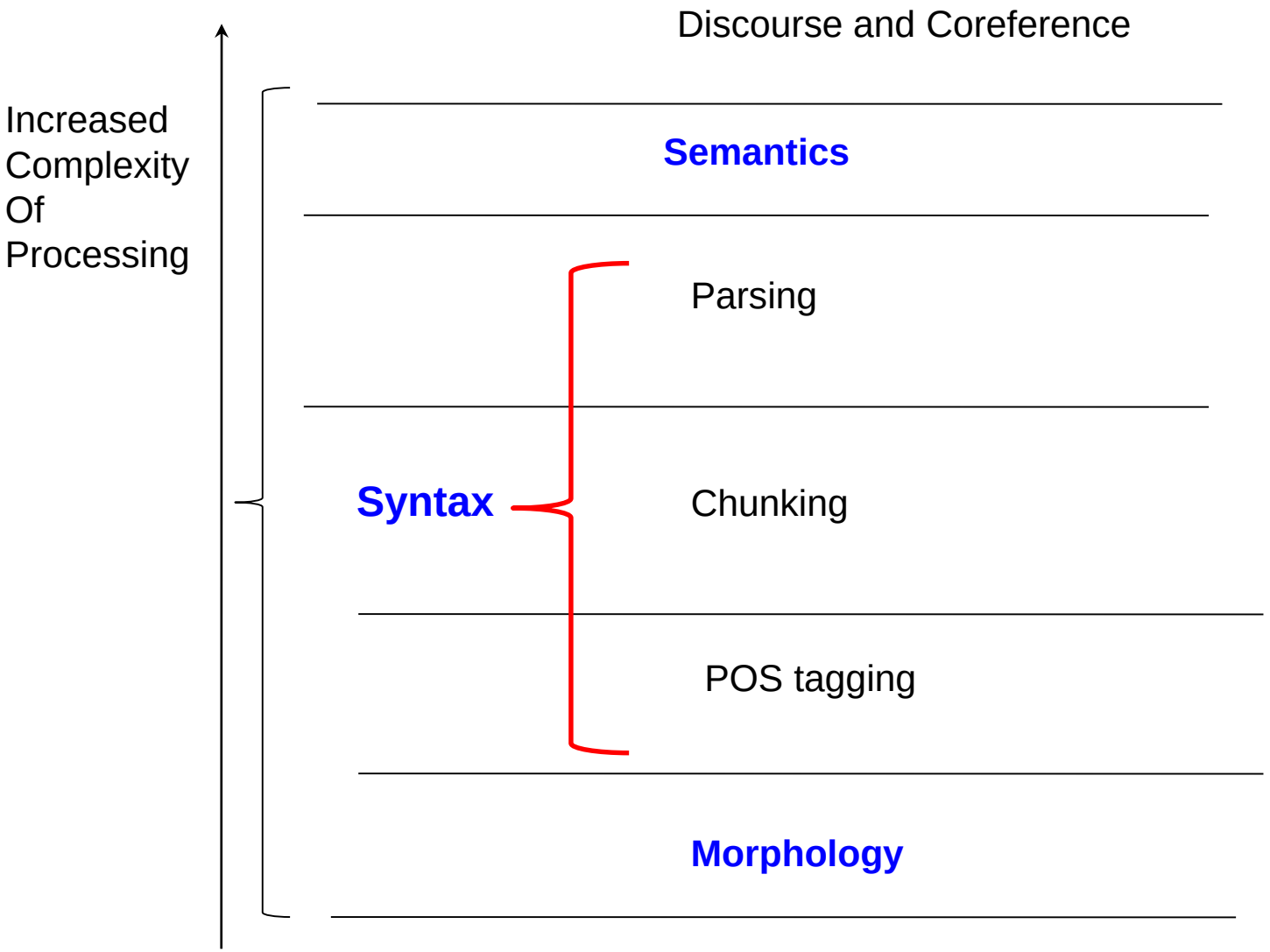
CS626: Speech, Natural Language Processing and the Web

Semantics, WN, FFNN, BP

Pushpak Bhattacharyya
Computer Science and Engineering Department
IIT Bombay

Week 9 of 19th September, 2022

NLP Layer and Linguistics



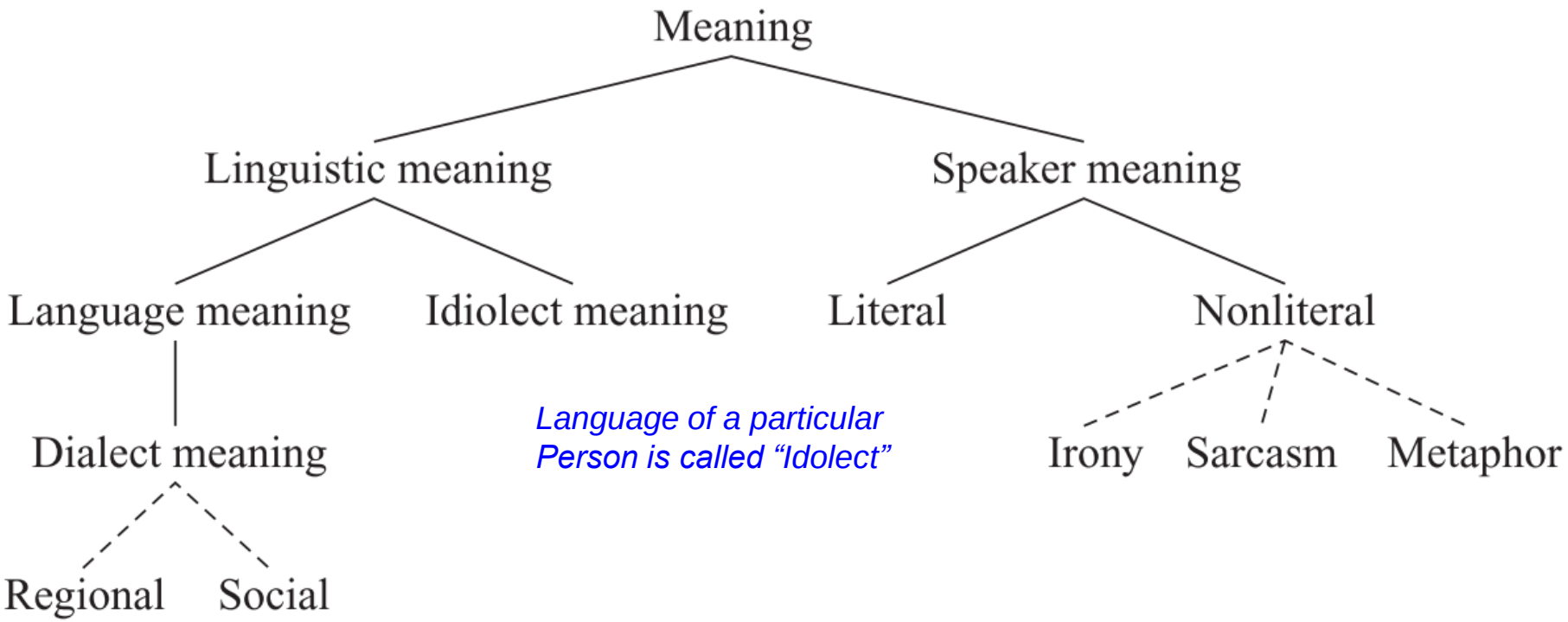
Sound-Structure-Meaning continuum

Sound:
Phonetics, Phonology

Structure:
Morphology, Syntax

Meaning:
Semantic, Pragmatics

The “meaning” ontology



From: Akmajian et al. 2010

Meaning of words

Syntagmatic and Paradigmatic Relations

- Syntagmatic and paradigmatic relations
  - Lexico-semantic relations: synonymy, antonymy, hypernymy, mernymy, troponymy etc. **CAT is-a ANIMAL**
  - Coccurence: **CATS MEW**
- Resources to capture semantics:
  - Wordnet: primarily paradigmatic relations
  - ConceptNet: primarily Syntagmatic Relations

Syntagmatic and Paradigmatic Relations cntd.

- There are interesting studies for English on the syntagmatic and paradigmatic association
- The study finds that when a subject hears a word the words that come on hearing, are 50% syntagmatic and 50% paradigmatic
- Thus on hearing ‘dog’, the words ‘animal’, ‘mammal’, ‘tail’ etc. are pulled as paradigmatic and ‘bark’, ‘friend’, ‘police’ etc. as syntagmatic
- In particular, word vectors capture syntagmatic relations

Selectional Preferences (Indian Tradition) (1/2)

- "Desire" of some words in the sentence ("aakaangksha").
  - I saw the boy with long hair.
  - The verb "saw" desires an object here.

- "Appropriateness" of some other words in the sentence to fulfil that desire ("yogyataa").
  - I saw the boy with long hair.
  - The PP "with long hair" can be appropriately connected only to "boy" and not "saw".

Selectional Preferences (Indian Tradition) (2/2)

- In case, the ambiguity is still present, “proximity” (“sannidhi”) can determine the meaning.
    - E.g. I saw the boy with a telescope.
    - The PP “with a telescope” can be attached to both “boy” and “saw”, so ambiguity still present. It is then attached to “boy” using the proximity check.

Selectional Preference (Recent Linguistic Theory) (1/2)

- There are words which demand arguments, like, verbs, prepositions, adjectives and sometimes nouns. These arguments are typically nouns.

- Arguments must have the property to fulfil the demand. They must satisfy Selectional preferences.

Selectional Preference (Recent Linguistic Theory) (2/2)

- Example
  - Give (verb)
    - » agent – animate
    - » obj – direct
    - » obj – indirect
  - I gave him the book
  - I gave him the book (yesterday in the school)
    → adjunct

Argument frame and Seelctional Preference

- Structure expressing the desire of a word is called the Argument Frame

- Selectional Preference
  - Properties of the “Words that meet the demand”

Verb Argument frame (example)

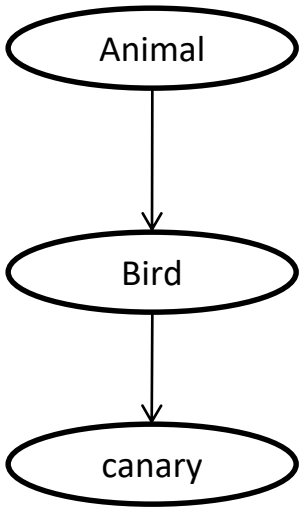
- Verb: give
- Give
  - Agent: <the giver>: animate
  - direct object: <the given>
  - indirect object: <the givee>: personifiable

- *I_agent gave a book_dobj to John_iobj*

Representing Word Meaning: ***Wordnet***

Psycholinguistic Evidence

- Human lexical memory for nouns as a hierarchy.
  - *Can canary sing? - Pretty fast response.*
  - *Can canary fly? - Slower response.*
  - *Does canary have skin? – Slowest response.*



(can move, has skin)

(can fly)

(can sing)

Wordnet- a lexical reference system based on psycholinguistic theories of human lexical memory.

Fundamental Device- Lexical Matrix (with examples)

Word Meanings	**Word Forms**				
	F_1	**F_2**	**F_3**	**…**	**F_n**
M_1	(*depend*) $E_{1,1}$	(*bank*) $E_{1,2}$	(rely) $E_{1,3}$		
M_2		*(bank)* $E_{2,2}$		(*embankment*) $E_{2,…}$	
M_3		*(bank)* $E_{3,2}$	$E_{3,3}$		
…				…	
M_m					$E_{m,n}$

Semantics to be contd.

Assignment-2

- Use word vectors to do word sense disambiguation
- Data will be specified
- You too should look at SemCor, StarSem, Semeval and such sites.

Continuation of Assignment-1

- Use word vectors to do POS tagging; 12 classes
- Train a feedforward neural net
- Compare results with HMM based POS tagging

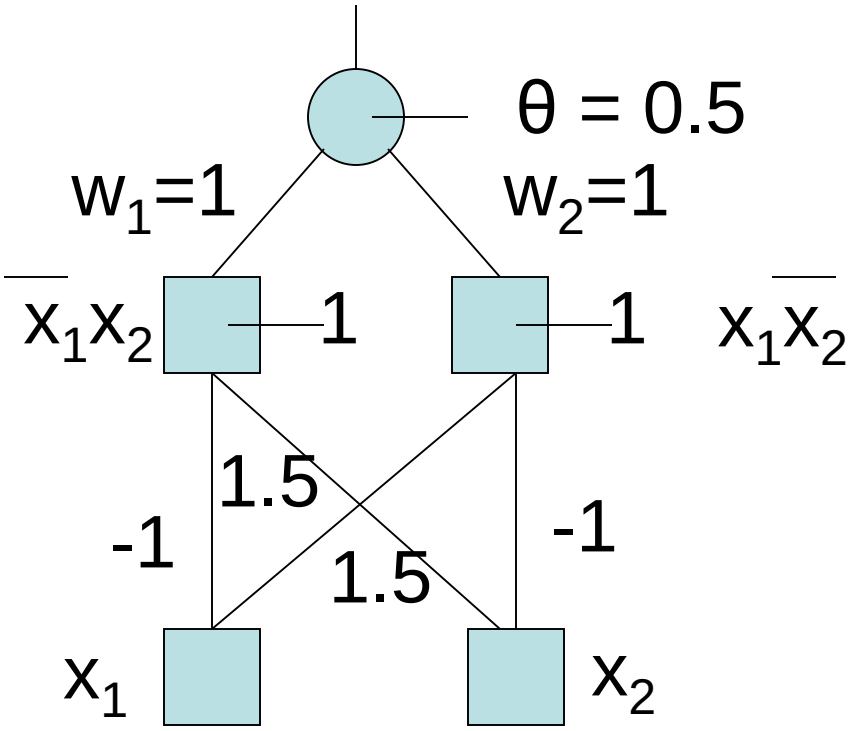
Project Ideas

- (1) Use existing libraries, tools etc. to convert a program spec in Indian Language into a Python code (refer Codex)
- (2) Covert NL sentences into predicate calculus expressions
- (3) Take the data called HateExplain and build a multilingual Hate Speech Detector
- (4) Design Loss Functions that incorporate explainability and do better prediction

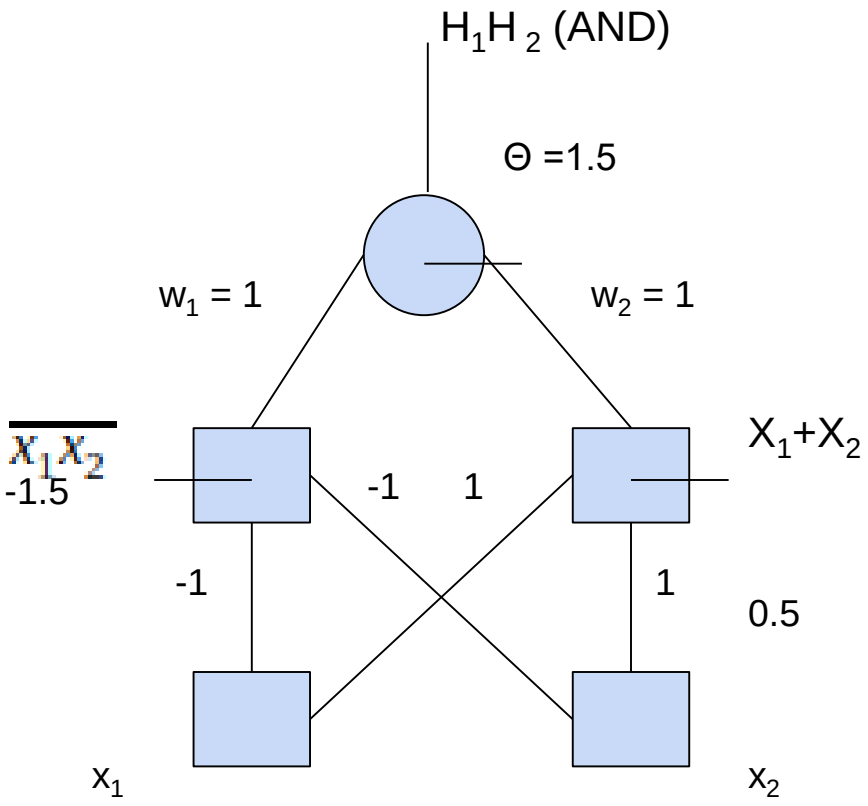
Back to FFNN, BP

Feedforward Network and Backpropagation

Example - XOR

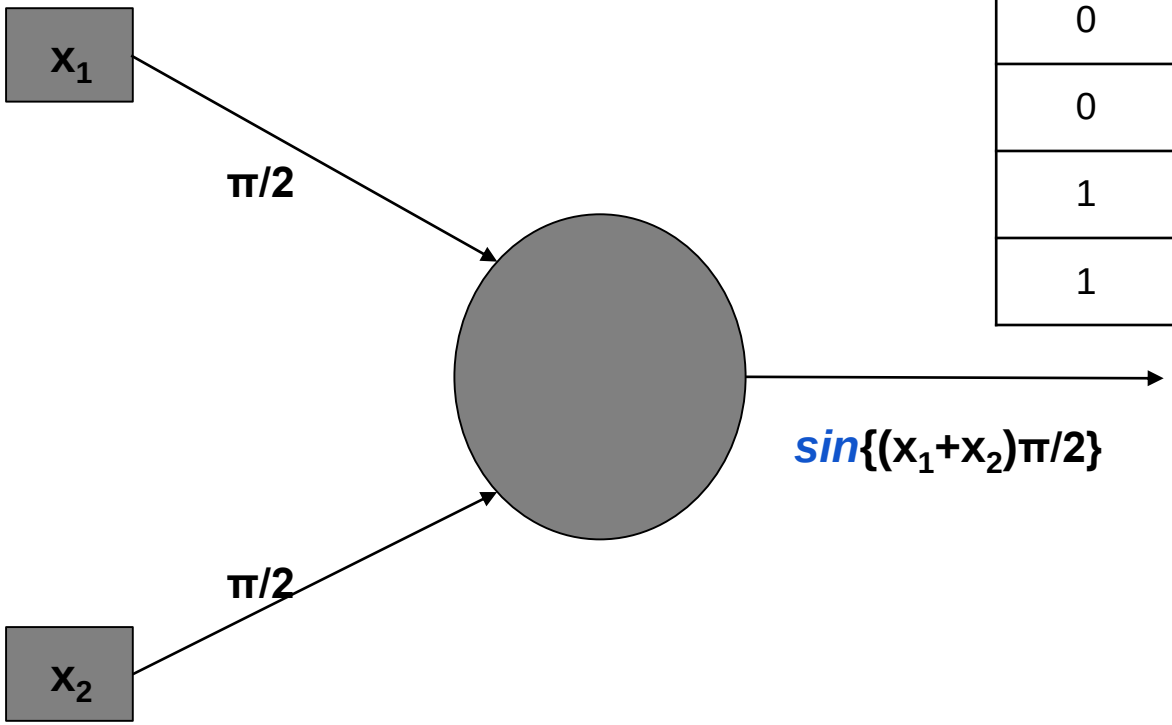


Alternative network for XOR



- XOR: not possible using a single perceptron
- Hidden layer gives more computational capability
- Deep neural network: With multiple hidden layers
- Kolmogorov's theorem of equivalence proves equivalence of multiple layer neural network to a single layer neural network, and each neuron have to correspond to an appropriate functions.

XOR neuron with sin()



x_1	x_2	Output
0	0	0
0	1	1
1	0	1
1	1	0

On the I-O functions for XOR

- No single neuron with monotonic I-O function can compute XOR
- Hidden neurons are a must
- Hidden neurons do not receive input or give output directly
- Sine is a rising and falling function, hence can compute XOR

On the hypothesis space for XOR

- There are at least two different architectures for XOR- through OR or through AND
- And even for a particular architecture there are infinite number of weights that computer XOR
- So the search space for the XOR n/w has infinite members
- OCCAM RAZOR principle- the simplest hypothesis generalizes the best

Exercise: Back-propagation

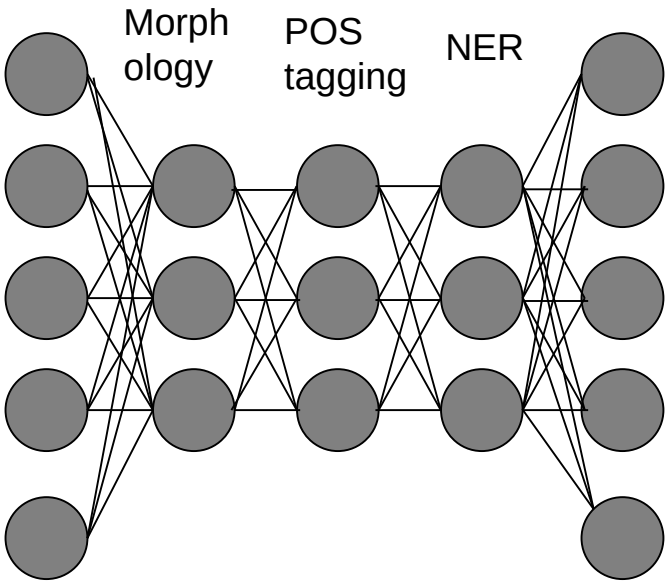
- Implement back-propagation for XOR network

- Observe
  - Check if it converges (error falls below a limit)

  - What is being done at the hidden layer

What a neural network can represent in NLP: Indicative diagram

- Each layer of the neural network possibly represents different NLP stages!!



Batch learning versus Incremental learning

- Batch learning is updating the parameters after ONE PASS over the whole dataset
- Incremental learning updates parameters after seeing each PATTERN
- An epoch is ONE PASS over the entire dataset
  - Take XOR: data set is V_1=(<0,0>, 0), V_2=(<0,1>, 1), V_3=(<1,0>, 1), V_4=(<1,1>, 0)
  - If the weight values are changed after each of Vi, then this is incremental learning
  - If the weight values are changed after one pass over all V_is, then it is batch learning

Can we use PTA for training FFN?

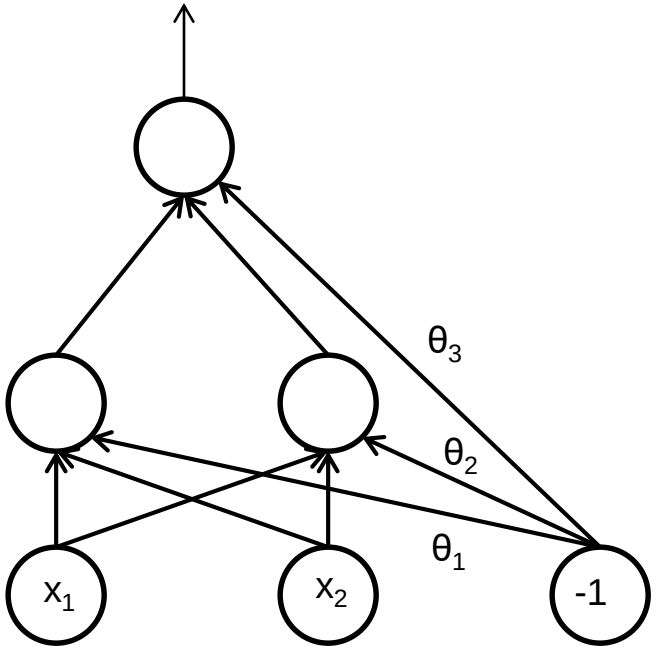
0, 0	0
0, 1	1
1, 0	1
1, 1	0

⇒

-1, 0, 0	0
-1, 0, 1	1
-1, 1, 0	1
-1, 1, 1	0

⇒

1, 0, 0	0
-1, 0, 1	1
-1, 1, 0	1
1, -1, -1	0



No, else the individual neurons are solving XOR, which is impossible.
Also, for the hidden layer neurons we do nothave the i/o behaviour.

Gradient Descent Technique

- Let E be the error at the output layer

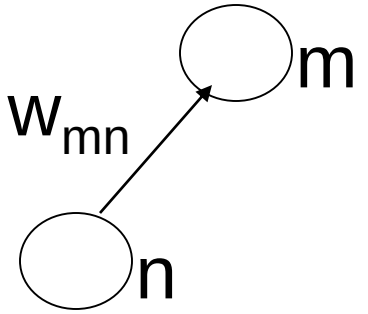
$$E = \frac{1}{2}\sum_{j=1}^{p}\sum_{i=1}^{n}(t_i - o_i)_j^2$$

- 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

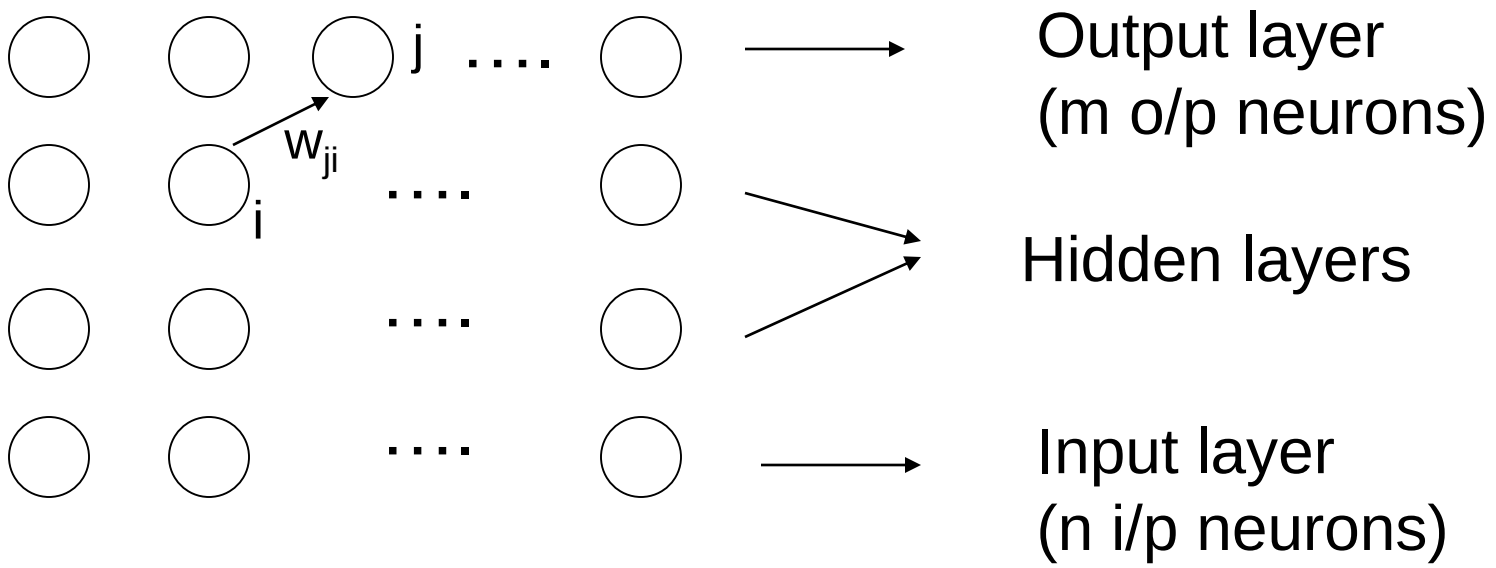
Weights in a FF NN

- 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



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

Backpropagation algorithm



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

Gradient Descent Equations

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

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

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

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

Backpropagation – for outermost layer

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

$$E = \frac{1}{2}\sum_{j=1}^{m}(t_j - o_j)^2$$

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

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

Observations from Δw_{ji}

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

$\Delta w_{ji} \to 0$ if,

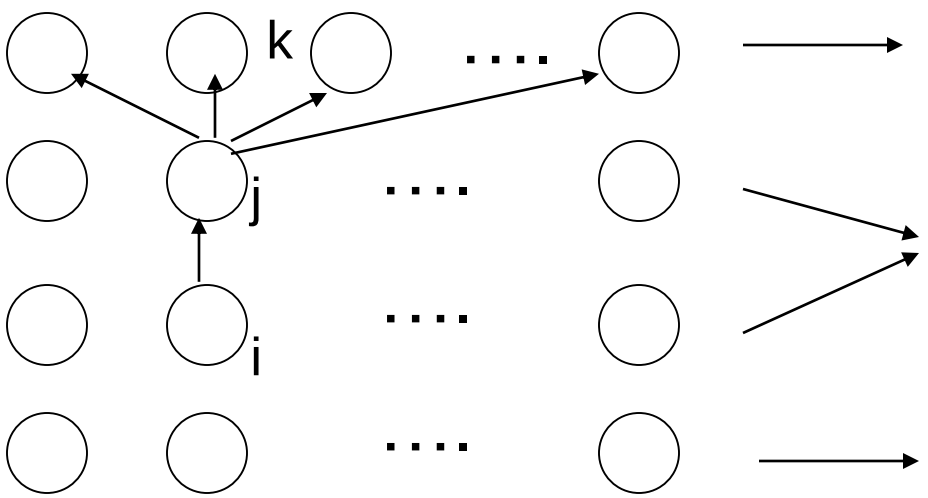
1. $O_j \to t_j$ and/or
2. $O_j \to 1$ and/or
3. $O_j \to 0$ and/or

} Saturation behaviour

4. $O_i \to 0$

} Credit/Blame assignment

Backpropagation for hidden layers



Output layer
(m o/p neurons)

Hidden layers

Input layer
(n i/p neurons)

δ_k is propagated backwards to find value of δ_j

Backpropagation – for hidden layers

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

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

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

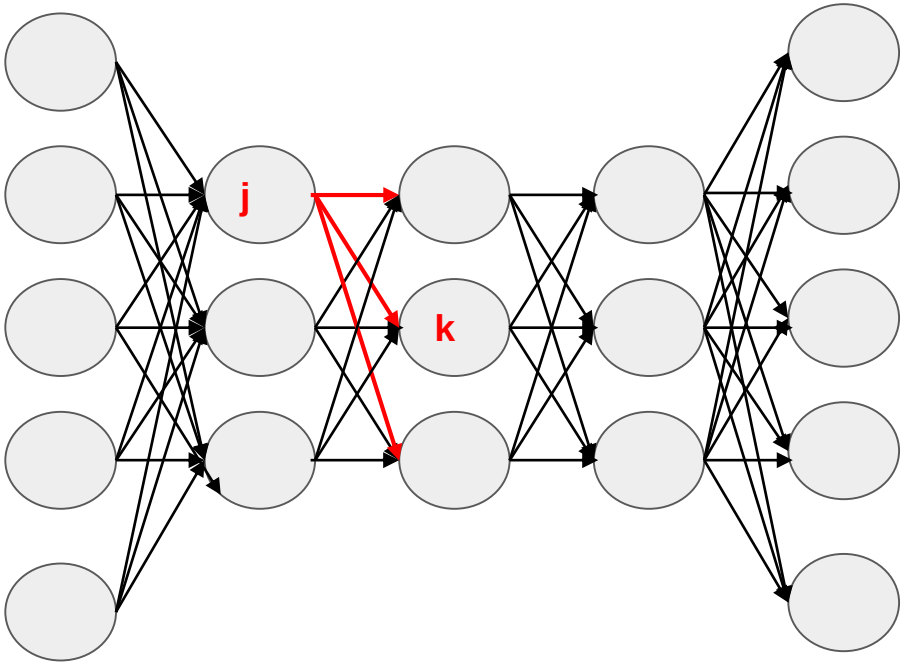
This recursion can give rise to vanishing and exploding Gradient problem

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

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

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

Back-propagation- for hidden layers: Impact on net input on a neuron



- O_j affects the net input coming to all the neurons in next layer

General Backpropagation Rule

- General weight updating rule:

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

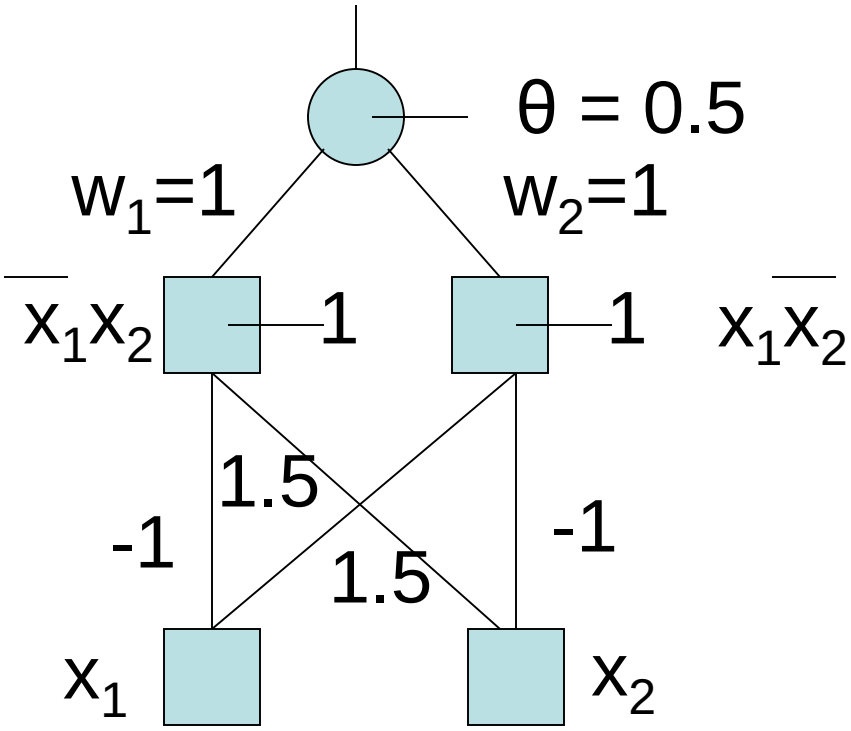
- Where

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

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

How does it work?

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



FFNN, BP discussion to continue…