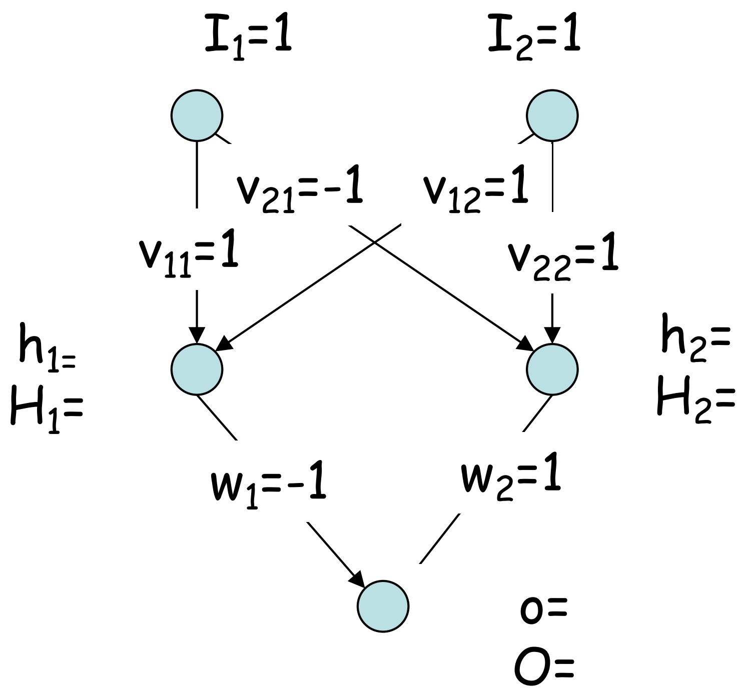
$$O = g(o)$$

What is the output (O) from the network?
What are the Δw_{ij} and Δv_{jk} values if the target value is 0 and ε=0.5?

Can you do it your self (ε=0.5).

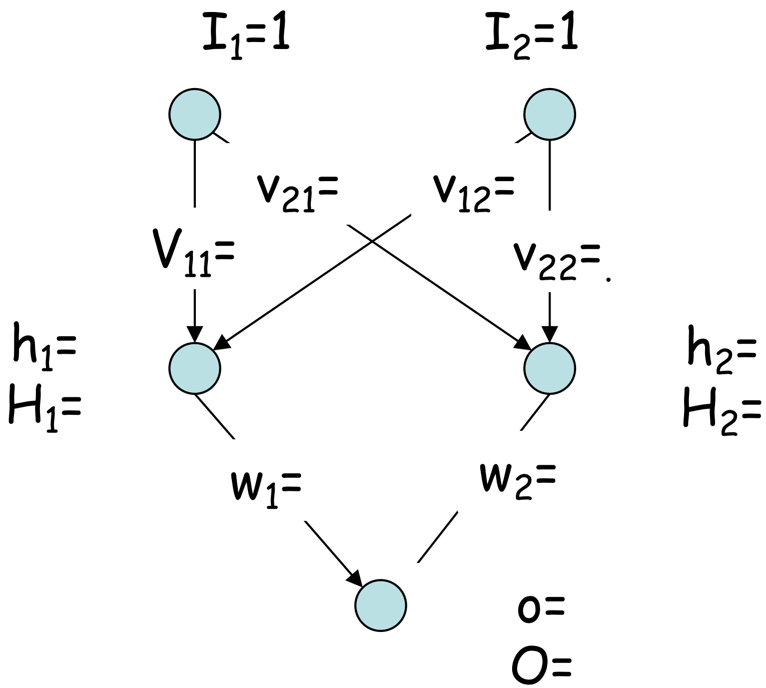
Has the error decreased?

Before



$$\Delta w_1 = ??$$
$$\Delta w_2 = ??$$

After

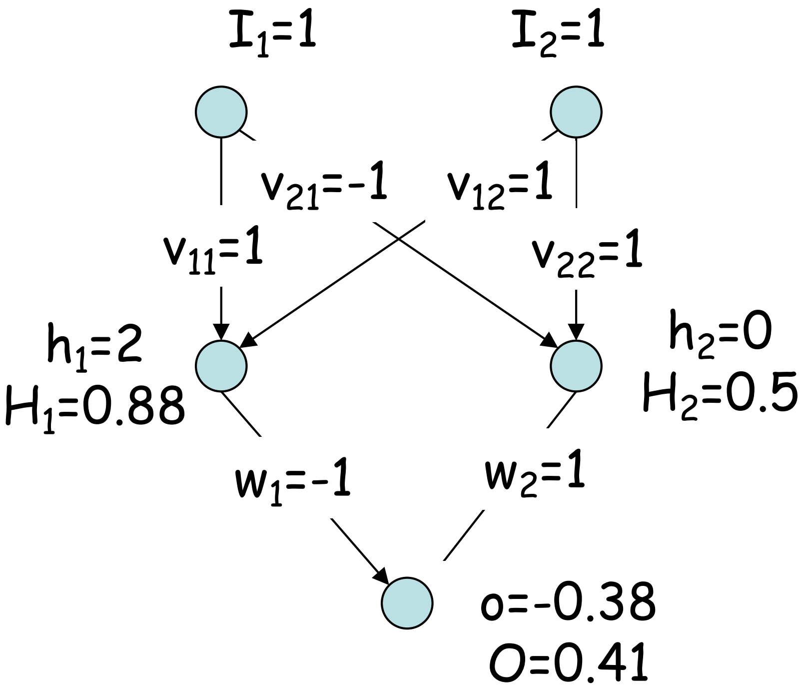


$$\Delta v_{11} = ??$$
$$\Delta v_{12} = ??$$
$$\Delta v_{21} = ??$$
$$\Delta v_{22} = ??$$

Can you do it your self (ε=0.5).

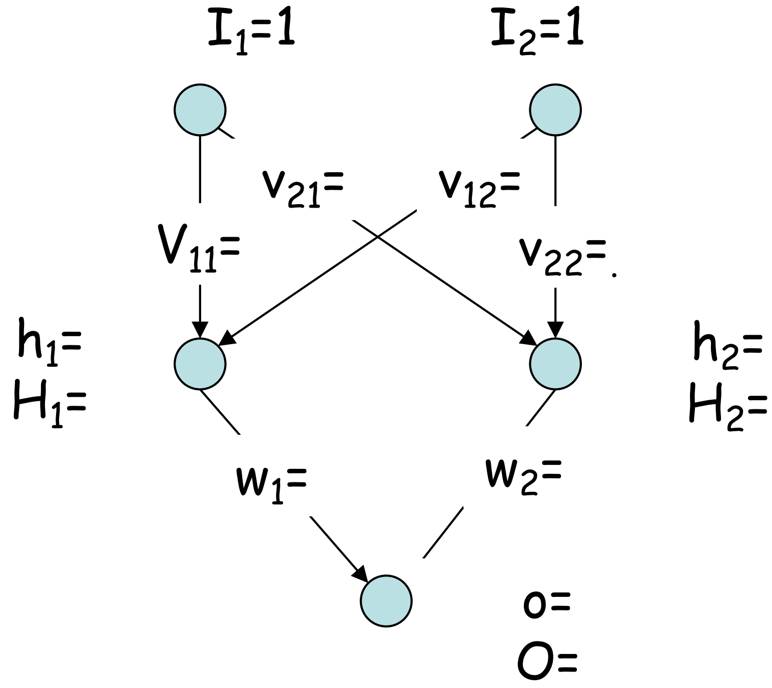
Has the error decreased?

Before



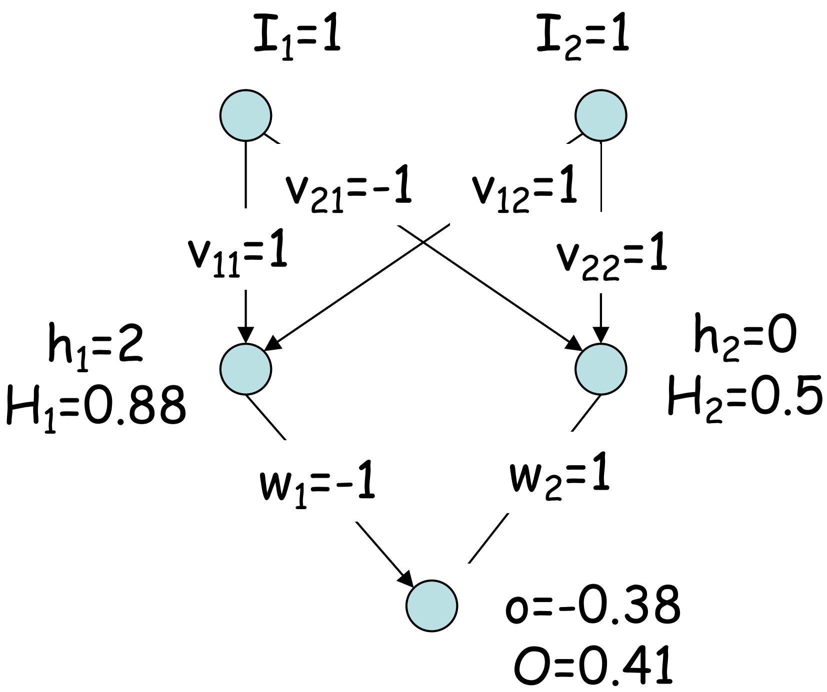
$$\Delta w_1 = ??$$
$$\Delta w_2 = ??$$

After



$$\Delta v_{11} = ??$$
$$\Delta v_{12} = ??$$
$$\Delta v_{21} = ??$$
$$\Delta v_{22} = ??$$

Can you do it your self?



- What is the output (O) from the network?
- What are the Δw_{ij} and Δv_{jk} values if the target value is 0?

$$\Delta w_j = -\varepsilon \cdot \frac{\partial E}{\partial w_j}; \Delta v_{jk} = -\varepsilon \cdot \frac{\partial E}{\partial v_{jk}}$$

$$\frac{\partial E}{\partial w_j} = (O-t)\cdot g'(o)\cdot H_j = \delta \cdot H_j$$

$$\delta = (O-t)\cdot g'(o) = 0.41\cdot 0.41\cdot(1-0.41) = 0.099$$

$$\Delta w_1 = -\varepsilon\cdot\delta\cdot 0.88 = -\varepsilon\cdot 0.087$$

$$\Delta w_2 = -\varepsilon\cdot\delta\cdot 0.5 = -\varepsilon\cdot 0.050$$

$$\frac{\partial E}{\partial v_{jk}} = g'(h_j)\cdot I_k\cdot(O-t)\cdot g'(o)\cdot w_j$$

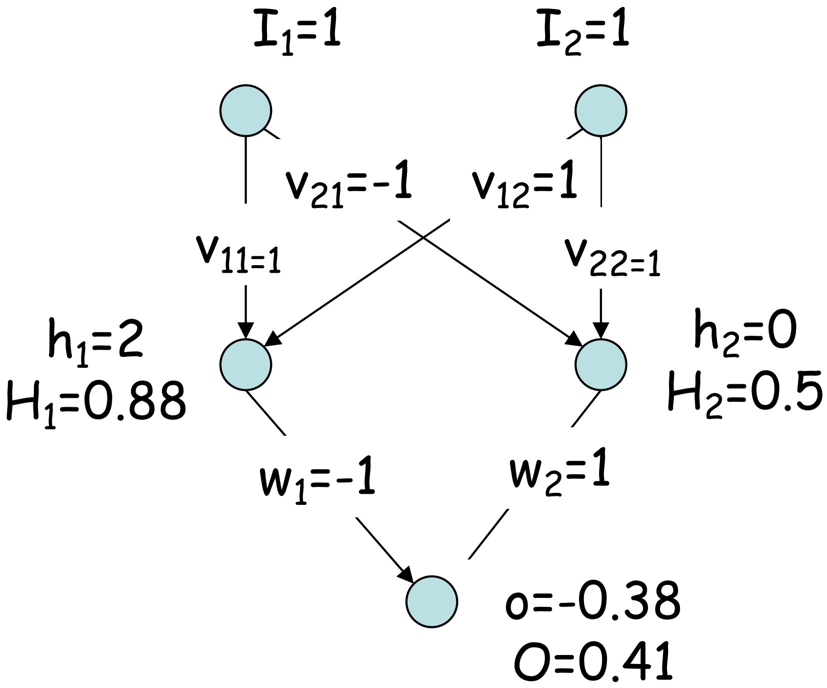
$$\Delta v_{11} = -\varepsilon\cdot H_1\cdot(1-H_1)\cdot 1\cdot\delta\cdot(-1) = \varepsilon\cdot 0.01$$

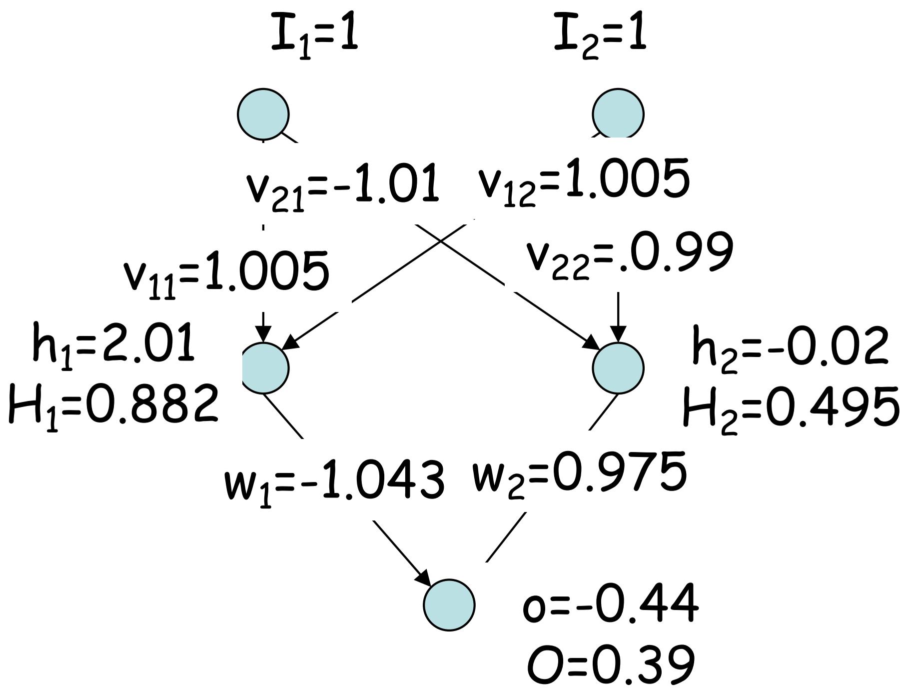
$$\Delta v_{12} = \Delta v_{11}$$

$$\Delta v_{21} = -\varepsilon\cdot H_2\cdot(1-H_2)\cdot 1\cdot\delta\cdot 1 = -\varepsilon\cdot 0.02$$

$$\Delta v_{22} = \Delta v_{21}$$

Can you do it your self (ε=0.5). Has the error decreased?





$$\Delta w_1 = -\varepsilon \cdot \delta \cdot 0.88 = -\varepsilon \cdot 0.087$$
$$\Delta w_2 = -\varepsilon \cdot \delta \cdot 0.5 = -\varepsilon \cdot 0.050$$

$$\Delta v_{11} = -\varepsilon \cdot H_1 \cdot (1 - H_1) \cdot 1 \cdot \delta \cdot (-1) = \varepsilon \cdot 0.01$$
$$\Delta v_{12} = \Delta v_{11}$$
$$\Delta v_{21} = -\varepsilon \cdot H_2 \cdot (1 - H_2) \cdot 1 \cdot \delta \cdot 1 = -\varepsilon \cdot 0.02$$
$$\Delta v_{22} = \Delta v_{21}$$

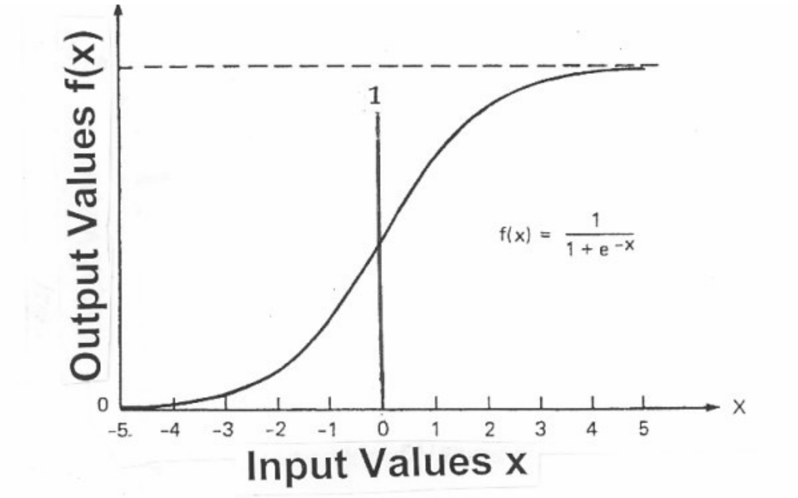
Sequence encoding

- Change in weight is linearly dependent on input value
- “True” sparse encoding (i.e 1/0) is therefore highly inefficient
- Sparse is most often encoded as
  - +1/-1 or 0.9/0.05

$$\boxed{\frac{\partial E}{\partial v_{jk}} = \delta \cdot w_j \cdot g'(h_j) \cdot I_k = \delta \cdot w_j \cdot x[1][j] \cdot (1 - x[1][j]) \cdot I_k}$$

Sequence encoding - rescaling

- Rescaling the input values



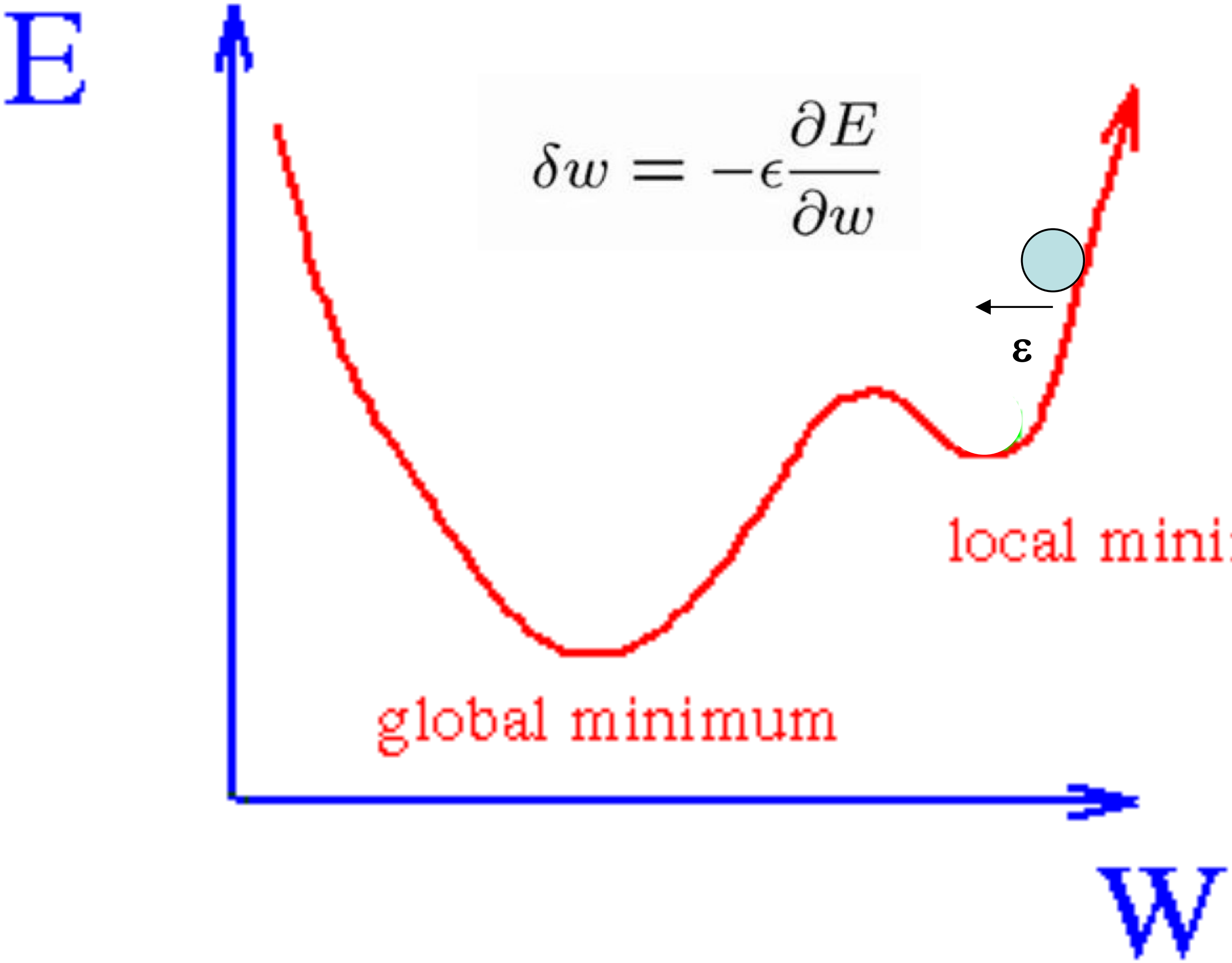
If the input (o or h) is too large or too small, g´ is zero and the weights are not changed. Optimal performance is when o,h are close to 0.5

$$\frac{\partial E}{\partial w_j} = (O - t) \cdot g'(o) \cdot H_j = \delta \cdot H_j$$

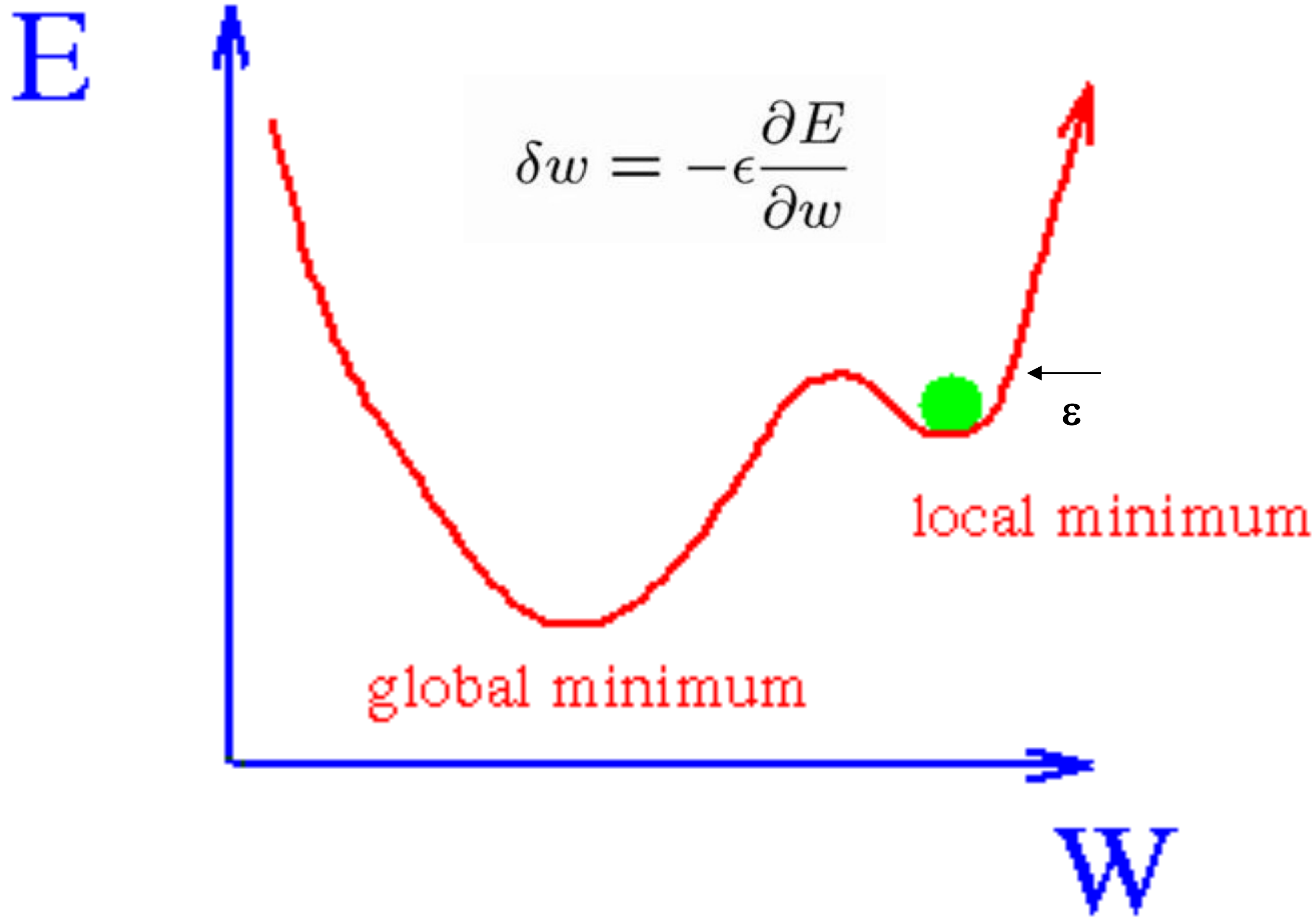
$$\frac{\partial E}{\partial v_{jk}} = g'(h_j) \cdot I_k \cdot (O - t) \cdot g'(o) \cdot w_j = g'(h_j) \cdot I_k \cdot \delta \cdot w_j$$

$$\delta = (O - t) \cdot g'(o)$$

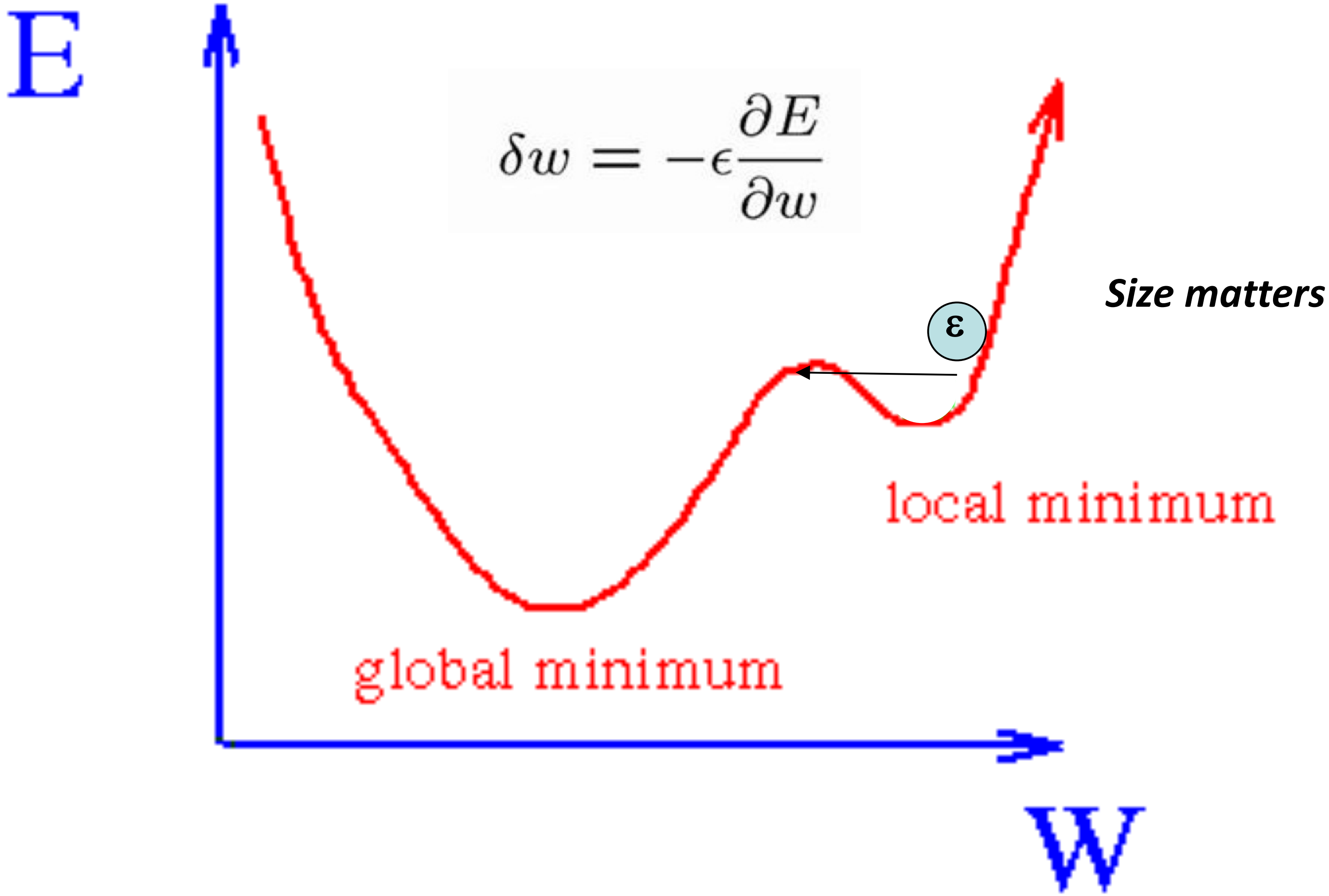
Training and error reduction



Training and error reduction



Training and error reduction



Demo

- http://playground.tensorflow.org/

Do hidden neurons matter?

CENTERFO RBIOLOGI CALSEQU ENCEANA LYSIS CBS

- The environment matters

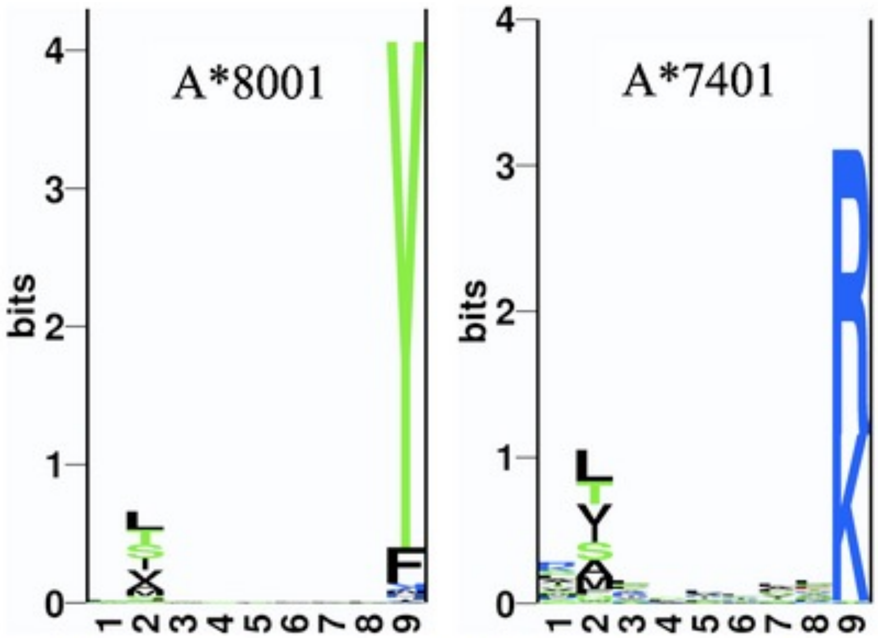
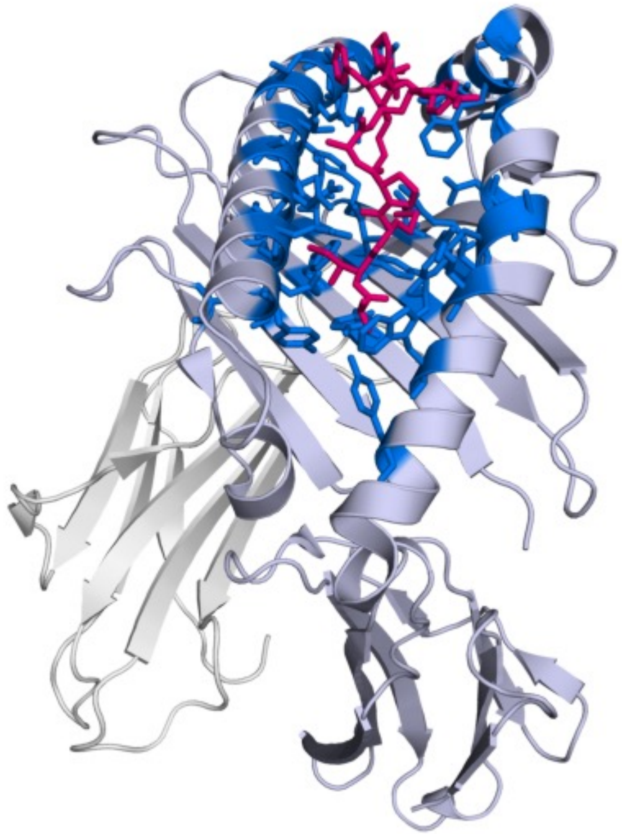


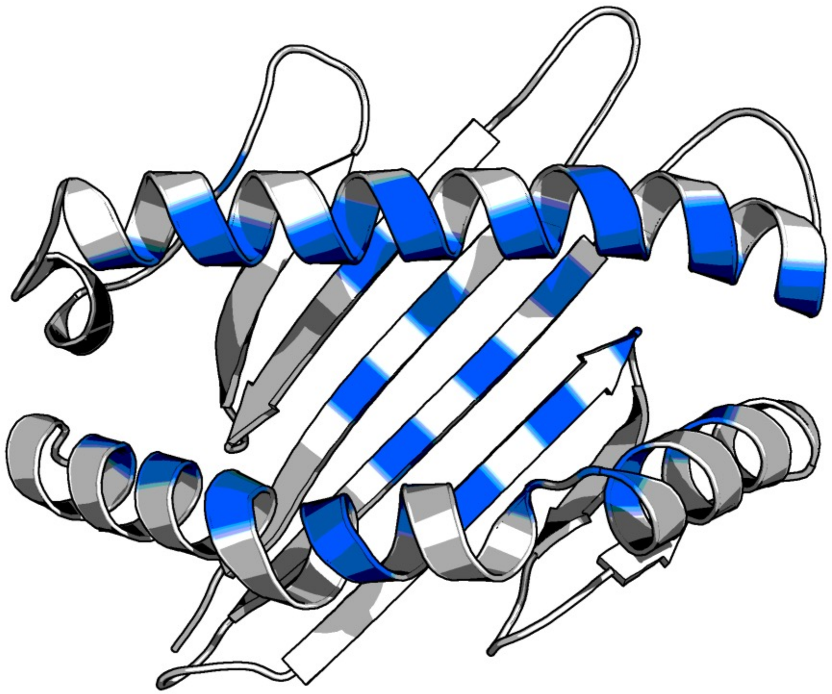
Figure 1. Prospective validation using hitherto uncharacterized HLA molecules.



NetMHCpan

Context matters

```
FMIDWILDA YFAMYGEKVAHTHVDTLYVRYHYYTWAVLAYTWY 0.89 A0201
FMIDWILDA YFAMYQENMAHTDANTLYIIYRDYTWVARVYRGY 0.08 A0101
DSDGSFFLY YFAMYGEKVAHTHVDTLYVRYHYYTWAVLAYTWY 0.08 A0201
DSDGSFFLY YFAMYQENMAHTDANTLYIIYRDYTWVARVYRGY 0.85 A0101
```



Summary

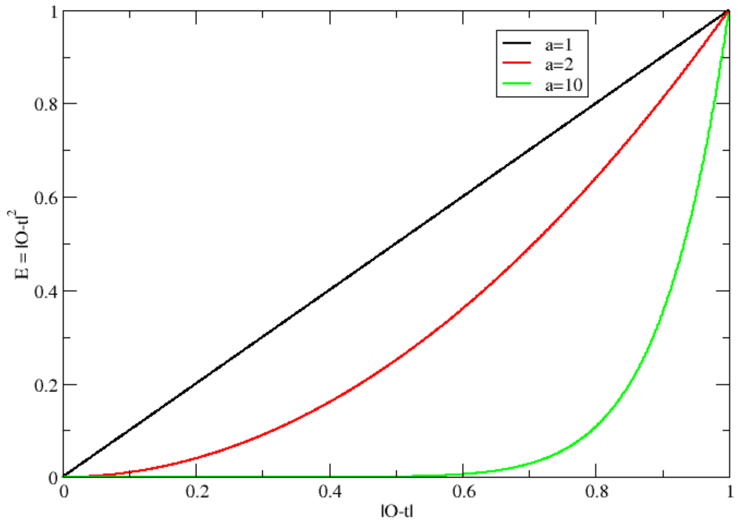
- Gradient decent is used to determine the updates for the synapses in the neural network
- Some relatively simple math defines the gradients
  - Networks without hidden layers can be solved on the back of an envelope (SMM exercise)
  - Hidden layers are a bit more complex, but still ok
- Always train networks using a test set to stop training
  - Be careful when reporting predictive performance
    - Use “nested” cross-validation for small data sets
- And hidden neurons do matter (sometimes)

And some more stuff for the long cold and rainy summer nights

CENTERFO RBIOLOGI CALSEQU ENCEANA LYSIS **CBS**

- Can it maybe be made differently?

$$E = \frac{1}{\alpha} \cdot (O - t)^{\alpha}$$

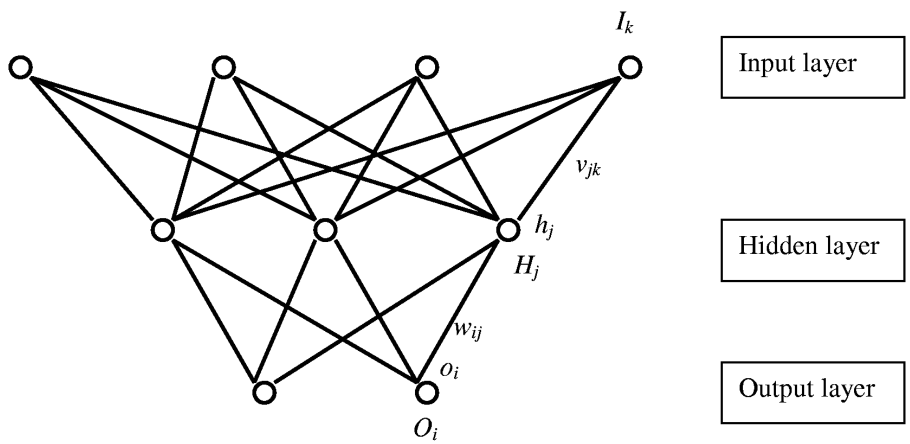


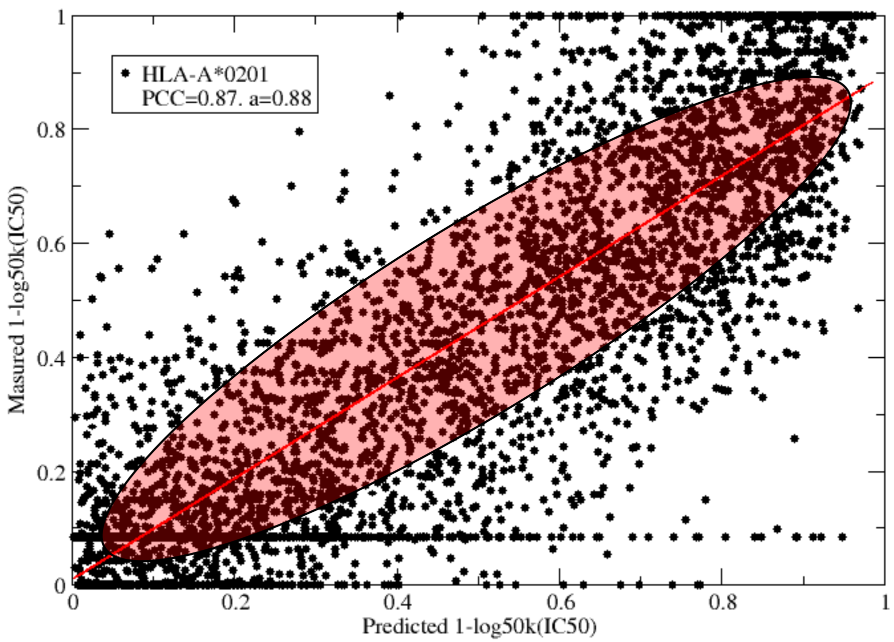
Predicting accuracy

- Can it be made differently?

$$E = \frac{1}{2} \cdot (O_1 - t)^2 \cdot O_2 + \lambda \cdot (1 - O_2)$$

Reliability





Making sense of ANN weights

- Identification of position specific receptor ligand interactions by use of artificial neural network decomposition. An investigation of interactions in the MHC:peptide system

Master thesis' by Frederik Otzen Bagger and Piotr Chmura

Making sense of ANN weights

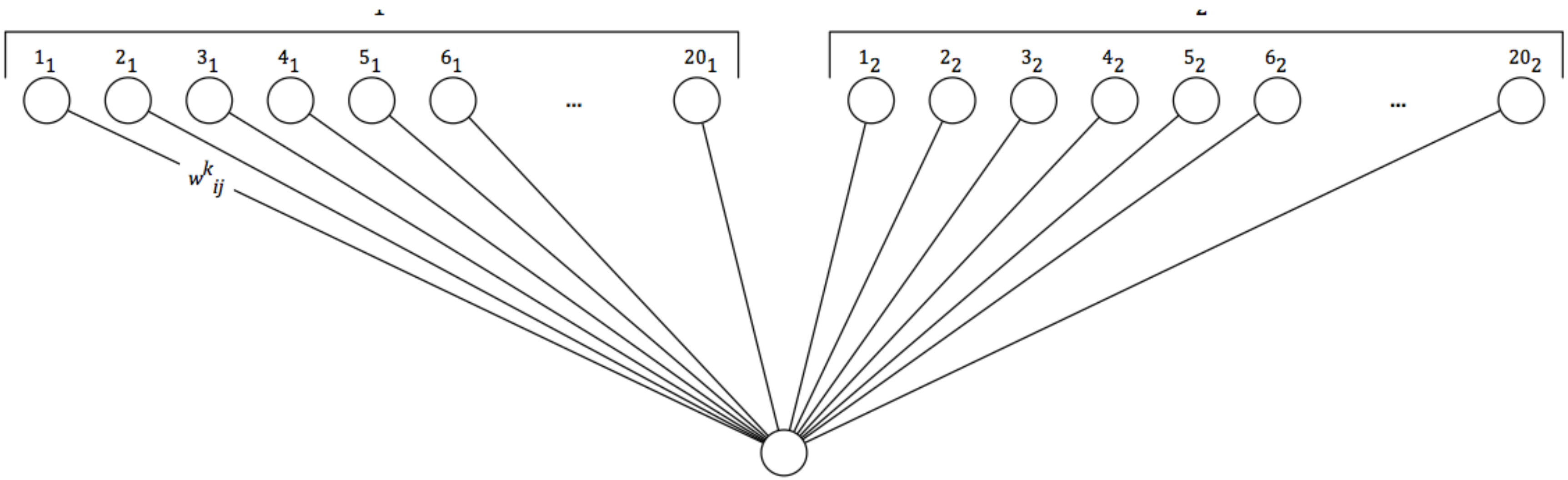
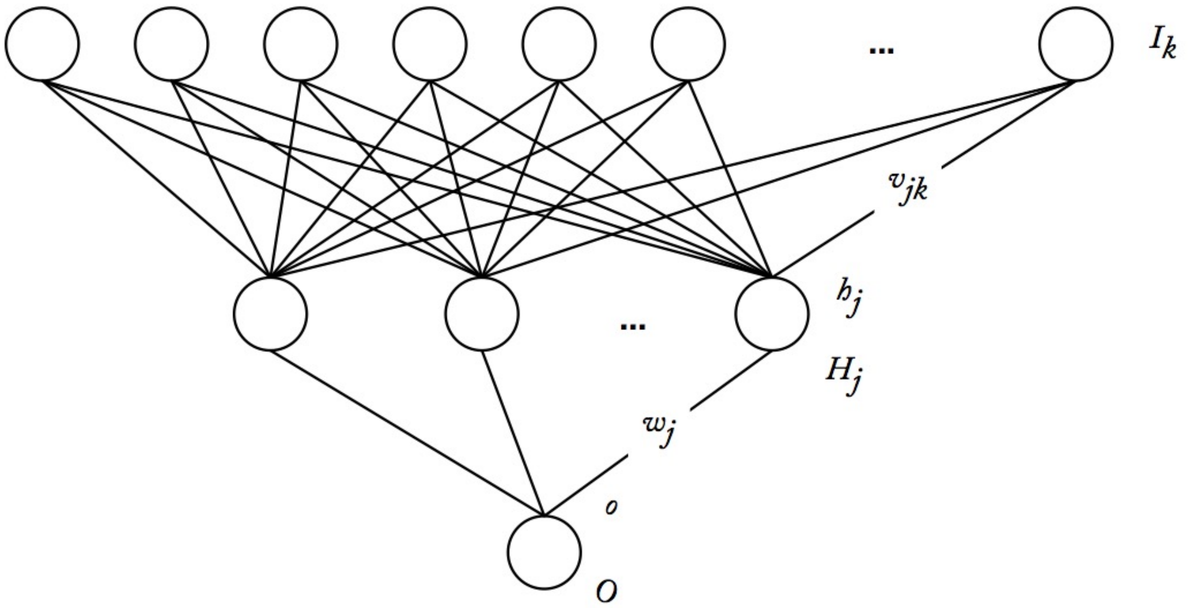


Figure 2.1. Two layer ANN with two amino acids as input, $p = \{1,2\}$, and one output neuron. The direction is downwards, and the graph is directed.



Making sense of ANN weights

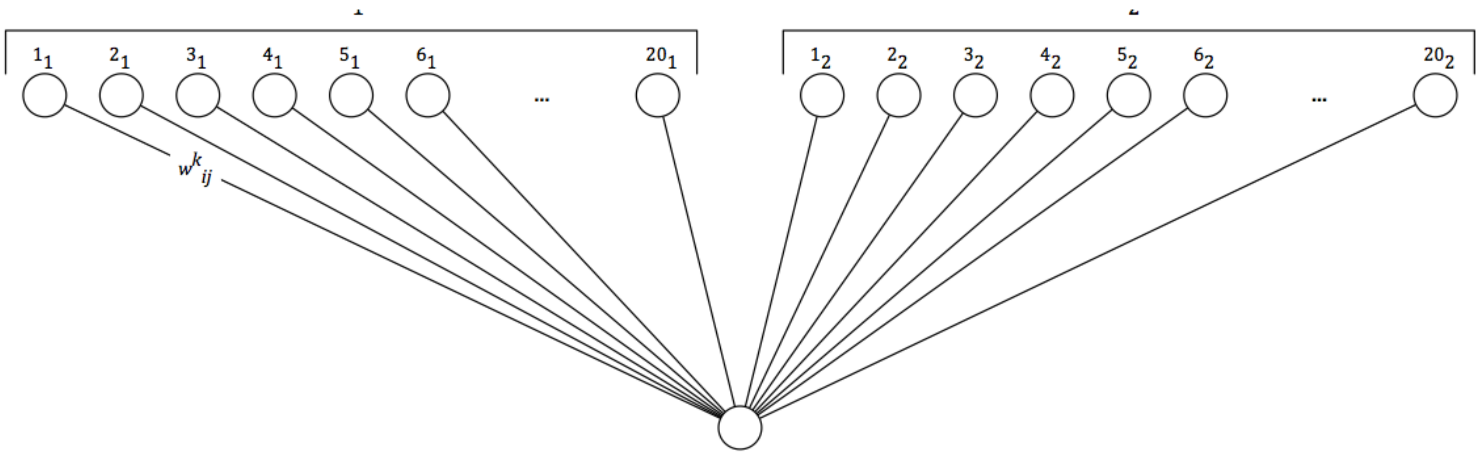
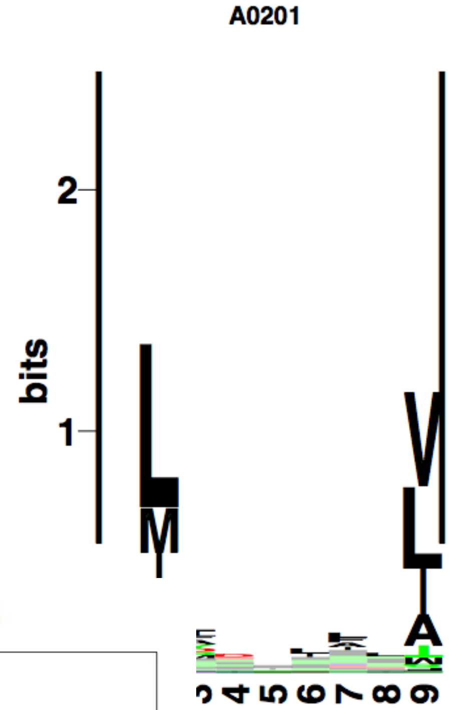
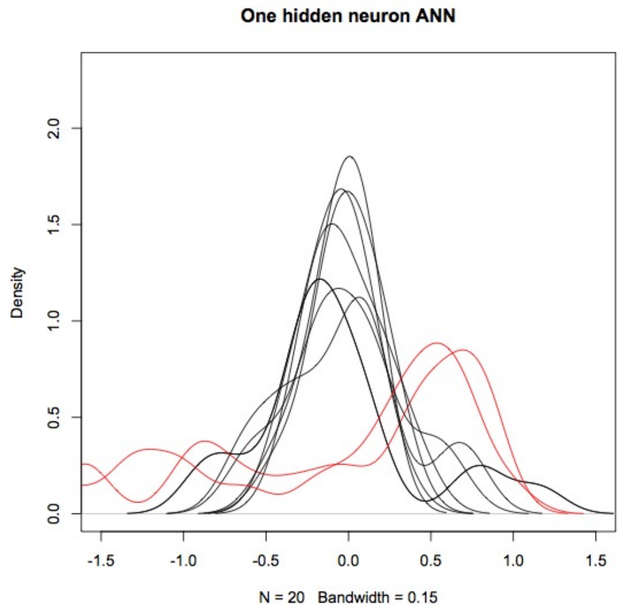


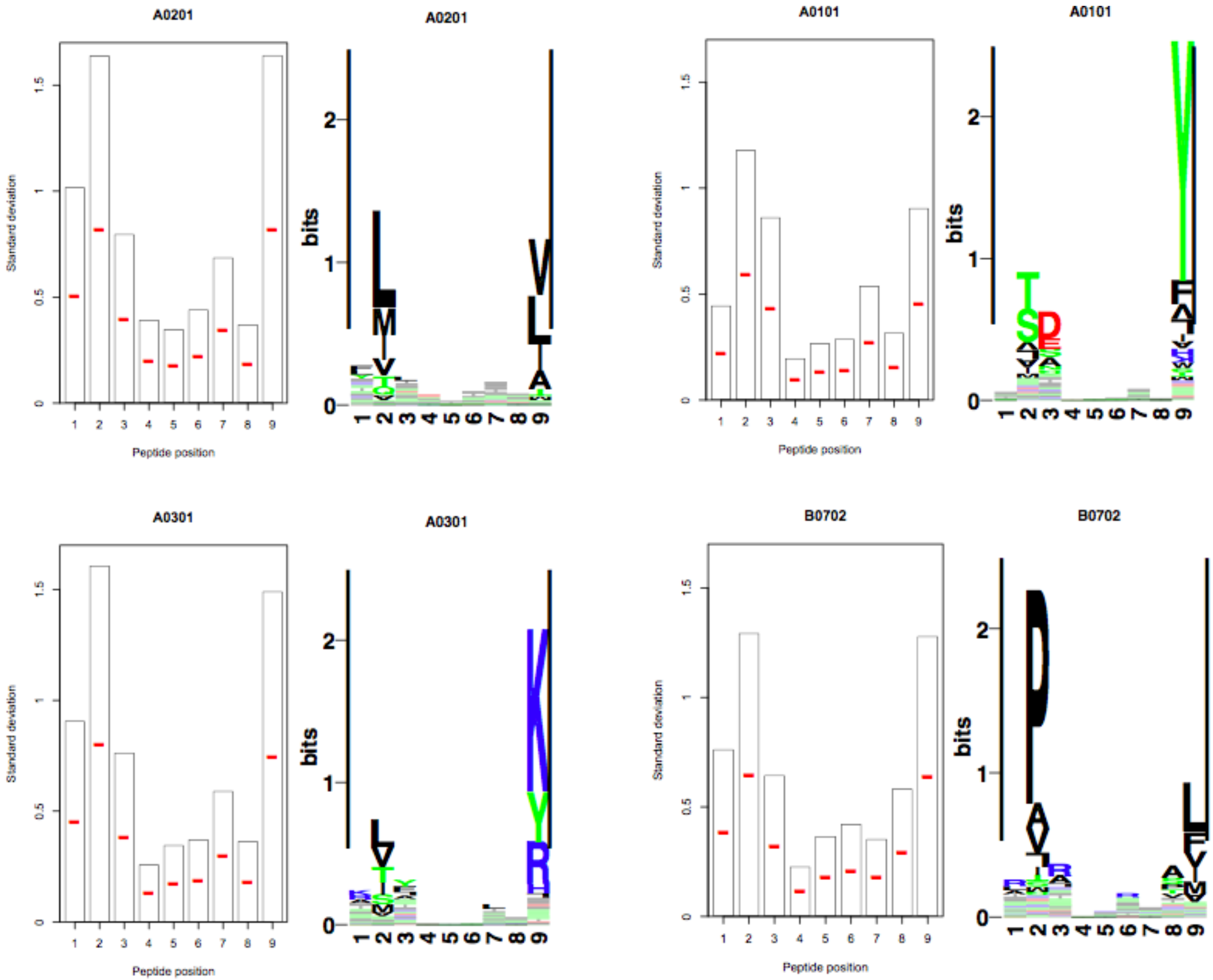
Figure 2.1. Two layer ANN with two amino acids as input, $p = \{1,2\}$, and one output neuron. The direction is downwards, and the graph is directed.



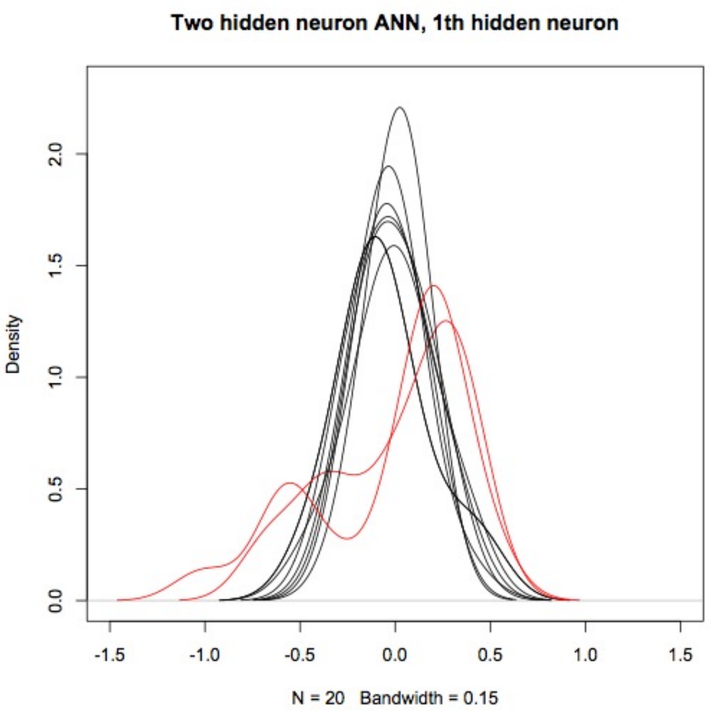


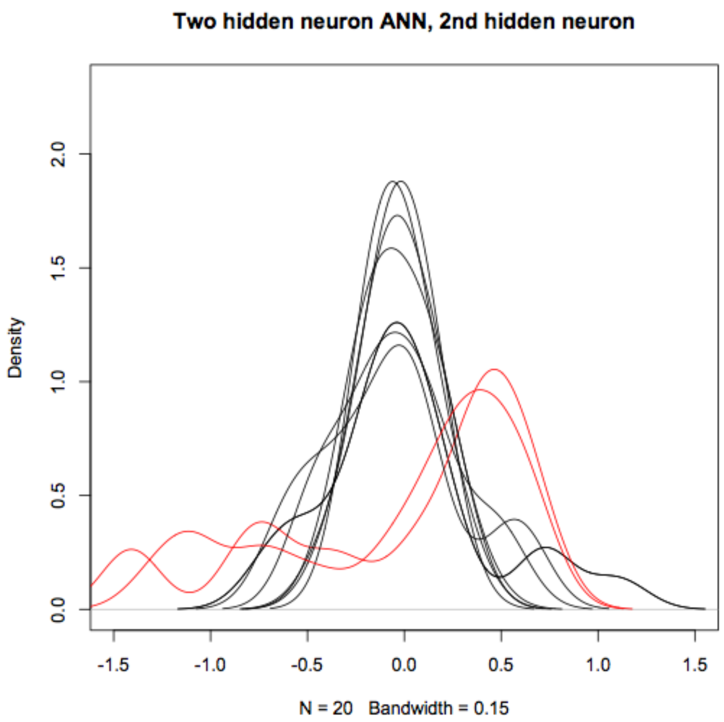
$$\delta_j^P = \sqrt{\frac{1}{N-1}\sum_A (v_{Aj}^P - \mu_j^P)^2}\ \text{, where}\ \mu_j^p = \frac{1}{N}\sum_A v_{Aj}^p$$

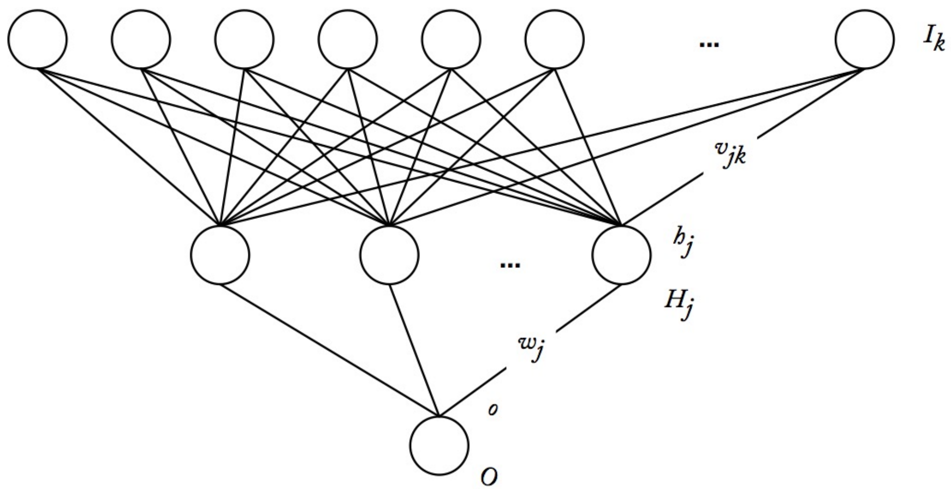
Making sense of ANN weights



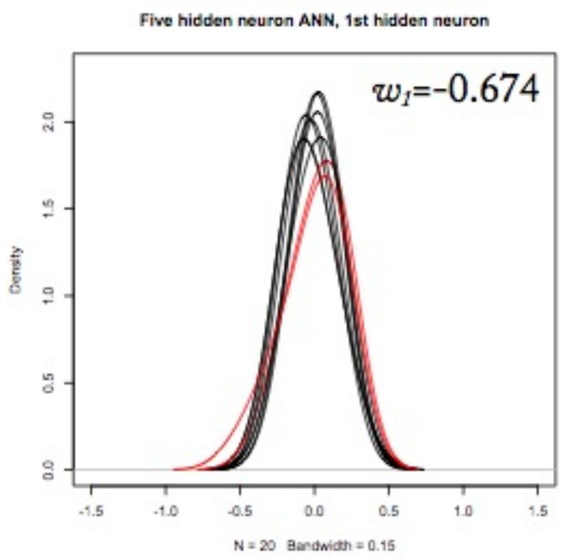
Making sense of ANN weights

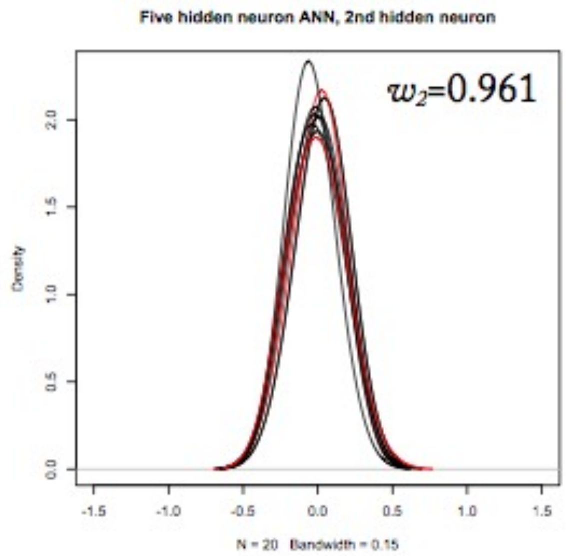


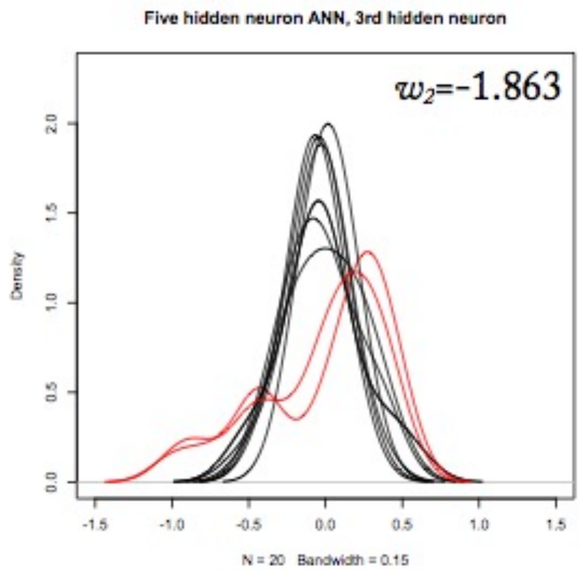


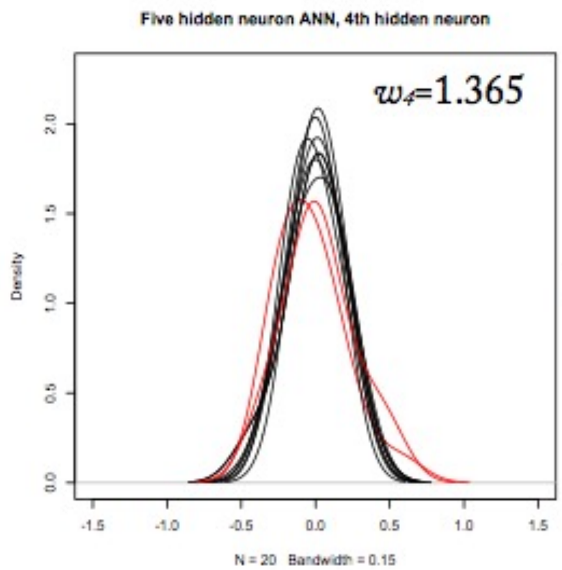


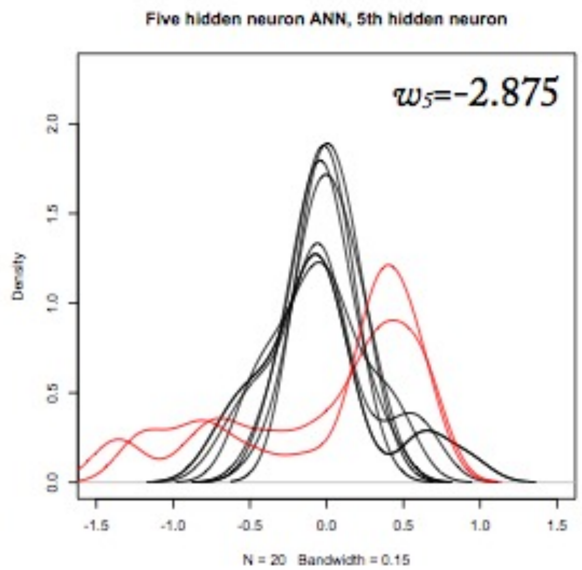
Making sense of ANN weights

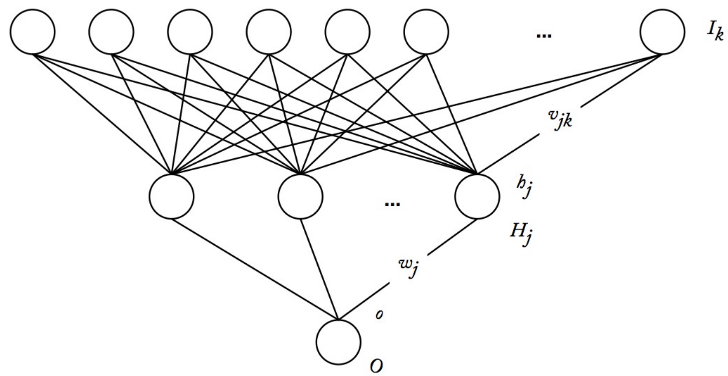




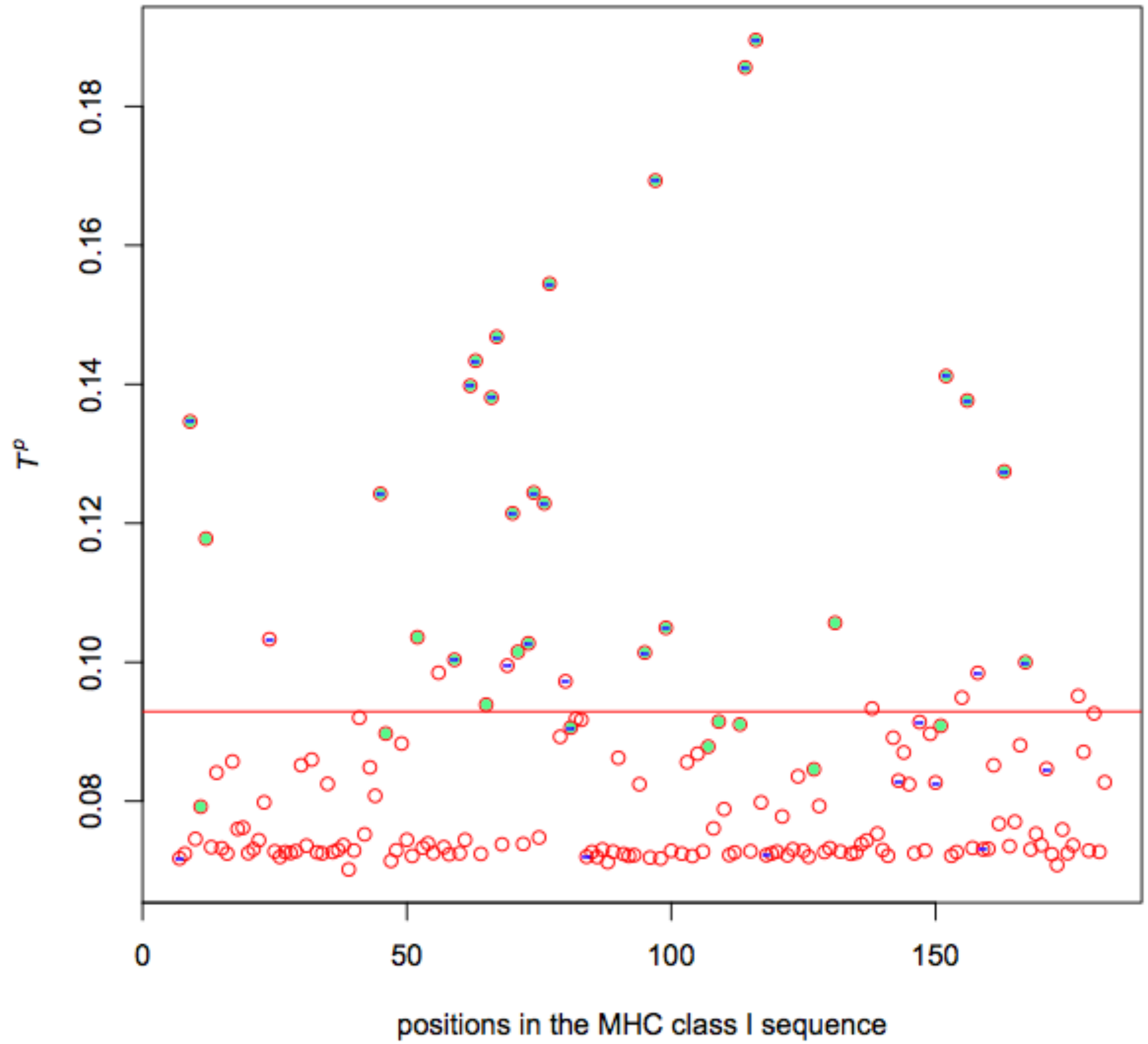


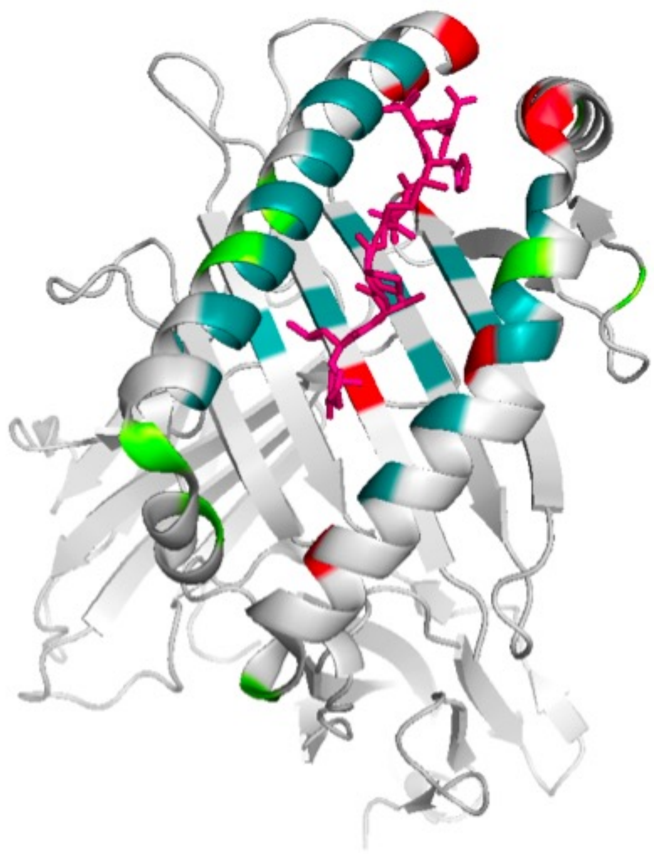






Making sense of ANN weights





Back Propagation

Advantages

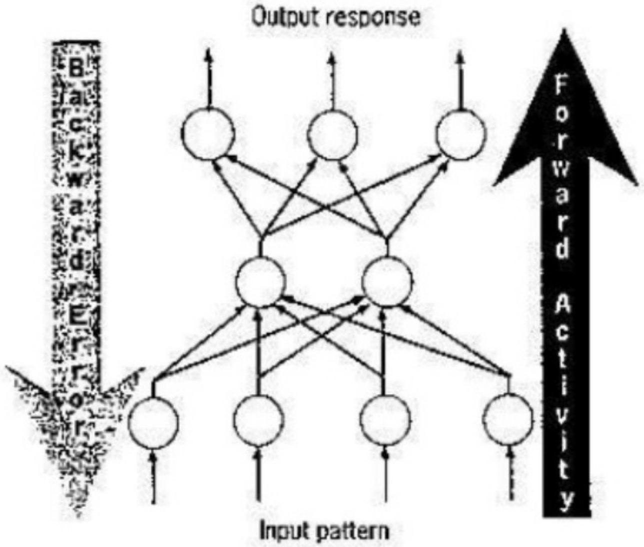
• Multi layer Perceptron network can be trained by the back propagation algorithm to perform any mapping between the input and the output.

What is wrong with back-propagation?

•It requires labeled training data.
  Almost all data is unlabeled.

•The learning time does not scale well
  It is very slow in networks with multiple hidden layers.

•It can get stuck in poor local optima.



A backpropagation network trains with a two-step procedure. The activity from the input pattern flows forward through the network, and the error signal flows backward to adjust the weights.

http://www.slideshare.net/hammawan/deep-neural-networks

Deeper(er) Network architecture

CENTERFORBIOLOGICALSEQUENCEANALYSIS CBS

$$E = \frac{1}{2} \cdot (O - t)^2$$

$$O = g(o), H = g(h)$$

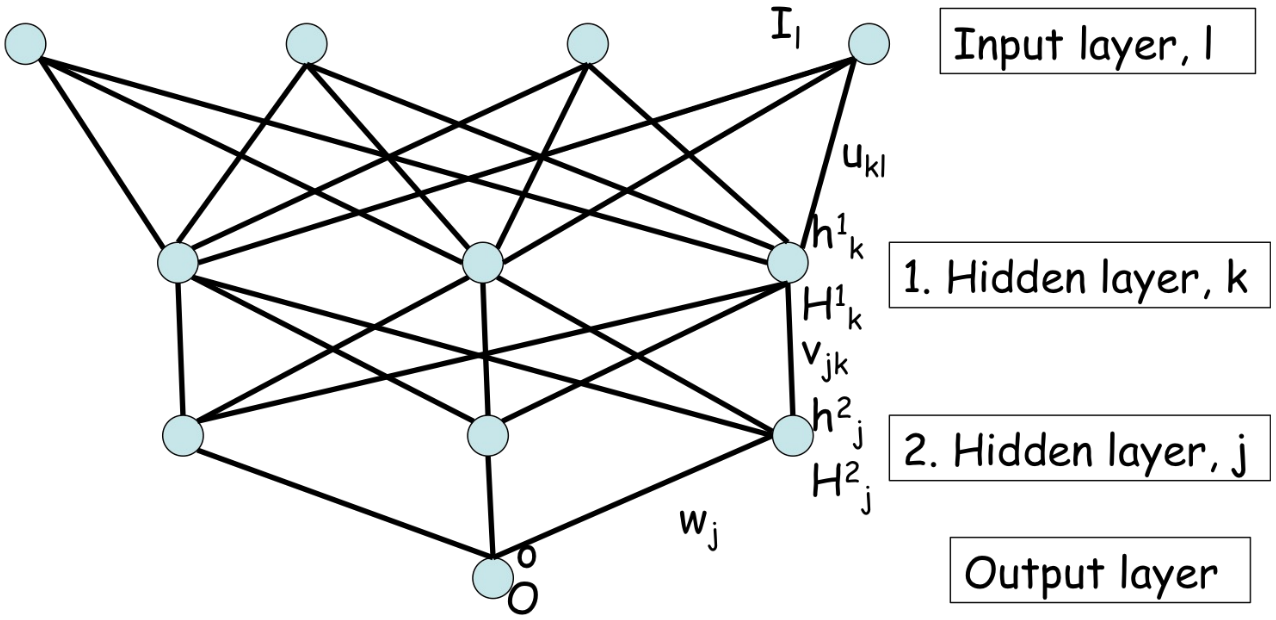
$$g(x) = \frac{1}{1 + e^{-x}}$$

$$o = \sum_j w_j \cdot H_j^2$$

$$h_j^2 = \sum_k v_{jk} \cdot H_k^1$$

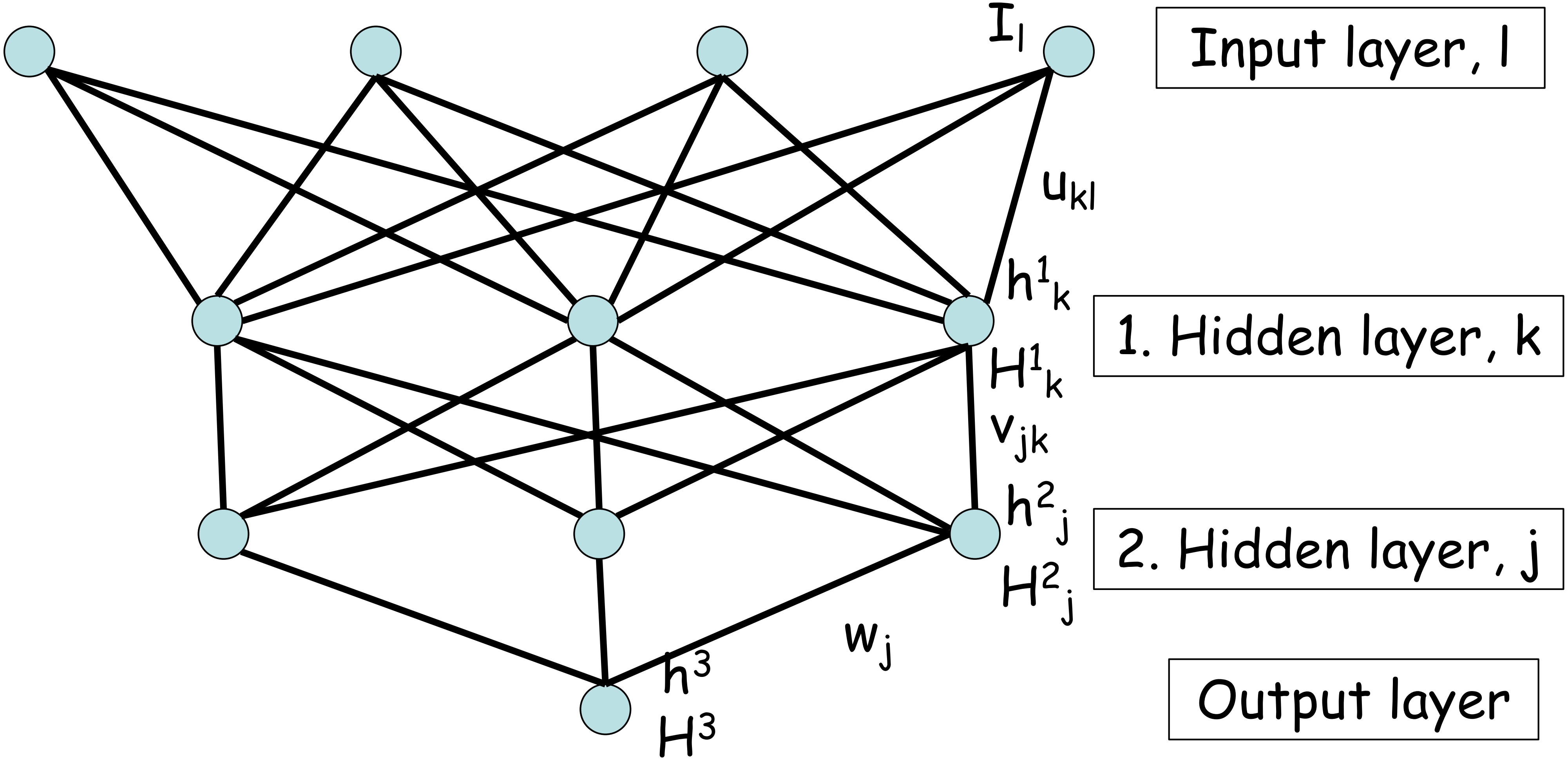
$$h_k^1 = \sum_l u_{kl} \cdot I_l$$

$$\Delta w_i = -\varepsilon \cdot \frac{\partial E}{\partial w_i}$$



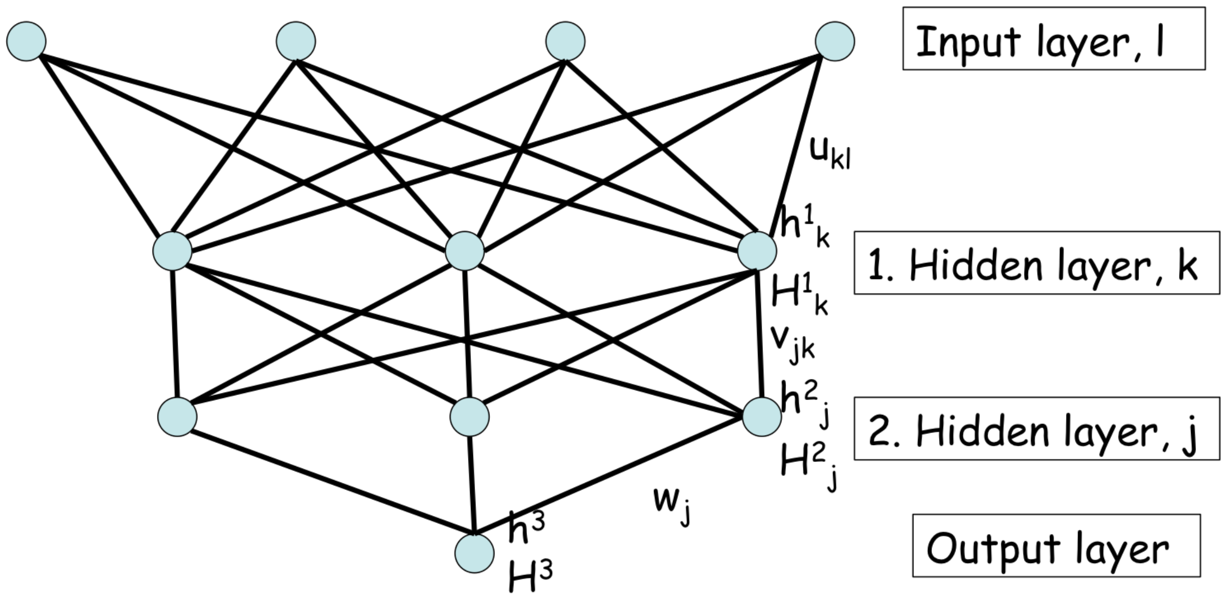
Deeper Network architecture

CENTERFORBIOLOGICALSEQUENCEANALYSIS CBS



$$\frac{\partial E}{\partial w_j} = \frac{\partial E(H^3(h^3(w_j)))}{\partial w_j} = \frac{\partial E}{\partial H^3} \cdot \frac{\partial H^3}{\partial h^3} \cdot \frac{\partial h^3}{\partial w_j} = (H^3 - t) \cdot g'(h^3) \cdot H^2_j$$

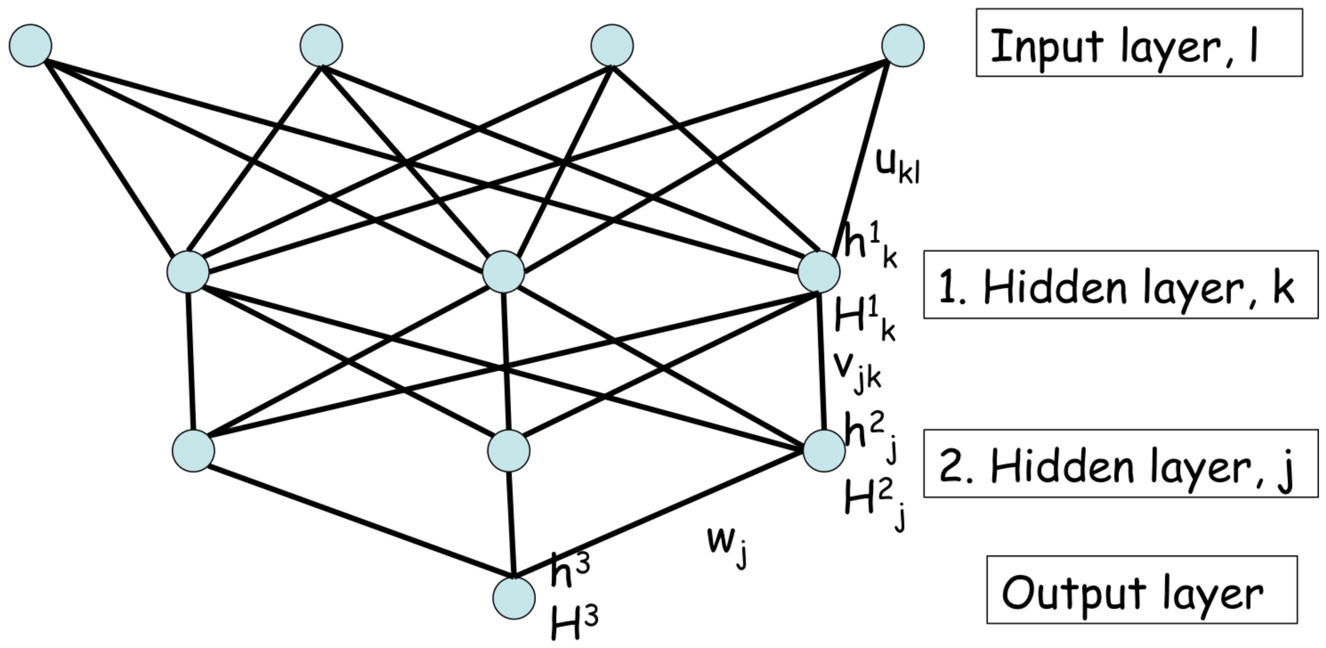
Network architecture (hidden to hidden)



$$\frac{\partial E}{\partial v_{jk}} = \frac{\partial E}{\partial H^3} \cdot \frac{\partial H^3}{\partial h^3} \cdot \frac{\partial h^3}{\partial H^2_j} \cdot \frac{\partial H^2_j}{\partial h^2_j} \cdot \frac{\partial h^2_j}{\partial v_{jk}}$$

$$= (H^3 - t) \cdot g'(h^3) \cdot w_j \cdot g'(h^2_j) \cdot H^1_k$$

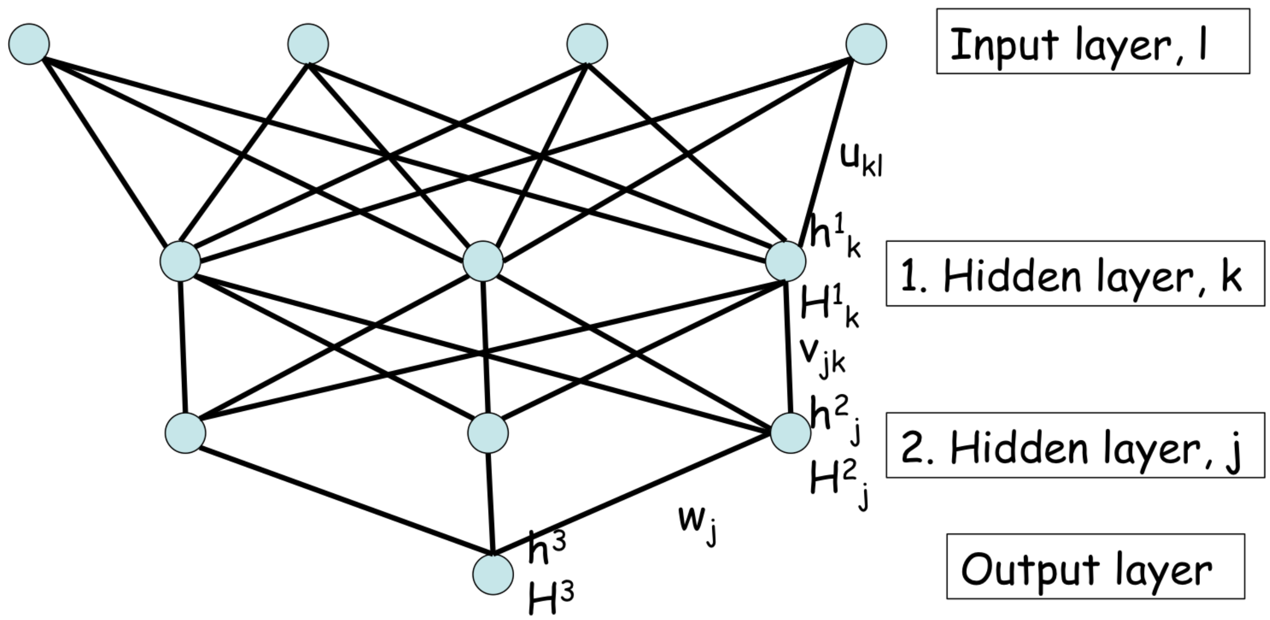
Network architecture (input to hidden)

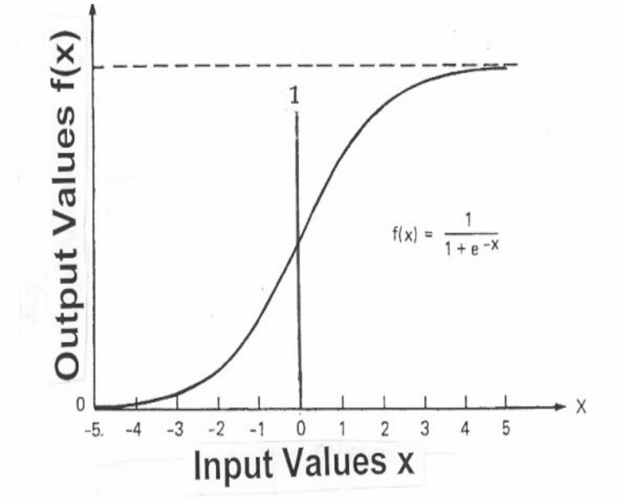


$$\frac{\partial E}{\partial u_{kl}} = \frac{\partial E}{\partial H^3} \cdot \frac{\partial H^3}{\partial h^3} \cdot \sum_j \frac{\partial h^3}{\partial H^2_j} \cdot \frac{\partial H^2_j}{\partial h^2_j} \cdot \frac{\partial h^2_j}{\partial H^1_k} \cdot \frac{\partial H^1_k}{\partial h^1_k} \cdot \frac{\partial h^1_k}{\partial u_{kl}}$$

$$= (H^3 - t) \cdot g'(h^3) \cdot \sum_j w_j \cdot g'(h^2_j) \cdot v_{jk} \cdot g'(h^1_k) \cdot I_l$$

Network architecture (input to hidden)

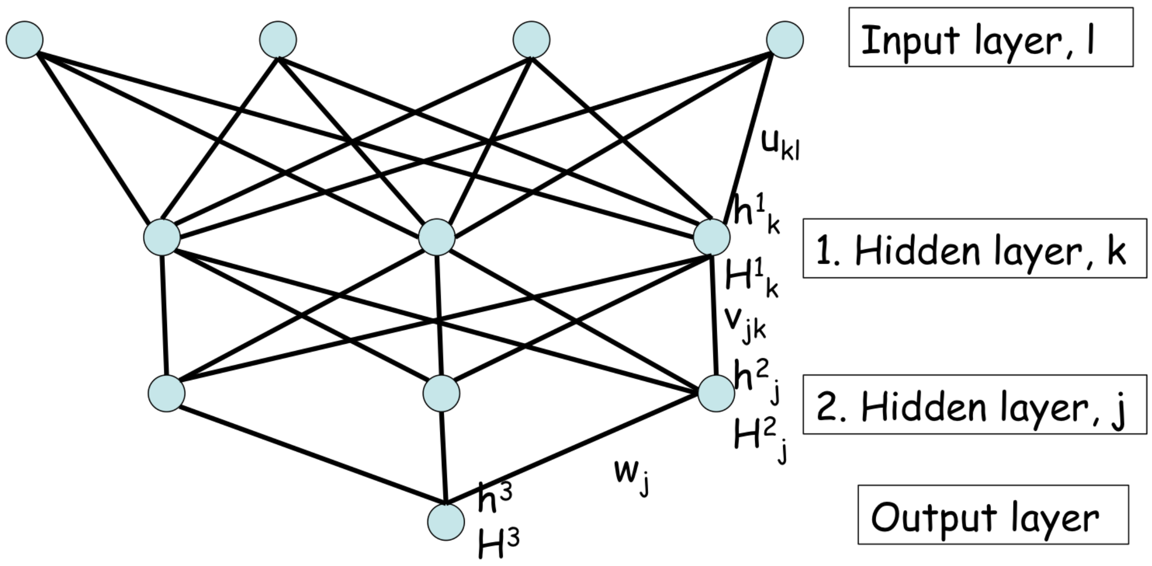


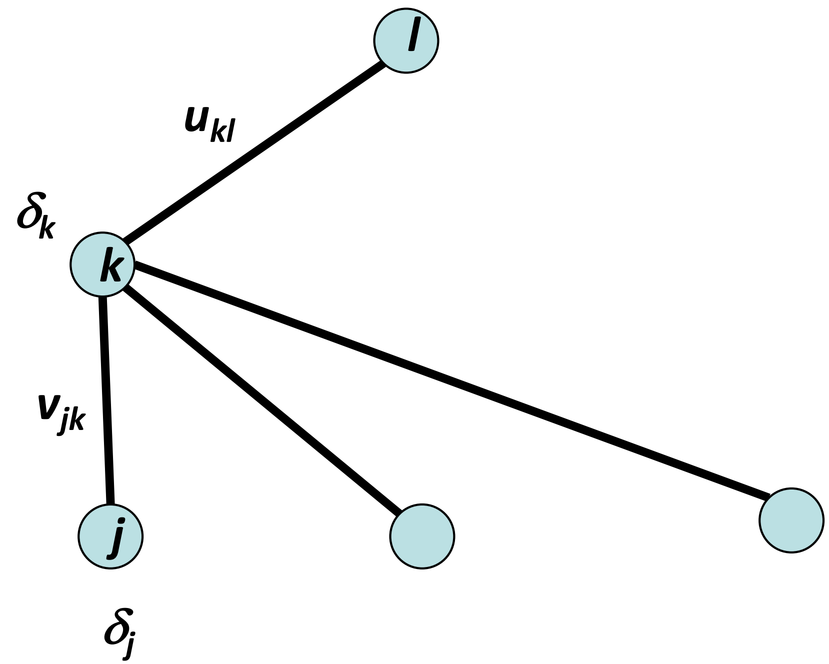


$$\frac{\partial E}{\partial u_{kl}} = \frac{\partial E}{\partial H^3} \cdot \frac{\partial H^3}{\partial h^3} \cdot \sum_j \frac{\partial h^3}{\partial H^2_j} \cdot \frac{\partial H^2_j}{\partial h^2_j} \cdot \frac{\partial h^2_j}{\partial H^1_k} \cdot \frac{\partial H^1_k}{\partial h^1_k} \cdot \frac{\partial h^1_k}{\partial u_{kl}}$$

$$= (H^3 - t) \cdot g'(h^3) \cdot \sum_j w_j \cdot g'(h^2_j) \cdot v_{jk} \cdot g'(h^1_k) \cdot I_l$$

Speed. Use delta's



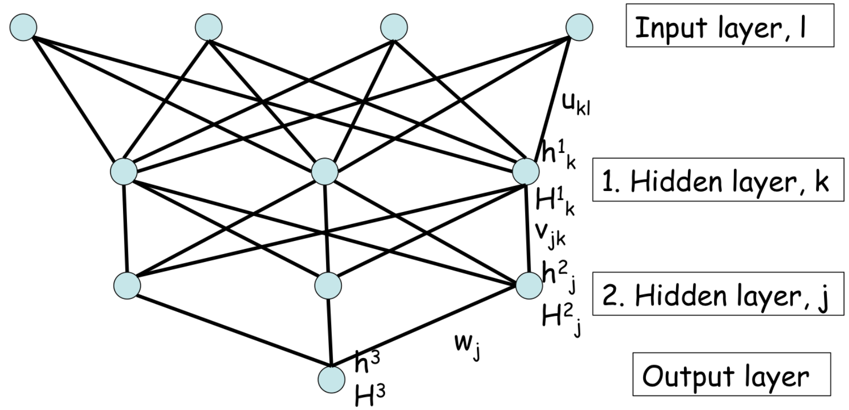


$$h_j^q = \sum_i w_{ji} H_i^{q-1}$$

$$H_i^{q-1} = g(h_i^{q-1})$$

Bishop, Christopher (1995). Neural networks for pattern recognition. Oxford: Clarendon Press. ISBN 0-19-853864-2.

Use delta's



$$h_j = \sum_i w_{ji} H_i$$

$$H_i = g(h_i)$$

$$\frac{\partial E}{\partial w^q_{ji}} = \frac{\partial E}{\partial h^q_j} \cdot \frac{\partial h^q_j}{\partial w^q_{ji}} = \delta^q_j \cdot H^{q-1}_i$$

$$\delta^q_j = \frac{\partial E}{\partial h^q_j}$$

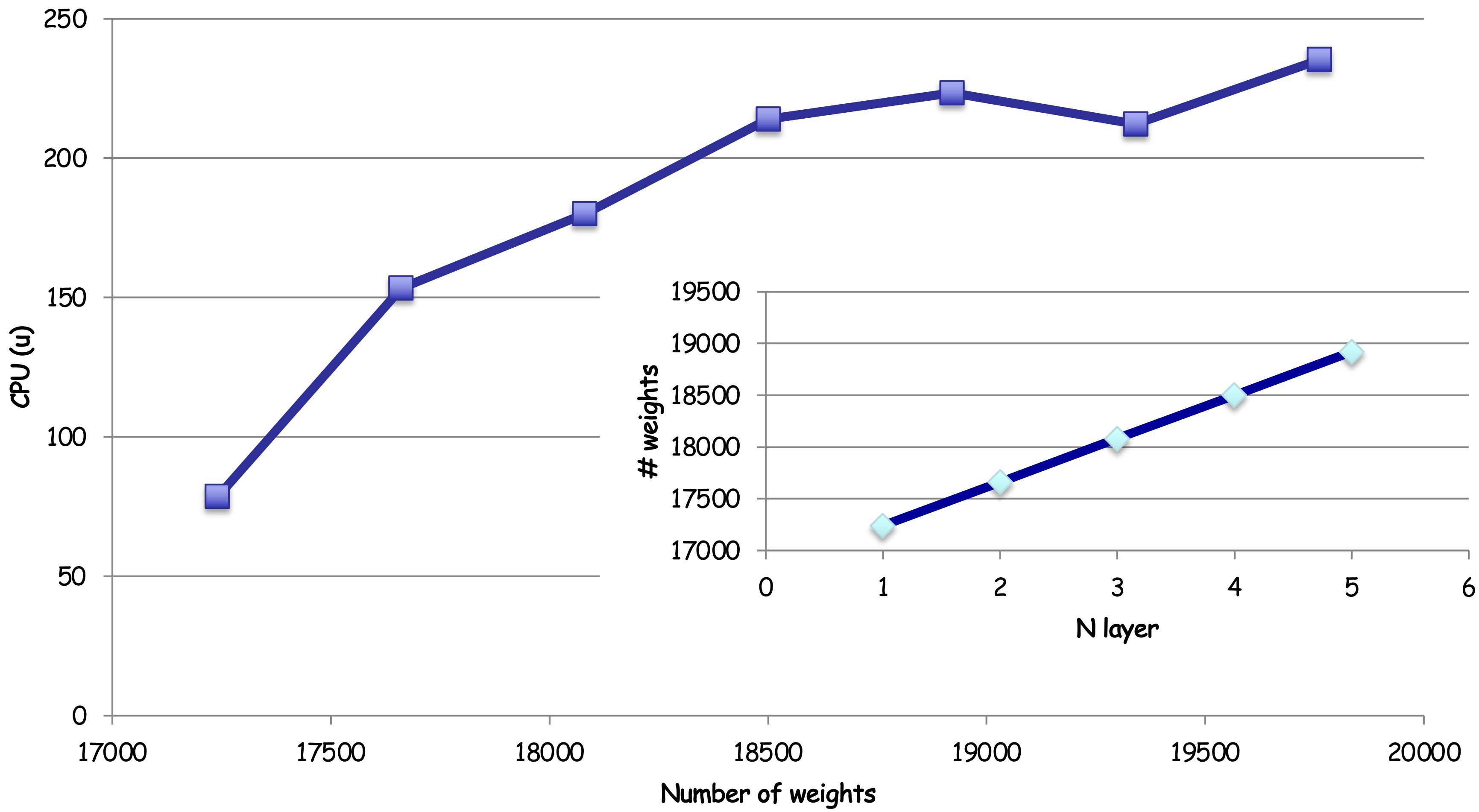
$$\delta^3 = \frac{\partial E}{\partial h^3} = \frac{\partial E}{\partial H^3} \cdot \frac{\partial H^3}{\partial h^3} = (H^3 - t) \cdot g'(h^3)$$

$$\delta^2_j = \frac{\partial E}{\partial h^2_j} = \frac{\partial E}{\partial h^3} \cdot \frac{\partial h^3}{\partial h^2_j} = \frac{\partial E}{\partial h^3} \cdot \frac{\partial h^3}{\partial H^2_j} \cdot \frac{\partial H^2_j}{\partial h^2_j} = g'(h^2_j) \cdot \delta^3 \cdot v_{jk}$$

$$\delta^1_k = \frac{\partial E}{\partial h^1_k} = \sum_j \frac{\partial E}{\partial h^2_j} \cdot \frac{\partial h^2_j}{\partial h^1_k} = \sum_j \frac{\partial E}{\partial h^2_j} \cdot \frac{\partial h^2_j}{\partial H^1_k} \cdot \frac{\partial H^1_k}{\partial h^1_k} = g'(h^1_k) \cdot \sum_j \delta^2_j \cdot v_{jk}$$

Deep learning - time is not an issue

CENTERFORBIOLOGICALSEQUENCEANALYSIS CBS



Deep learning

CENTERFO
RBIOLOGI
CALSEQU
ENCEANA
LYSIS CBS

Deep Neural Networks

- Standard learning strategy
  - Randomly initializing the weights of the network
  - Applying gradient descent using backpropagation
- But, backpropagation does not work well (if randomly initialized)
  - Deep networks trained with back-propagation (without unsupervised pre-train) perform worse than shallow networks
  - ANN have limited to one or two layers

http://www.slideshare.net/hammawan/deep-neural-networks

Deep learning

Recent Deep Learning Highlights

- Google Goggles uses ***Stacked Sparse Auto Encoders*** (Hartmut Neven @ ICML 2011)
- The monograph or review paper ***Learning Deep Architectures for AI*** (Foundations & Trends in Machine Learning, 2009).
- ***Exploring Strategies for Training Deep Neural Networks***, Hugo Larochelle, Yoshua Bengio, Jerome Louradour and Pascal Lamblin in: The Journal of Machine Learning Research, pages 1-40, 2009.
- The LISA publications database contains ***a deep architectures category***. http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/ReadingOnDeepNetworks
- Deep Machine Learning – ***A New Frontier in Artificial Intelligence Research*** – a survey paper by Itamar Arel, Derek C. Rose, and Thomas P. Karnowski.

http://www.slideshare.net/hammawan/deep-neural-networks