CS217: Artificial Intelligence and Machine Learning (associated lab: CS240)

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

Week4 of 27jan25, Perceptron capacity, FFNN

Main points covered: week3 of 20jan25

Monotonicity

- A heuristic *h(p)* is said to satisfy the monotone restriction, if for all *'p'*, *h(p)<=h(p_c)+cost(p, p_c)*, where *'p_c'* is the child of *'p'*.

Theorem

- If monotone restriction (also called triangular inequality) is satisfied, then for nodes in the closed list, redirection of parent pointer is not necessary. In other words, if any node *'n'* is chosen for expansion from the open list, then $g(n)=g^*(n)$, where $g(n)$ is the cost of the path from the start node *'s'* to *'n'* at that point of the search when *'n'* is chosen, and $g^*(n)$ is the cost of the optimal path from *'s'* to *'n'*

Relationship between Monotonicity and Admissibility

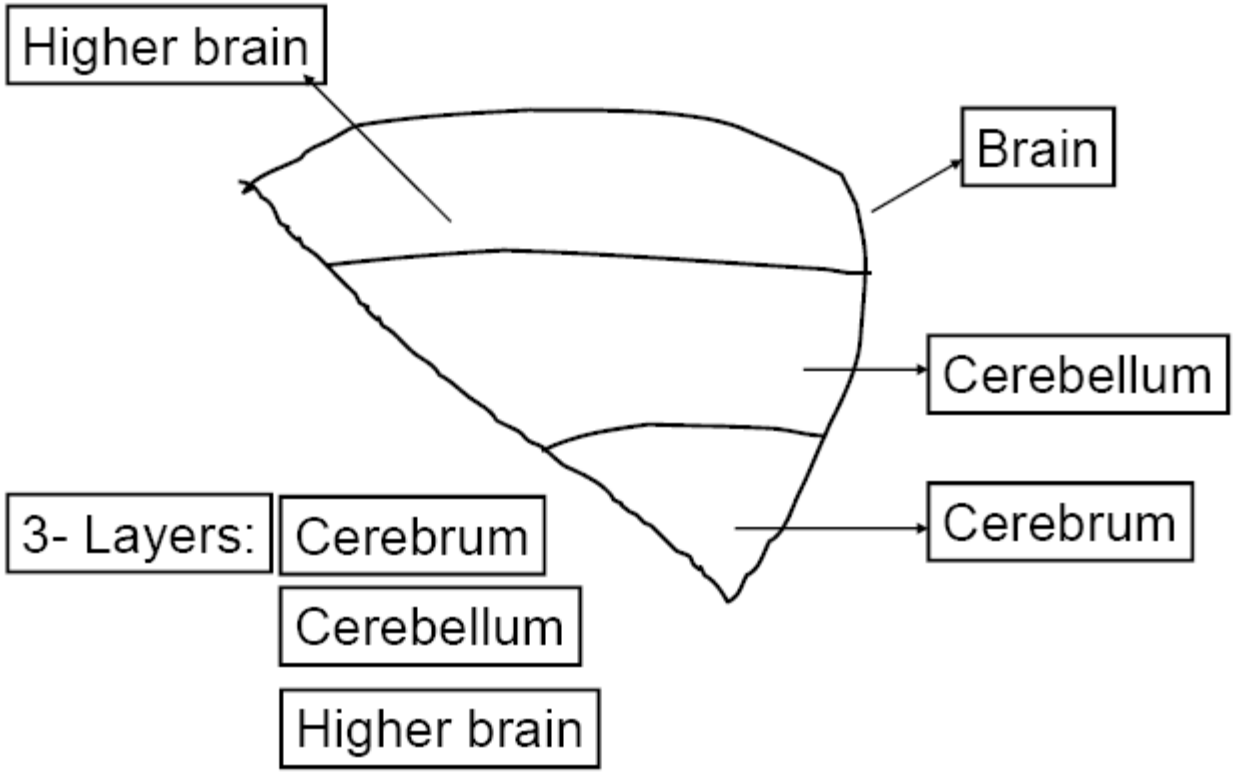
- Observation:

  Monotone Restriction $\rightarrow$ Admissibility

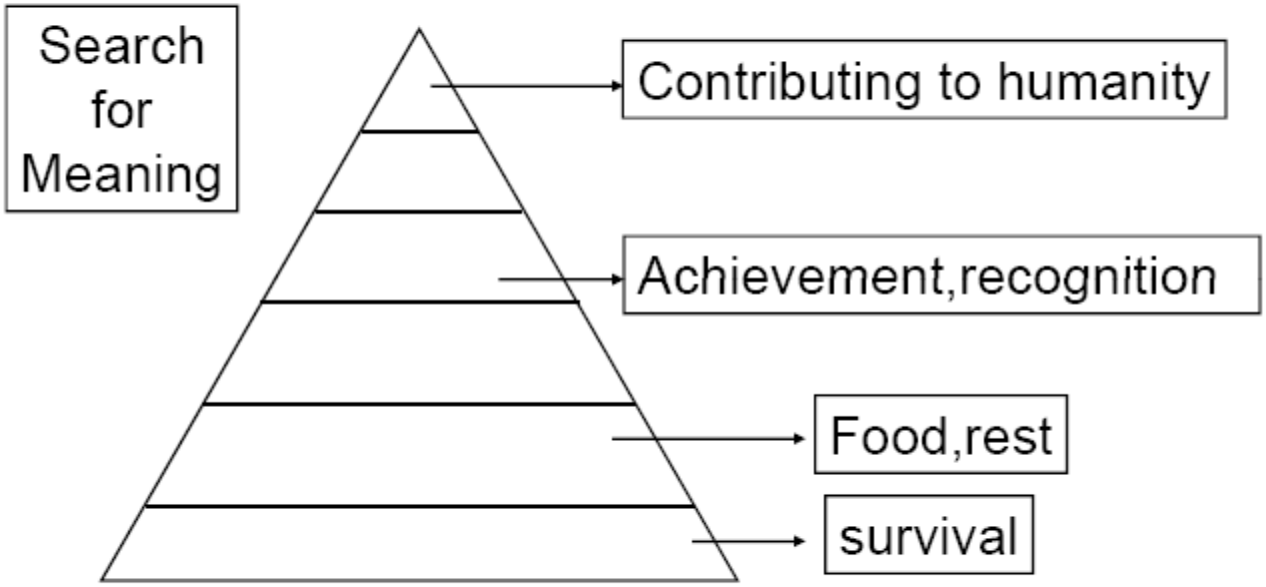
  but not vice-versa

- Statement: *If* $h(n_i) <= h(n_j) + c(n_i, n_j)$ *for all i, j*

  then $h(n_i) <= h^*(n_i)$ *for all i*

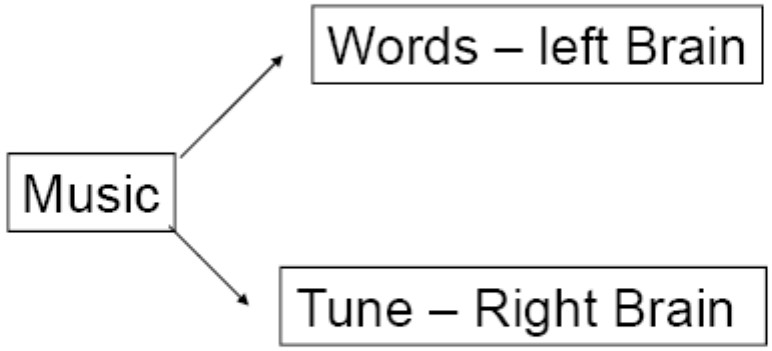

Higher brain
Brain
Cerebellum
Cerebrum
3- Layers:
Cerebrum
Cerebellum
Higher brain

Maslow's hierarchy



Left Brain – Logic, Reasoning, Verbal ability

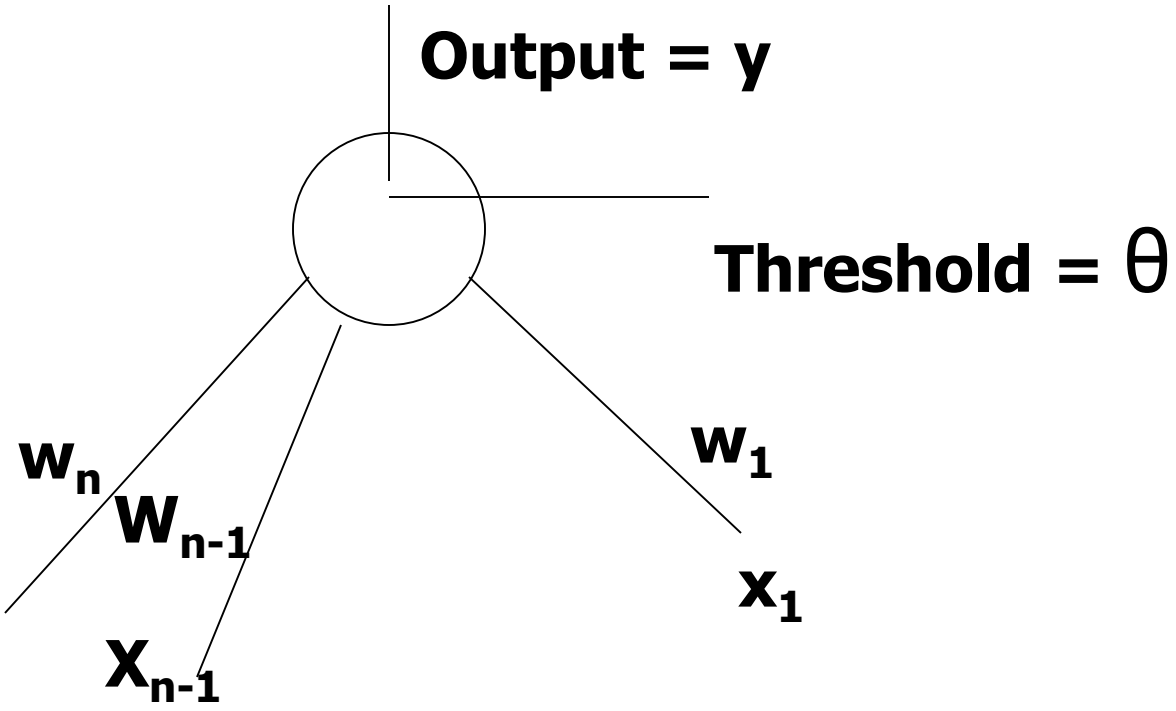
Right Brain – Emotion, Creativity



Maps in the brain. Limbs are mapped to brain

The Perceptron Model

A perceptron is a computing element with input lines having associated weights and the cell having a threshold value. The perceptron model is motivated by the biological neuron.



Perceptron Training Algorithm

1. Start with a random value of w

   ex: <0,0,0…>
2. Test for $wx_i > 0$

   If the test succeeds for i=1,2,…n

   then return w
3. Modify w, $w_{next} = w_{prev} + x_{fail}$

Convergence of PTA

- Statement:

  Whatever be the initial choice of weights and whatever be the vector chosen for testing, PTA converges if the vectors are from a linearly separable function.

End of main points

Perceptrons and their computing power

Fundamental Observation

- The number of TFs computable by a perceptron is equal to the number of regions produced by 2^n hyper-planes,obtained by plugging in the values $<x_1,x_2,x_3,\ldots,x_n>$ in the equation

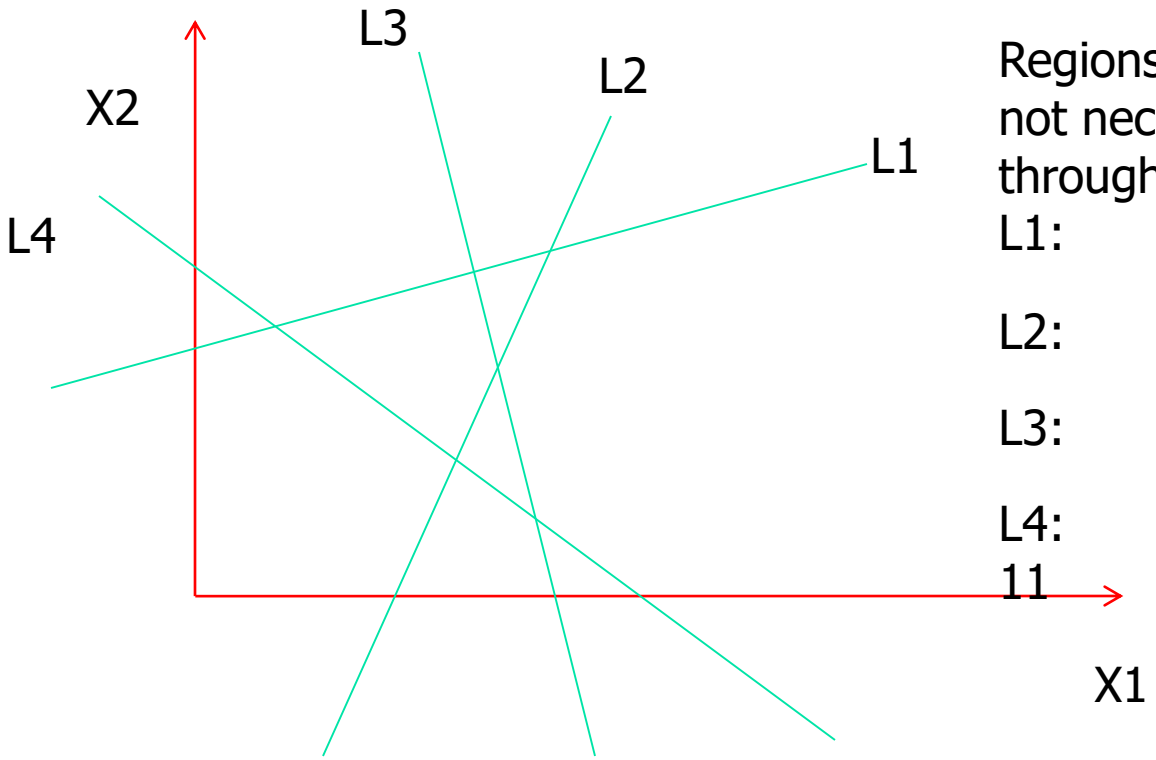
$$\sum_{i=1}^{n} w_i x_i = \theta$$

The geometrical observation

- **Problem:** *m* linear surfaces called hyper-planes (each hyper-plane is of *(d-1)*-dim) in d-dim, then what is the max. no. of regions produced by their intersection?

  i.e., $R_{m,d} = ?$

Regions produced by lines



Regions produced by lines not necessarily passing through origin

L1: 2

L2: 2+2 = 4

L3: 2+2+3 = 7

L4: 2+2+3+4 = 11

New regions created = Number of intersections on the incoming line by the original lines

Total number of regions = Original number of regions + New regions created

Number of computable functions by a neuron

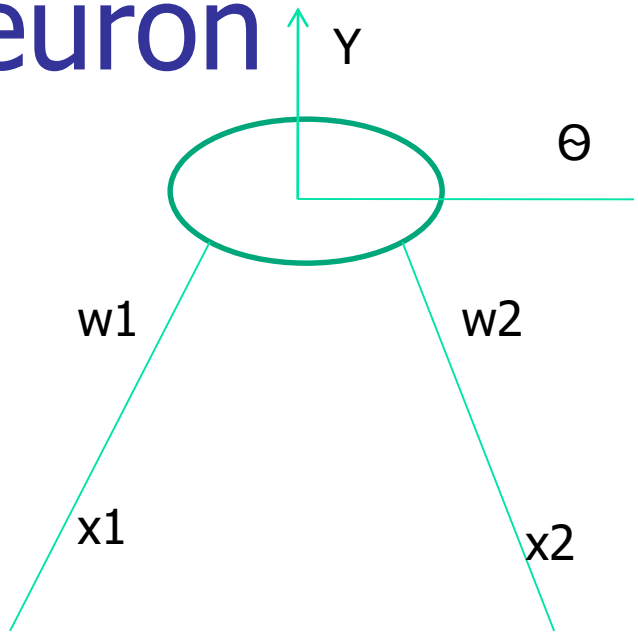
$$w1 * x1 + w2 * x2 = \theta$$

$$(0,0) \Rightarrow \theta = 0 : P1$$

$$(0,1) \Rightarrow w2 = \theta : P2$$

$$(1,0) \Rightarrow w1 = \theta : P3$$

$$(1,1) \Rightarrow w1 + w2 = \theta : P4$$

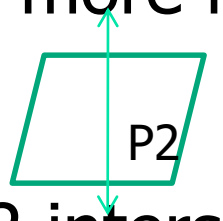


P1, P2, P3 and P4 are planes in the <W1,W2, Θ> space

Number of computable functions by a neuron (cont…)

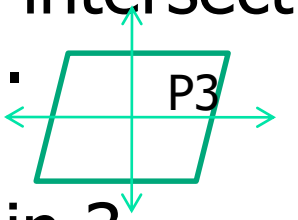
- P1 produces 2 regions
- P2 is intersected by P1 in a line. 2 more new regions are produced.

  Number of regions = 2+2 = 4



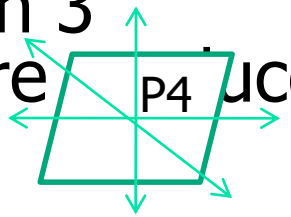
- P3 is intersected by P1 and P2 in 2 intersecting lines. 4 more regions are produced.

  Number of regions = 4 + 4 = 8



- P4 is intersected by P1, P2 and P3 in 3 intersecting lines. 6 more regions are produced.

  Number of regions = 8 + 6 = 14

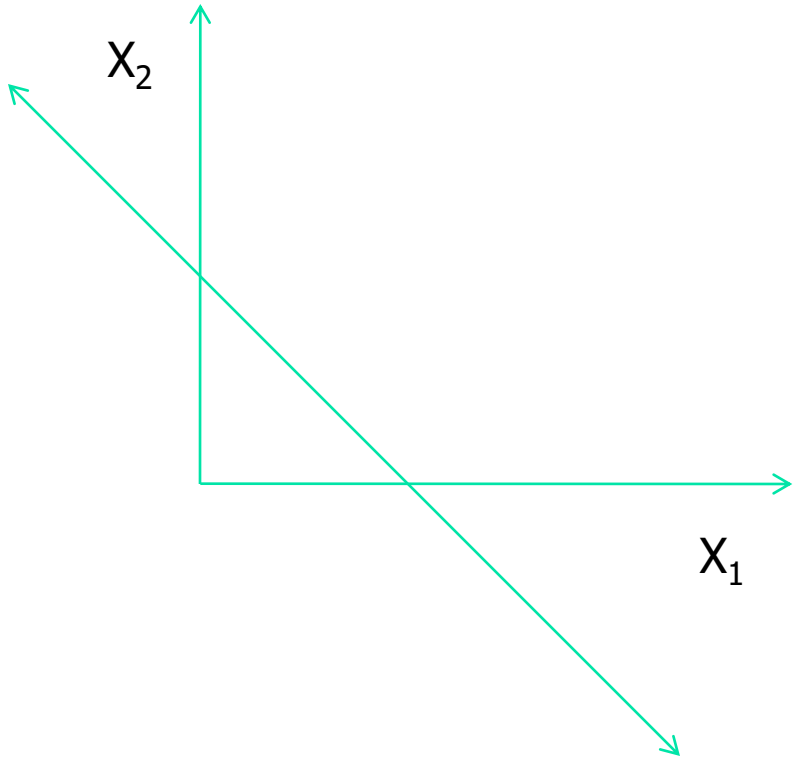


- Thus, a single neuron can compute 14 Boolean functions which are linearly separable.

Points in the same region

If
$W_1 * X_1 + W_2 * X_2 > \Theta$
$W_1' * X_1 + W_2' * X_2 > \Theta'$
Then

If $<W_1, W_2, \Theta>$ and $<W_1', W_2', \Theta'>$ share a region then they compute the same function



No. of Regions produced by Hyperplanes

Number of regions founded by n hyperplanes in d-dim passing through origin is given by the following recurrence relation

$$R_{n,\,d} = R_{n\,-\,1,\,d} + R_{n\,-\,1,\,d\,-\,1}$$

we use generating function as an operating function

Boundary condition:

$R_{1,\,d} = 2$ 1 hyperplane in d-dim

$R_{n,1} = 2$ n hyperplanes in 1-dim,

Reduce to n points thru origin

The generating function is

$$f(x, y) = \sum_{n=1}^{\infty} \sum_{d=1}^{\infty} R_{n,\,d} \cdot x^n y^d$$

From the recurrence relation we have,

$$R_{n,\,d} - R_{n-1,\,d} - R_{n-1,\,d-1} = 0$$

$R_{n-1,d}$ corresponds to 'shifting' n by 1 place, => multiplication by x

$R_{n-1,d-1}$ corresponds to 'shifting' n and d by 1 place => multiplication by xy

On expanding $f(x,y)$ we get

$$\begin{aligned} f(x,y) = {} & R_{1,1}\cdot xy + R_{1,2}\cdot x\,y^2 + R_{1,3}\cdot x\,y^3 + \ldots + R_{1,\,d}\cdot x\,y^d + \ldots.\infty \\ & + R_{2,1}\cdot x^2 y + R_{2,2}\cdot x^2 y^2 + R_{2,3}\cdot x^2 y^3 + \ldots + R_{2,\,d}\cdot x^2 y^d + \ldots.\infty \\ & \ldots\ldots \\ & + R_{n,1}\cdot x^n y + R_{n,2}\cdot x^n y^2 + R_{n,3}\cdot x^n y^3 + \ldots + R_{n,\,d}\cdot x^n y^d + \ldots.\infty \end{aligned}$$

$$f(x,y) = \sum_{n=1}^{\infty}\sum_{d=1}^{\infty} R_{n,d} \cdot x^n y^d$$

$$x \cdot f(x,y) = \sum_{n=1}^{\infty}\sum_{d=1}^{\infty} R_{n,d} \cdot x^{n+1} y^d = \sum_{n=2}^{\infty}\sum_{d=1}^{\infty} R_{n-1,d} \cdot x^n y^d$$

$$xy \cdot f(x,y) = \sum_{n=1}^{\infty}\sum_{d=1}^{\infty} R_{n,d} \cdot x^{n+1} y^{d+1} = \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} R_{n-1,d-1} \cdot x^n y^d$$

$$x \cdot f(x,y) = \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} R_{n-1,d} \cdot x^n y^d + \sum_{n=2}^{\infty} R_{n-1,1} \cdot x^n y$$

$$= \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} R_{n-1,d} \cdot x^n y^d + 2 \cdot \sum_{n=2}^{\infty} x^n y$$

$$f(x,y) = \sum_{n=1}^{\infty}\sum_{d=1}^{\infty} R_{n,d} \cdot x^n y^d$$

$$= \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} R_{n,d} \cdot x^n y^d + \sum_{d=1}^{\infty} R_{1,d} \cdot xy^d + \sum_{n=1}^{\infty} R_{n,1} \cdot x^n y - R_{1,1} \cdot xy$$

$$= \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} R_{n,d} \cdot x^n y^d + 2x \cdot \sum_{d=1}^{\infty} xy^d + 2y \cdot \sum_{n=1}^{\infty} x^n y - 2xy$$

After all this expansion,

$$f(x,y) - x \cdot f(x,y) - xy \cdot f(x,y)$$

$$= \sum_{n=2}^{\infty}\sum_{d=2}^{\infty} (R_{n,d} - R_{n-1,d} - R_{n-1,d-1}) x^n y^d$$

$$+ 2y \cdot \sum_{n=1}^{\infty} x^d - 2xy - 2y \cdot \sum_{n=2}^{\infty} x + 2x \cdot \sum_{d=1}^{\infty} y^d$$

$$= 2x \cdot \sum_{d=1}^{\infty} y^d$$

since other two terms become zero

This implies

$$[1-x-xy]\,f(x,y)=2x\cdot\sum_{d=1}^{\infty}y^{d}$$

$$f(x,y)=\frac{1}{[1-x(1+y)]}\cdot 2x\cdot\sum_{d=1}^{\infty}y^{d}$$

$$=2x\cdot[y+y^{2}+y^{3}+\ldots+y^{d}+\ldots..\infty]\cdot$$

$$[1+x(1+y)+x^{2}(1+y)^{2}+\ldots+x^{d}(1+y)^{d}+\ldots..\infty]$$

also we have,

$$f(x,y)=\sum_{n=1}^{\infty}\sum_{d=1}^{\infty}R_{n,\,d}\cdot x^{n}y^{d}$$

Comparing coefficients of each term in RHS we get,

Comparing co-efficients we get

$$R_{n,\,d} = 2\sum_{i=0}^{d-1} C_i^{n-1}$$

Implication

Important.

- *R(n,d)* becomes for a perceptron with *m* weights and 1 threshold $R(2^m, m+1)$

$$= 2\sum_{i=0}^{m+1-1} C_i^{2^m-1}$$

$$= 2\sum_{i=0}^{m} C_i^{2^m-1}$$

$$= O(2^{m^2})$$

- Total no of Boolean Function is $2^{2\text{^m}}$. Shows why #TF << #BF

PTA convergence

Statement of Convergence of PTA

- Statement:

  Whatever be the initial choice of weights and whatever be the vector chosen for testing, PTA converges if the vectors are from a linearly separable function.

Proof of Convergence of PTA

- Suppose w_n is the weight vector at the n^{th} step of the algorithm.
- At the beginning, the weight vector is w_0
- Go from w_i to w_{i+1} when a vector X_j fails the test $w_i X_j > 0$ and update w_i as

$$w_{i+1} = w_i + X_j$$

- Since Xjs form a linearly separable function,

$\exists$ w* s.t. $w^* X_j > 0$ $\forall j$

Proof of Convergence of PTA (cntd.)

- Consider the expression

$$G(w_n) = \frac{w_n \,.\, w^*}{|w_n|}$$

where w_n = weight at nth iteration

- $G(w_n) = \frac{|w_n| \,.\, |w^*| \,.\, \cos\theta}{|w_n|}$

where θ = angle between w_n and w^*

- $G(w_n) = |w^*| \,.\, \cos\theta$
- $G(w_n) \leq |w^*|$ (as $-1 \leq \cos\theta \leq 1$)

Behavior of Numerator of G

$w_n \cdot w^* = (w_{n-1} + X^{n-1}_{fail}) \cdot w^*$

$= w_{n-1} \cdot w^* + X^{n-1}_{fail} \cdot w^*$

$= (w_{n-2} + X^{n-2}_{fail}) \cdot w^* + X^{n-1}_{fail} \cdot w^* \ldots..$

$= w_0 \cdot w^* + (X^0_{fail} + X^1_{fail} + \ldots. + X^{n-1}_{fail}). w^*$

$w^*.X^i_{fail}$ is always positive: note carefully

- Suppose $|X_j| \geq \delta$, where δ is the minimum magnitude.
- Num of G $\geq |w_0 \cdot w^*| + n\,\delta \cdot |w^*|$
- So, numerator of G grows with n.

Behavior of Denominator of G

- $|w_n| = \sqrt{w_n \cdot w_n}$
- $= \sqrt{(w_{n-1} + X^{n-1}_{fail})^2}$
- $= \sqrt{(w_{n-1})^2 + 2.\, w_{n-1}.\, X^{n-1}_{fail} + (X^{n-1}_{fail})^2}$
- $\leq \sqrt{(w_{n-1})^2 + (X^{n-1}_{fail})^2}$ (as $w_{n-1}.\, X^{n-1}_{fail} \leq 0$)
- $\leq \sqrt{(w_0)^2 + (X^0_{fail})^2 + (X^1_{fail})^2 + \ldots + (X^{n-1}_{fail})^2}$

- $|X_j| \leq \rho$ (max magnitude)
- So, Denom $\leq \sqrt{(w_0)^2 + n\rho^2}$

Some Observations

- Numerator of G grows as n
- Denominator of G grows as $\sqrt{n}$

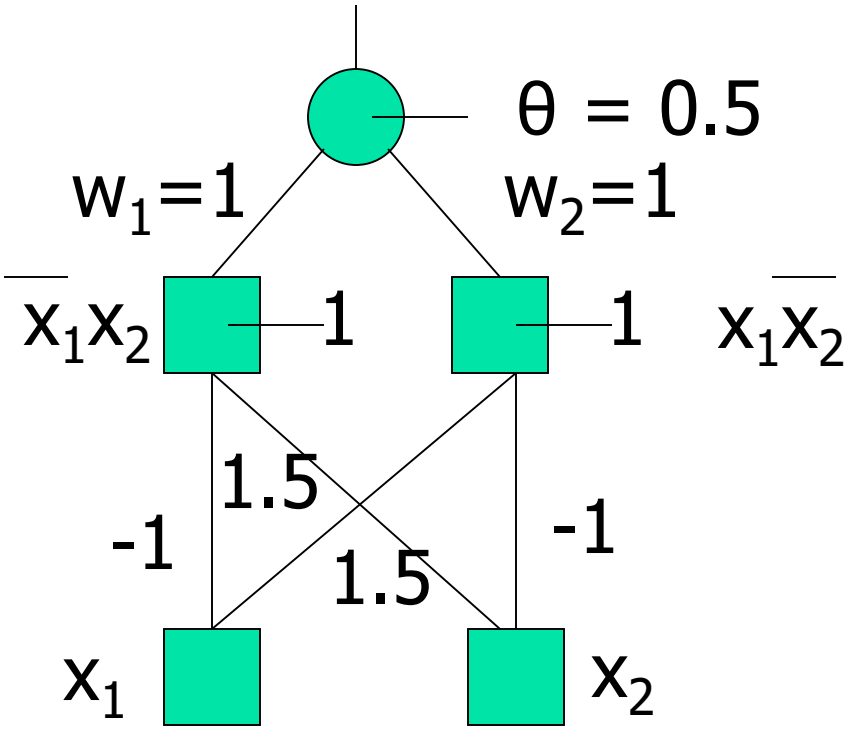
  => Numerator grows faster than denominator
- If PTA does not terminate, $G(w_n)$ values will become unbounded.

Some Observations contd.

- But, as $|G(w_n)| \leq |w^*|$ which is finite, this is impossible!
- Hence, PTA has to converge.
- Proof is due to Marvin Minsky.

Feedforward Network and Backpropagation

Example - XOR



Gradient Descent Technique

- Let E be the error at the output layer

$$E = \frac{1}{2}\sum_{j=1}^{p}\sum_{i=1}^{n}(t_i - o_i)_j^2$$

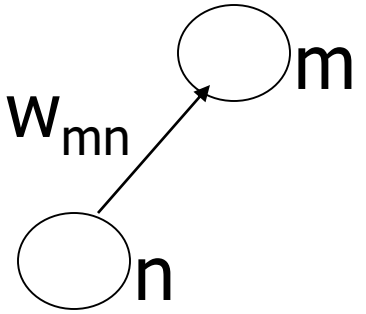
p is here number of inputs.

- t_i = target output; o_i = observed output

- i is the index going over n neurons in the outermost layer
- j is the index going over the p patterns (1 to p)
- Ex: XOR:– p=4 and n=1

Weights in a FF NN

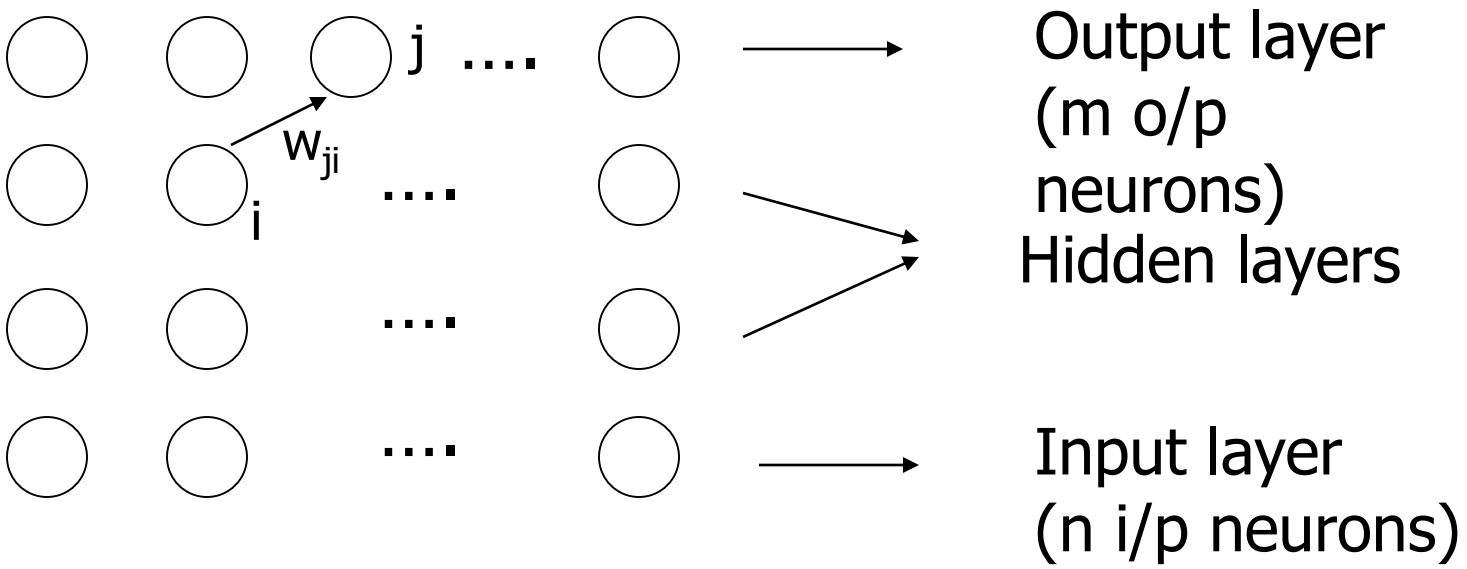
- w_{mn} is the weight of the connection from the n^{th} neuron to the m^{th} neuron
- E *vs* $\overline{W}$ surface is a complex surface in the space defined by the weights w_{ij}
- $-\frac{\delta E}{\delta w_{mn}}$ gives the direction in which a movement of the operating point in the w_{mn} co-ordinate space will result in maximum decrease in error



Important.

$$\Delta w_{mn} \propto -\frac{\delta E}{\delta w_{mn}}$$

Backpropagation algorithm



- Fully connected feed forward network
- Pure FF network (no jumping of connections over layers)

Gradient Descent Equations

$$\Delta w_{ji} = -\eta \frac{\delta E}{\delta w_{ji}} (\eta = \text{learning rate}, 0 \leq \eta \leq 1)$$

$$\frac{\delta E}{\delta w_{ji}} = \frac{\delta E}{\delta net_j} \times \frac{\delta net_j}{\delta w_{ji}} (net_j = \text{input at the j}^{th} \text{ neuron})$$

Important!

$$\frac{\delta E}{\delta net_j} = -\delta j$$

Hidden layers we don't know the errors as label is not known, we backpropagate the errors.

$$\Delta w_{ji} = \eta \delta j \frac{\delta net_j}{\delta w_{ji}} = \eta \delta j o_i$$

Nobel prize 2024.

Backpropagation – for outermost layer

$$\delta j = -\frac{\delta E}{\delta net_j} = -\frac{\delta E}{\delta o_j} \times \frac{\delta o_j}{\delta net_j} \quad (net_j = \text{input at the j}^{th} \text{ layer})$$

$$E = \frac{1}{2}\sum_{p=1}^{m}(t_p - o_p)^2$$

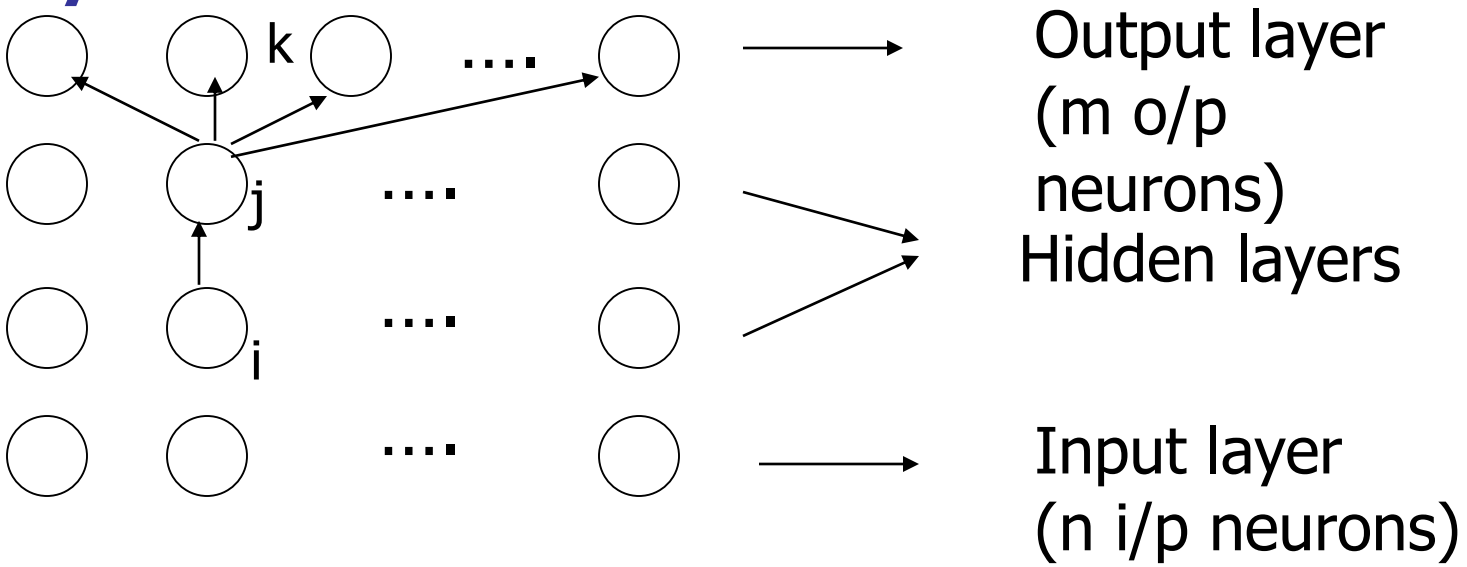
tj is the label of particular output neuron.

if sigmoid neuron.

$$\text{Hence, } \delta j = -(-(t_j - o_j)o_j(1-o_j))$$

$$\Delta w_{ji} = \eta(t_j - o_j)o_j(1-o_j)o_i$$

Backpropagation for hidden layers



You've to make good choice for number of hidden layers as well as the neurons at the hidden layer to choose for the objective function.

δ_k is propagated backwards to find value of δ_j

Backpropagation – for hidden layers

$$\Delta w_{ji} = \eta \delta j o_i$$

$$\delta j = -\frac{\delta E}{\delta net_j} = -\frac{\delta E}{\delta o_j} \times \frac{\delta o_j}{\delta net_j}$$

$$= -\frac{\delta E}{\delta o_j} \times o_j (1 - o_j)$$

$$= -\sum_{k \in \text{next layer}} \left(\frac{\delta E}{\delta net_k} \times \frac{\delta net_k}{\delta o_j}\right) \times o_j (1 - o_j)$$

$$\text{Hence, } \delta_j = -\sum_{k \in \text{next layer}} (-\delta_k \times w_{kj}) \times o_j (1 - o_j)$$

$$= \sum_{k \in \text{next layer}} (w_{kj} \delta_k) o_j (1 - o_j)$$

For sigmoid neuron.

General Backpropagation Rule

- General weight updating rule:

$$\Delta w_{ji} = \eta \delta j o_i$$

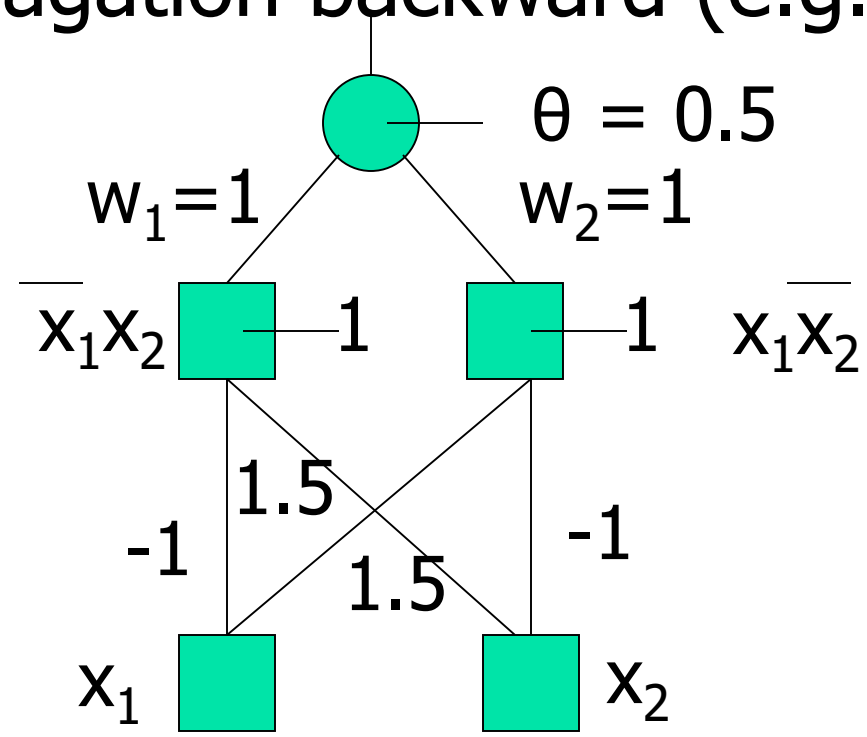
- Where

$$\delta_j = (t_j - o_j) o_j (1 - o_j) \quad \text{for outermost layer}$$

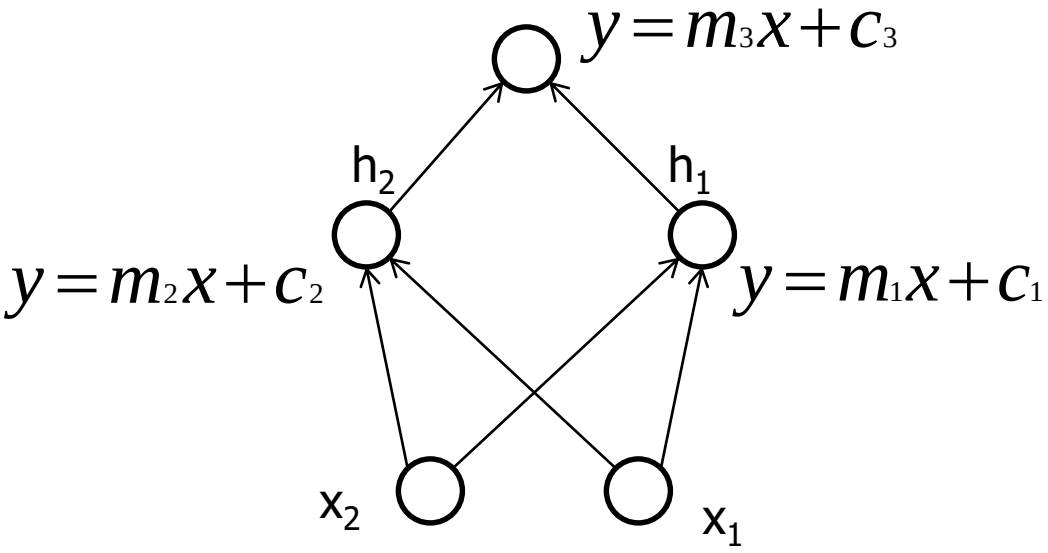
$$= \sum_{k \in \text{next layer}} (w_{kj} \delta_k) o_j (1 - o_j) o_i \quad \text{for hidden layers}$$

How does it work?

- Input propagation forward and error propagation backward (e.g. XOR)



Can Linear Neurons Work?



$$h_1 = m_1(w_1x_1 + w_2x_2) + c_1$$

$$h_1 = m_1(w_1x_1 + w_2x_2) + c_1$$

$$Out = (w_5h_1 + w_6h_2) + c_3$$

$$= k_1x_1 + k_2x_2 + k_3$$

Note: The whole structure shown in earlier slide is reducible to a single neuron with given behavior

$$Out = k_1 x_1 + k_2 x_2 + k_3$$

Claim: A neuron with linear I-O behavior can't compute X-OR.

Proof: Considering all possible cases:

[assuming 0.1 and 0.9 as the lower and upper thresholds]

For (0,0), Zero class:

$$m(w_1.0 + w_2.0 - \theta) + c < 0.1$$
$$\Rightarrow c - m.\theta < 0.1$$

For (0,1), One class:

$$m(w_2.1 + w_1.0 - \theta) + c > 0.9$$
$$\Rightarrow m.w_1 - m.\theta + c > 0.9$$

For (1,0), One class: $m.w_1 - m.\theta + c > 0.9$

For (1,1), Zero class: $m.w_1 - m.\theta + c > 0.9$

These equations are inconsistent. Hence X-OR can't be computed.

Observations:

1. A linear neuron can't compute X-OR.
2. A multilayer FFN with linear neurons is collapsible to a single linear neuron, hence **no a additional power due to hidden layer.**
3. Non-linearity is essential for power.

If neurons are linear then it collapses to one single neuron hence, we redirect it through the sigmoid which is a non-linear function.

An application in Medical Domain

Expert System for Skin Diseases Diagnosis

- Bumpiness and scaliness of skin
- Mostly for symptom gathering and for developing diagnosis skills
- Not replacing doctor’s diagnosis

Architecture of the FF NN

- 96-20-10
- 96 input neurons, 20 hidden layer neurons, 10 output neurons
- Inputs: skin disease symptoms and their parameters
  - *Location, distribution, shape, arrangement, pattern, number of lesions, presence of an active norder, amount of scale, elevation of papuls, color, altered pigmentation, itching, pustules, lymphadenopathy, palmer thickening, results of microscopic examination, presence of herald pathc, result of dermatology test called KOH*

Output

- 10 neurons indicative of the diseases:
  - *psoriasis, pityriasis rubra pilaris, lichen planus, pityriasis rosea, tinea versicolor, dermatophytosis, cutaneous T-cell lymphoma, secondery syphilis, chronic contact dermatitis, soberrheic dermatitis*

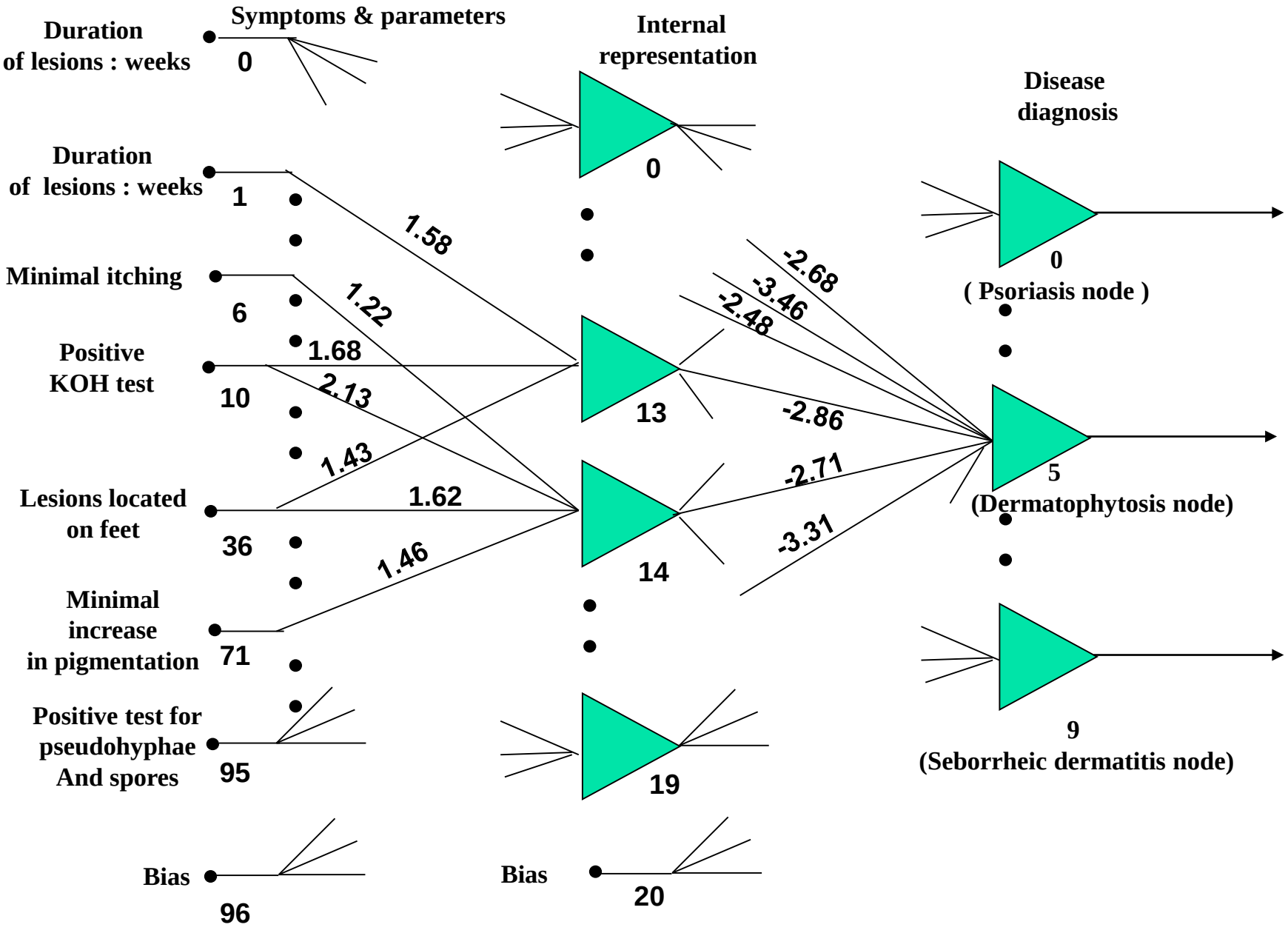


Figure : Explanation of dermatophytosis diagnosis using the DESKNET expert system.

Training data

- Input specs of 10 model diseases from 250 patients
- 0.5 is some specific symptom value is not known
- Trained using standard error backpropagation algorithm

Testing

- Previously unused symptom and disease data of 99 patients
- Result:
- Correct diagnosis achieved for 70% of papulosquamous group skin diseases
- Success rate above 80% for the remaining diseases except for psoriasis
- psoriasis diagnosed correctly only in 30% of the cases These were the challenges for the agent.
- Psoriasis resembles other diseases within the papulosquamous group of diseases, and is somewhat difficult even for specialists to recognise.

Explanation capability

- Rule based systems reveal the explicit path of reasoning through the textual statements
- Connectionist expert systems reach conclusions through complex, non linear and simultaneous interaction of many units
- Analysing the effect of a single input or a single group of inputs would be difficult and would yield incorrect results

Explanation contd.

- The hidden layer re-represents the data
- Outputs of hidden neurons are neither symtoms nor decisions

Discussion

- Symptoms and parameters contributing to the diagnosis found from the n/w

How were these used to check the importance of parameters? mean, s.d. etc.

- Standard deviation, mean and other tests of significance used to arrive at the importance of contributing parameters
- The n/w acts as apprentice to the expert