CS626: Speech, Natural Language Processing and the Web

Algorithmics of Parsing, dependency parsing

Pushpak Bhattacharyya
Computer Science and Engineering Department
IIT Bombay

Week 7 of 5^{th} September, 2022

Algorithmics of Parsing

Problem Statement

INPUT: (a) grammar rules, (b) input sentence

OUPUT: Parse Tree (Constituency/Dependency)

Top Down

- Start with the *S* symbol and draw its children: say, *NP* and *VP*, assuming the input to be a declarative sentence.
- Now the subtrees under *NP*, followed by that under the *VP* are developed.
- For example, *NP* $\rightarrow$ *DT NN* could be applied.
- After this, only POS tags will need to be resolved. *DT* will absorb, say, the word '*the*' in the input and *NN*, '*man*'.
- This will complete constructing the *NP* subtree.
- Similarly, VP subtree also will be constructed.

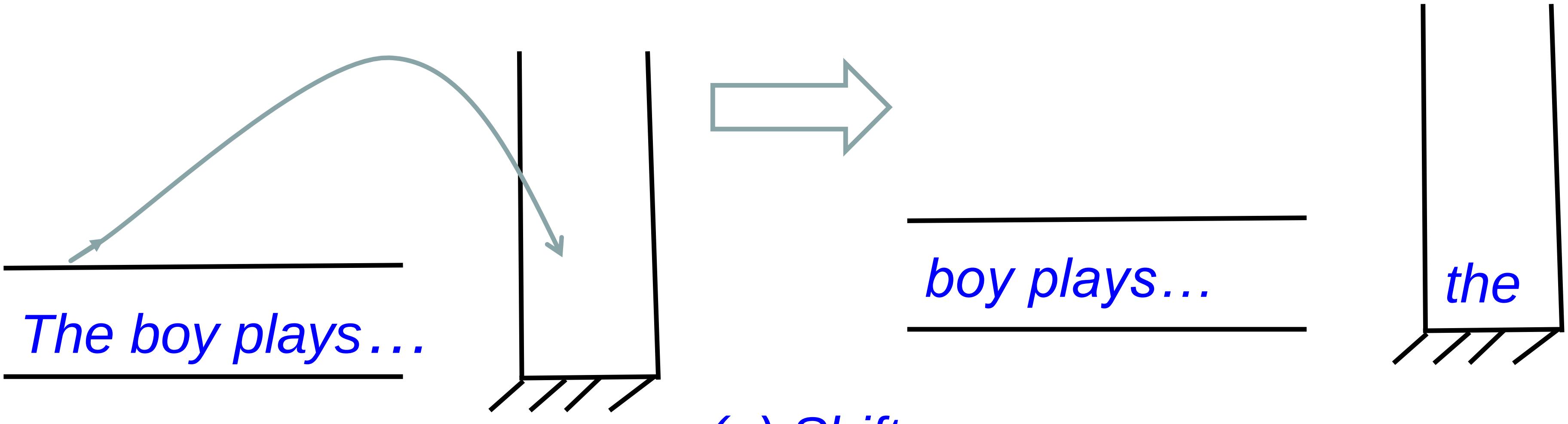
Bottom Up

- The words are resolved to their POS tags.
- Then POS tags are combined by constituency rules, e.g., $NP \rightarrow DT\ NN$. Generated non terminals are then attempted to be combined.
- For example, after generating *JJP, NP* they are combined to form a bigger *NP*, by applying $NP \rightarrow JJP\ NP$.

Main Operations

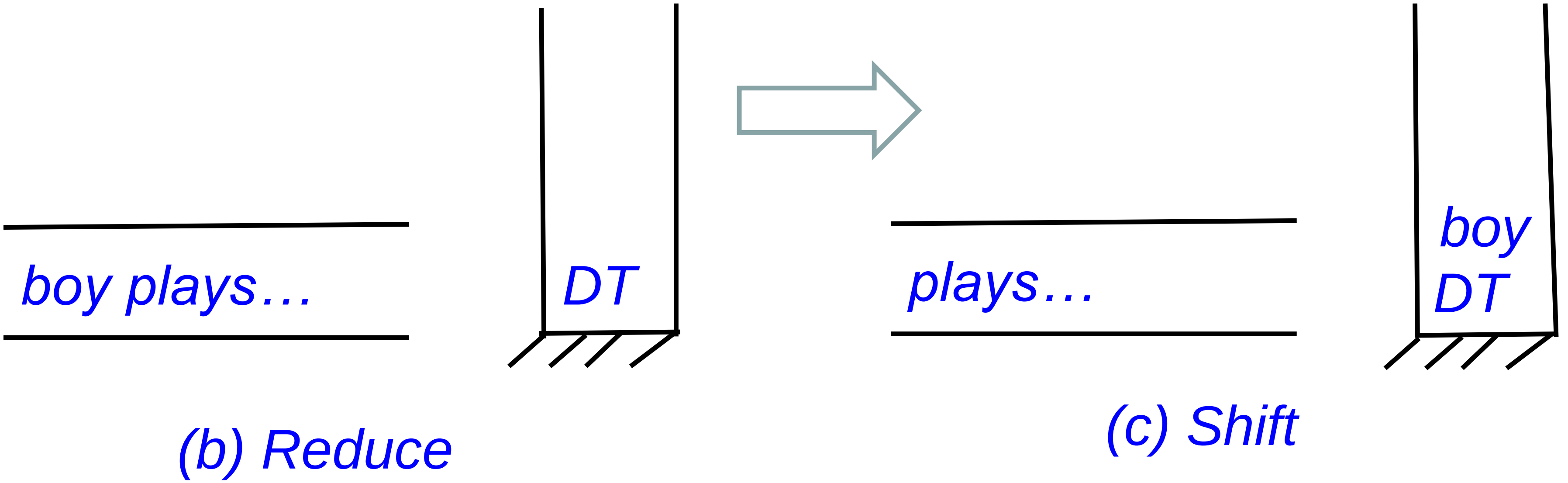
- Doing a left to right scan of the input sentence
- At every word, deciding if the word should (a) create a new constituent or (b) wait until more words get a look-in to create a constituent, and
- On creation of a new constituent, examining if the new constituent can be merged with an adjacent one to form a bigger constituent.

Shift Reduce (1/3)

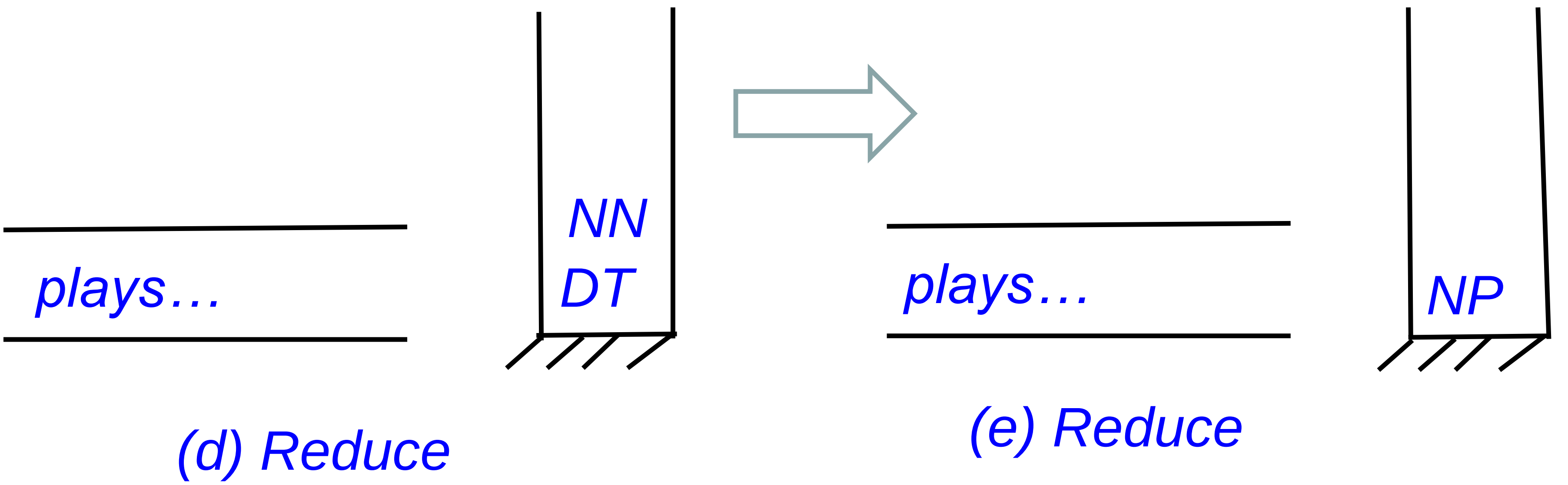


(a) Shift

Shift Reduce (2/3)



Shift Reduce (3/3)



Top-Down Parsing

	State	Backup State	Action
1.	((S) 0)	--	Expand S
2.	((NP VP) 0)	-	Expand NP; have backup
3.	((DT NN VP) 0)	((NN VP) 0)	Match DT; fail; bring backup
4.	((NN VP) 0)	--	Consume '*People*'; pop NN; advance input pointer
5.	((VP) 1)	--	Expand VP
6.	((VB RB) 1)	((VB) 1)	Consume '*laugh*'; pop VB advance input pointer
7.	((RB) 2)	--	Match RB; fail; bring backup; Retract input pointer
8.	((VB) 1)	--	Consume 'laugh'; pop VB
9.	(() 2)	--	Stack empty; input over; Parsing **Succeeds**

A Grammar and an input sentence

Grammar:

(1) $S \rightarrow NP\ VP$

(2) $NP \rightarrow DT\ N \mid NN$

(3) $VP \rightarrow VB\ RB \mid VB$

Sentence is

$_0\ People\ _1\ laugh\ _2$

Bottom-Up Parsing

people *laugh*

0 1 2

NN_{01} NN_{12}

VB_{01} VB_{12}

$NP_{01} \rightarrow NN_{01} \circ$ $VP_{12} \rightarrow VB_{12} \circ$

$VP_{01} \rightarrow VB_{01} \circ$ $S_{02} \rightarrow NP_{01}\ VP_{12} \circ$

$S_{02} \rightarrow NP_{01} \circ VP_{12}$

Commentary on Top-Down Parsing

- Top down parsing- goal driven
- Goal- to reach a state of stack-empty and input-over.
- AKA, *recursive descent parsing*, *predictive parsing* as well as *expectation driven parsing*.
- Names arise from properties:
  - handling of recursive rules, descending from *S* to *NP VP* and their children, and predicting or expecting constituents at different positions in the input

Limitations of Top-Down Parsing (1/2)

- Useless rule expansions: *NP* → *DT NN*, *VP* → *VB RB*
- Bringing backup states on to stack, retracting the input pointer- these are expensive operations.
- Precedence to textually earlier appearance. If *NP* → *NN* appeared before *NP* → *DT NN*, backtracking would have been avoided. Similarly for *VP* → *VB* as against *VP* → *VB RB*

Limitations of Top-Down Parsing (2/2)

- Left recursion causing infinite loop.
  - Consider the rule *JJP* → *JJP JJ*. Once applied, this rule will keep pushing symbols *JJP* and *JJ* onto the stack *ad infinitum*.

- The root of all problems with top-down parsing is that the algorithm is blind to the actual data!

Top-Down Bottom-Up Parsing

0 *people* *1* *laugh* *2*

$S_{02} \rightarrow \circ\ NP_{01}\ VP_{12}$

$NP_{01} \rightarrow \circ\ NN_{01}$

$NP_{01} \rightarrow \circ\ DT_{01}\ NN_{12}$

NN_{01}

$NP_{01} \rightarrow NN_{01}\ \circ$

$S_{02} \rightarrow NP_{01}\ \circ\ VP_{12}$

$VP_{12} \rightarrow \circ\ VB_{12}$

$VP_{12} \rightarrow \circ\ VB_{12}\ RB_{23}$

VB_{12}

$VP_{12} \rightarrow VB_{12}\ \circ$

$S_{02} \rightarrow NP_{01}\ VP_{12}\ \circ$

CYK Parsing

Positions (row-col)	**1**	**2**
0	NN→ NP	S
1		VB→ VP

More involved CYK Parsing: A segment of English

- S → NP VP
- NP → DT NN
- NP → NNS
- NP → NP PP
- PP → P NP
- VP → VP PP
- VP → VBD NP

- DT → the
- NN → gunman
- NN → building
- VBD → sprayed
- NNS → bullets

CYK Parsing: Start with (0,1)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT						
1	--------						
2	--------	----------					
3	--------	----------	---------				
4	---------	----------	---------	----------			
5	---------	----------	---------	----------	----------		
6	---------	----------	---------	----------	----------	----------	

CYK: Keep filling diagonals

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	1	2	3	4	5	6	7
0	DT						
1	-------	NN					
2	-------	---------					
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: Try getting higher level structures

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To **From**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT	NP					
1	-------	NN					
2	-------	---------					
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: Diagonal continues

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To **From**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT	NP					
1	-------	NN					
2	-------	---------	VBD				
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7.**

To From	1	2	3	4	5	6	7
0	DT	NP	---------				
1	-------	NN	---------				
2	-------	---------	VBD				
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7.**

To From	1	2	3	4	5	6	7
0	DT	NP	--------				
1	-------	NN	--------				
2	-------	---------	VBD				
3	-------	---------	--------	DT			
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------			
3	-------	---------	--------	DT			
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: starts filling the 5th column

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7.**

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------			
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: S found, but NO termination!

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S		
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S		
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	

CYK: Control moves to last column

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	**DT**	**NP**	--------	---------	**S**	---------	
1	-------	**NN**	--------	---------	---------	---------	
2	-------	---------	**VBD**	---------	**VP**	---------	
3	-------	---------	--------	**DT**	**NP**	---------	**NP**
4	--------	---------	--------	---------	**NN**	---------	---------
5	--------	---------	--------	---------	---------	**P**	**PP**
6	--------	---------	--------	---------	---------	---------	**NP** **NNS**

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	VP
3	-------	---------	--------	DT	NP	---------	NP
4	--------	---------	--------	---------	NN	---------	---------
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK: filling the last column

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7.**

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	---------
2	-------	---------	VBD	---------	VP	---------	VP
3	-------	---------	--------	DT	NP	---------	NP
4	--------	---------	--------	---------	NN	---------	---------
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK: terminates with S in (0,7)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7** *.*

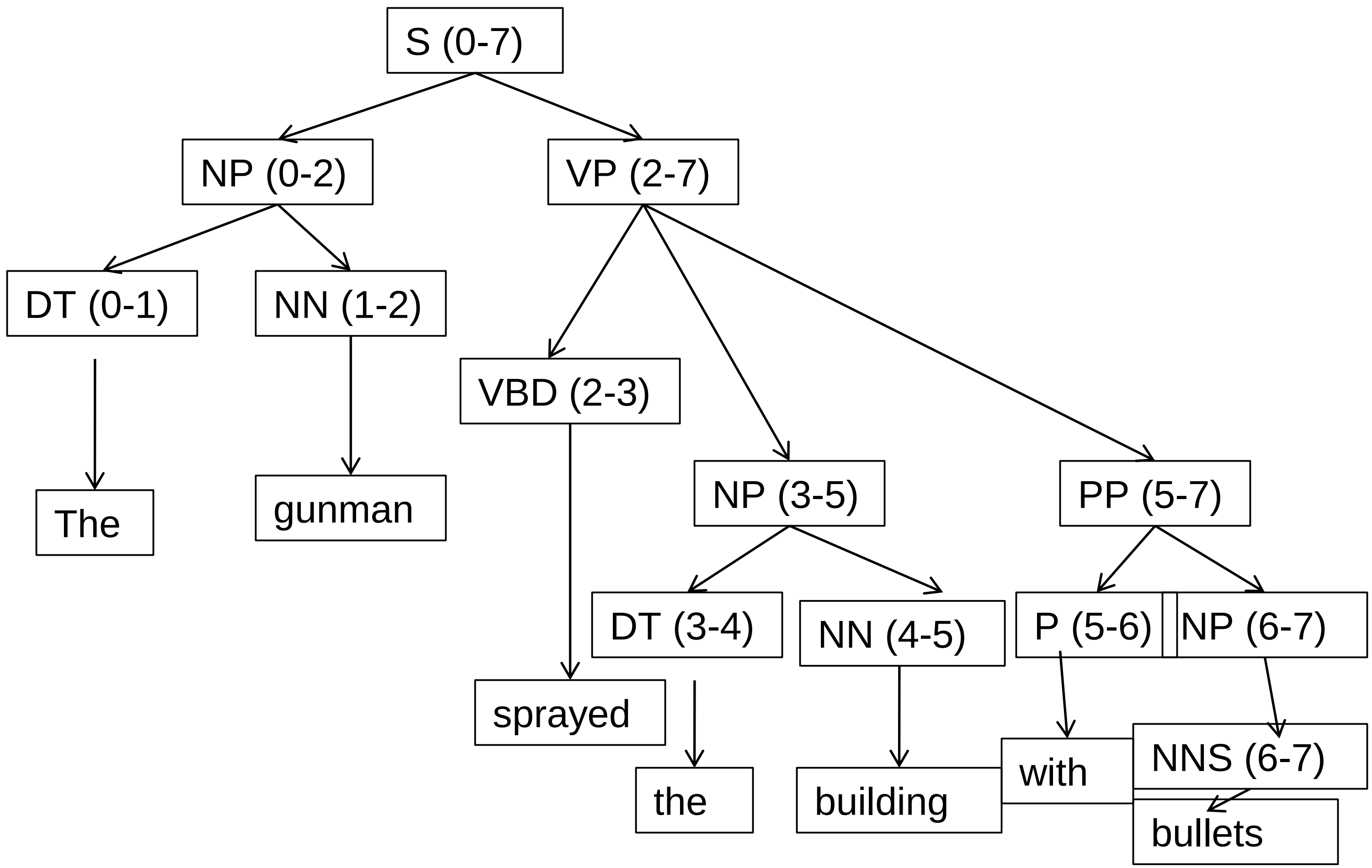
To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	S
1	-------	NN	--------	---------	---------	---------	---------
2	-------	---------	VBD	---------	VP	---------	VP
3	-------	---------	--------	DT	NP	---------	NP
4	--------	---------	--------	---------	NN	---------	---------
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK: Extracting the Parse Tree

- The parse tree is obtained by keeping back pointers.

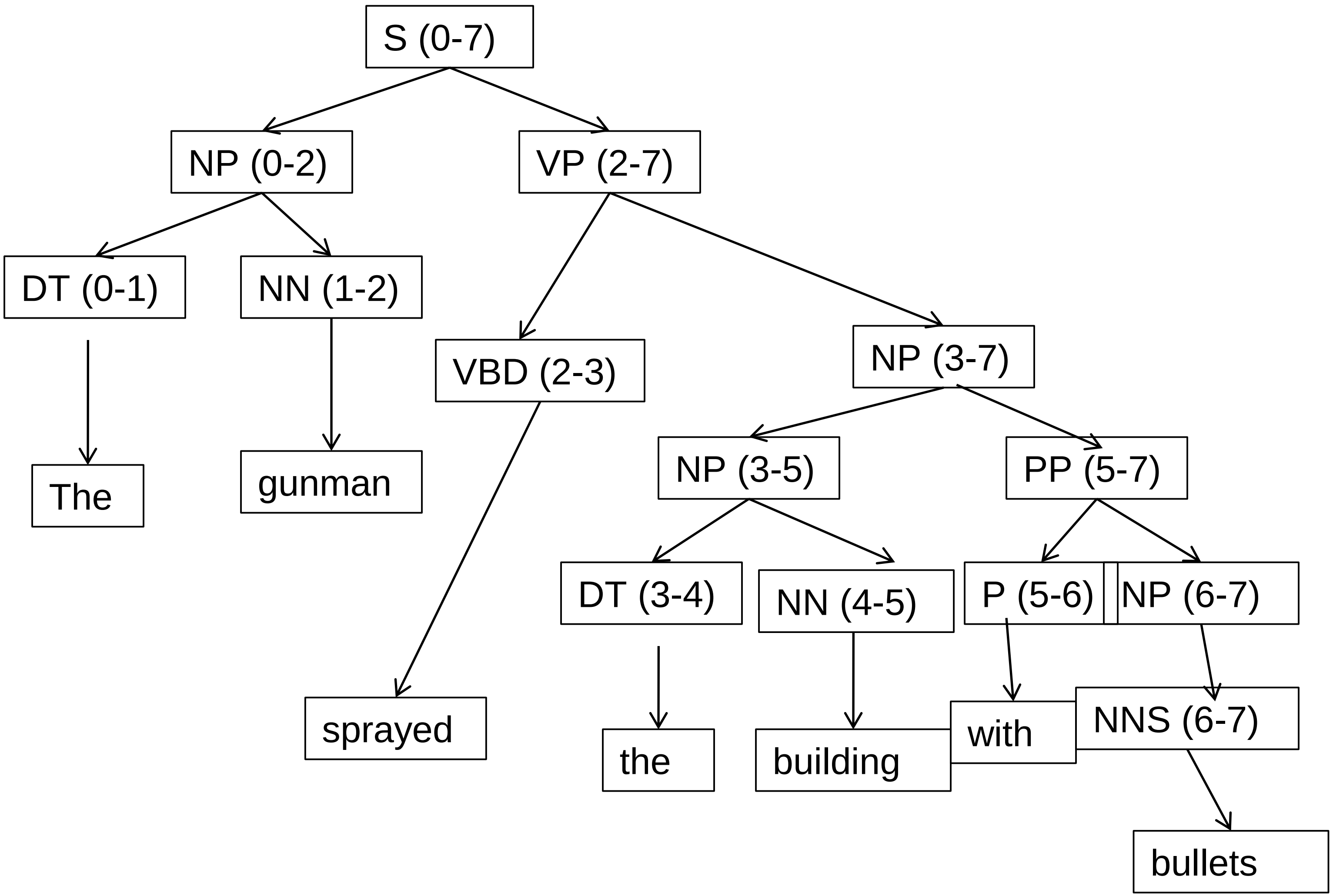
Parse Tree #1

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*



Parse Tree #2

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

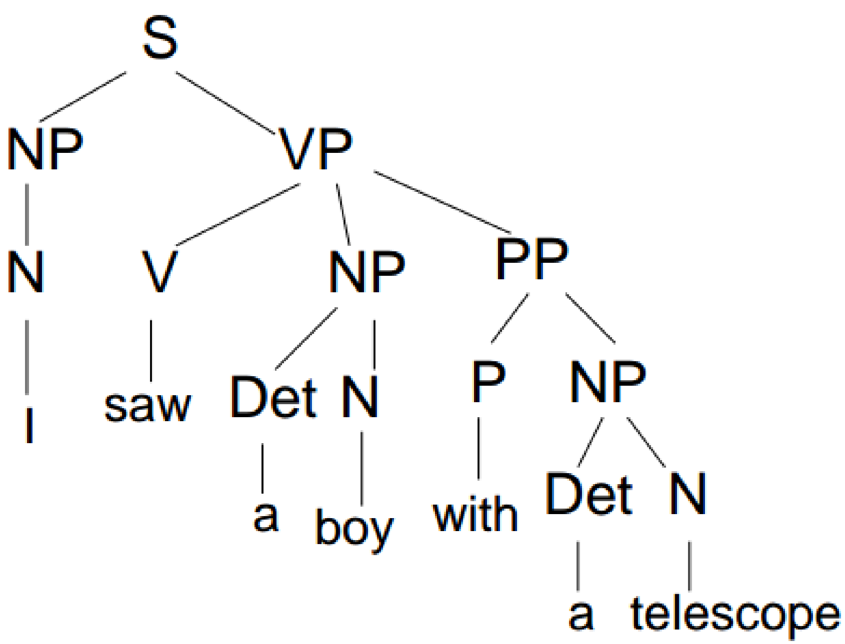


Notion of Domination

- A sentence is dominated by the symbol S through domination of segments by phrases

- Analogy
    - The capital of a country dominates the whole country.
    - The capital of a state dominates the whole state.
    - The district headquarter dominates the district.

Domination: Example



- I saw a boy with a telescope
  - Meaning: I used the telescope to see the boy

- Dominations
  - NP dominates "a telescope"
  - VP dominates "saw a boy with a telescope
  - S dominates the whole sentence
- Domination is composed of many sub-domination.

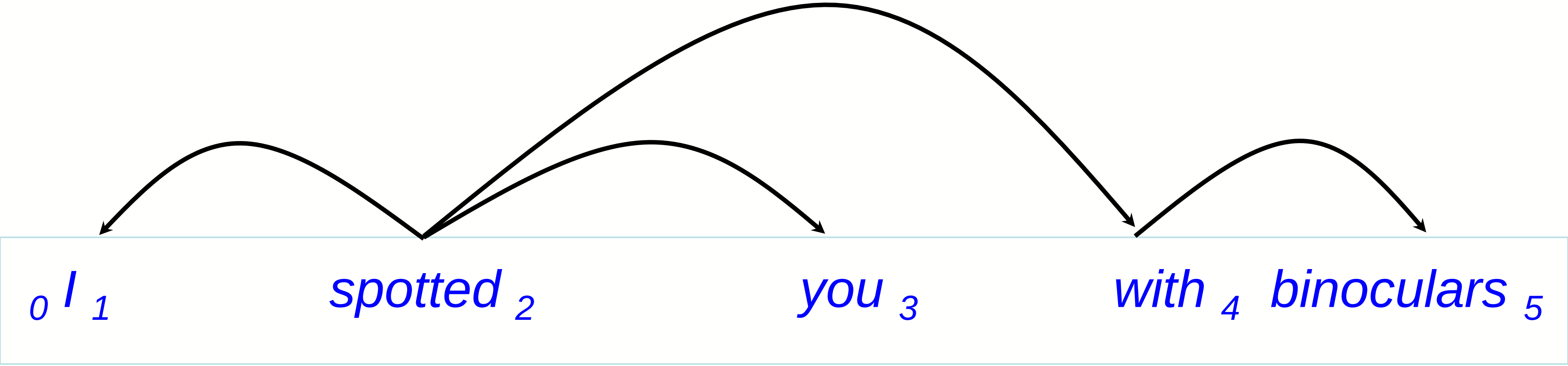
Dependency Parsing

“I spotted you with binoculars”.

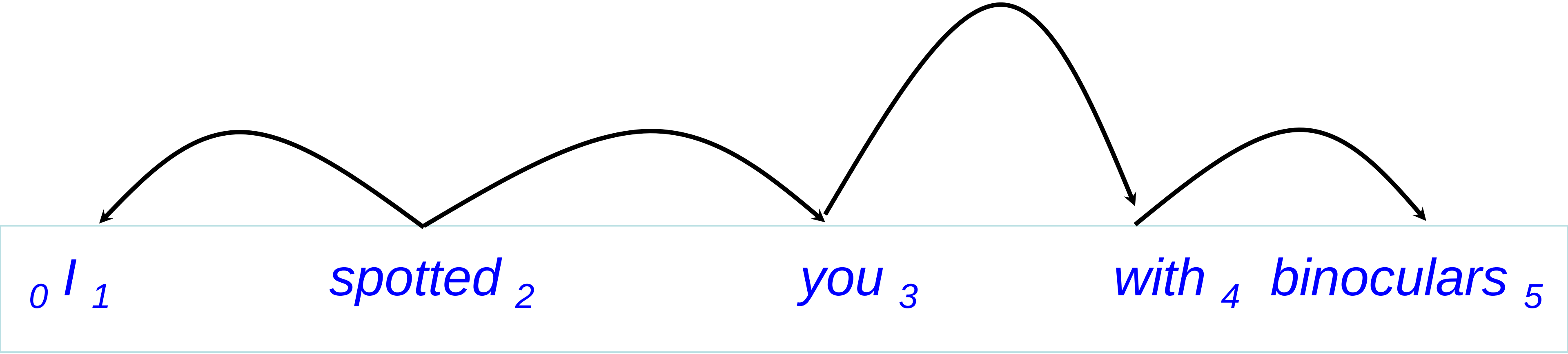
$_0$ *I* $_1$ *spotted* $_2$ *you* $_3$ *with* $_4$ *binoculars* $_5$

- Has two meanings
- I have the binoculars OR
- You have the binoculars

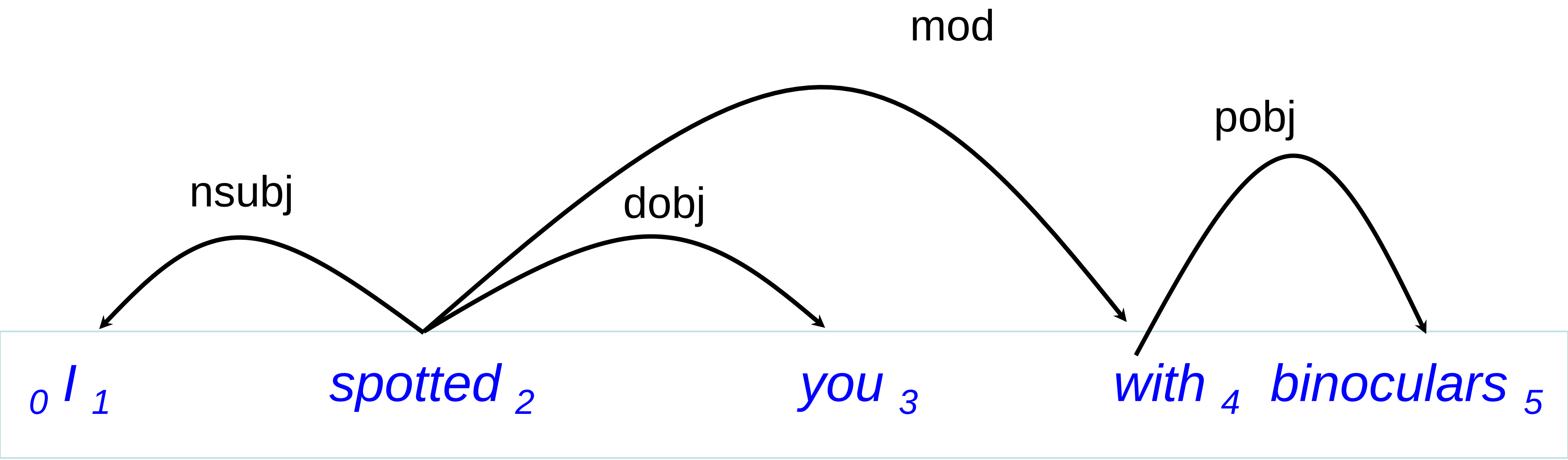
Unlabeled Dependency Tree-1



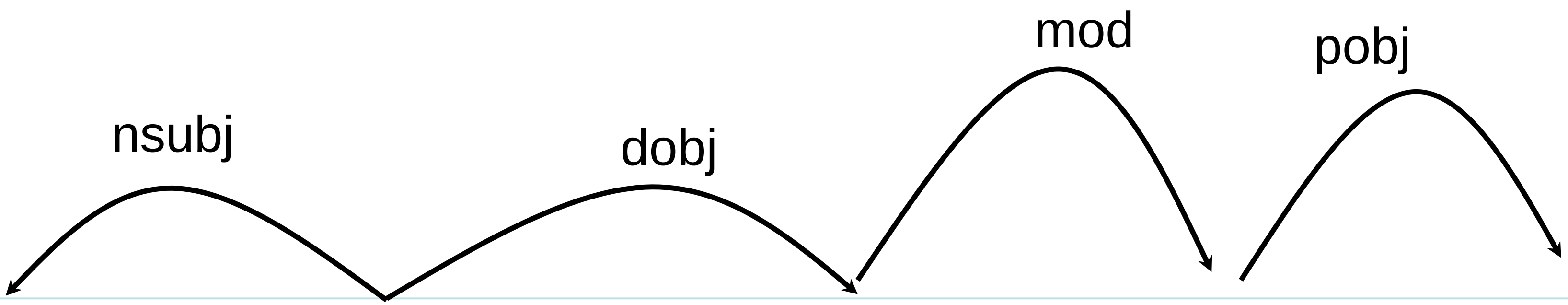
Unlabeled Dependency Tree-2



Labeled Dependency Tree-1



Labeled Dependency Tree-2



$_0$ *I* $_1$ *spotted* $_2$ *you* $_3$ *with* $_4$ *binoculars* $_5$

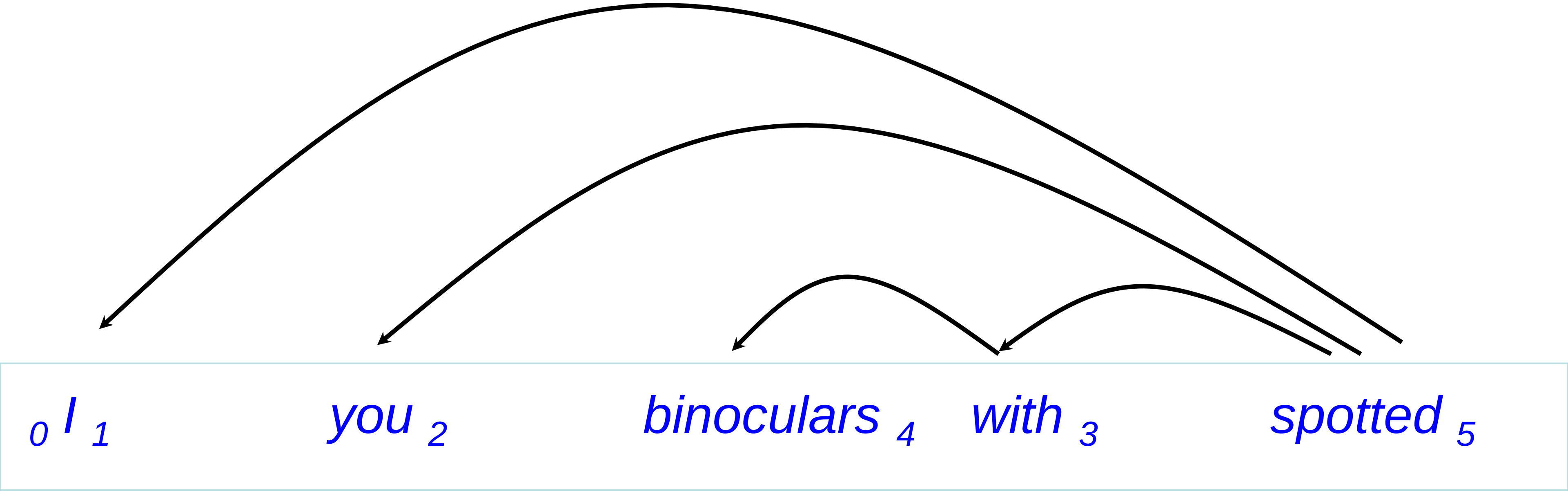
For SOV Syntax

"I you binoculars with spotted"

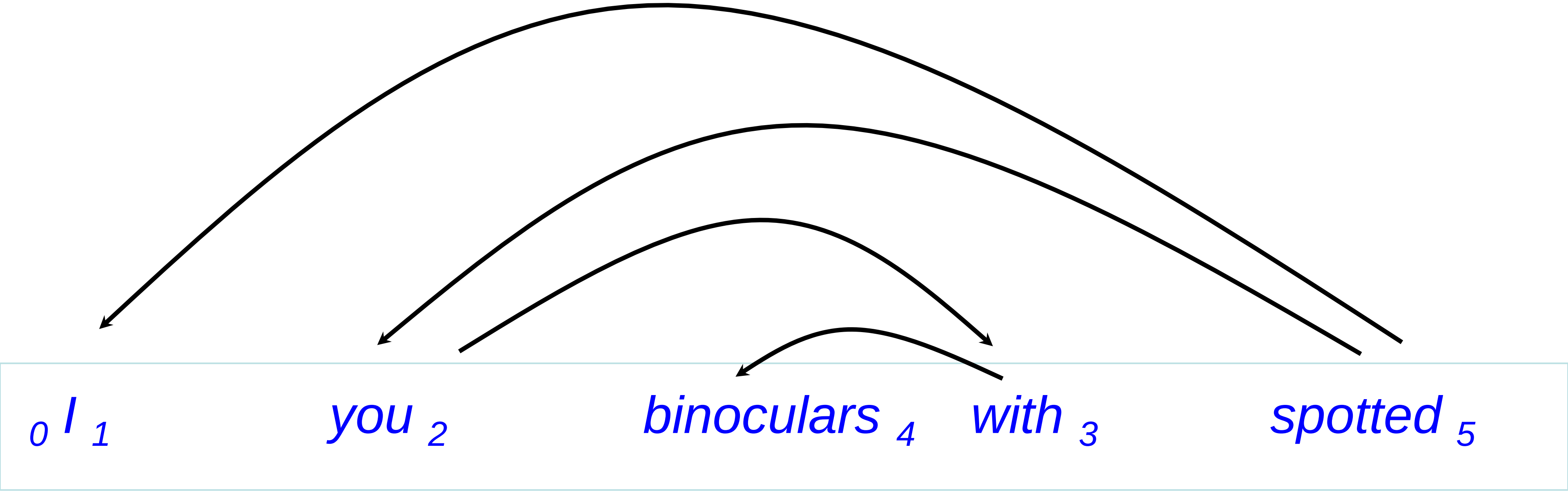
$_0$ *I* $_1$ *spotted* $_2$ *you* $_3$ *with* $_4$ *binoculars* $_5$

- Has two meanings
- I have the binoculars OR
- You have the binoculars

Unlabeled Dependency Tree-1



Unlabeled Dependency Tree-2



Dependency Parsing Example

- Transition based parsing
- Shift and Reduce

Q8: Justification: Parse-1

1. [root]	[I spotted you with binoculars]	shift	no-relation-added
2. [root I]	[spotted you with binoculars]	shift	no-relation-added
3. [root I spotted]	[you with binoculars]	left-arc	I←spotted
4. [root spotted]	[you with binoculars]	shift	no-relation-added
5. [root spotted you]	[with binoculars]	right-arc	spotted→you
6. [root spotted]	[with binoculars]	shift	no-relation-added
7. [root spotted with]	[binoculars]	shift	no-relation added
8. [root spotted with binoculars]	[]	right-arc	with→binoculars
9. [root spotted with]	[]	right-arc	spotted→with
10. [root spotted]	[]	right-arc	root→spotted
11. [root]	[]	parsing ends	

Q8: Justification: Parse-2

1. [root]	[I spotted you with binoculars]	shift	no-relation-added
2. [root I]	[spotted you with binoculars]	shift	no-relation-added
3. [root I spotted]	[you with binoculars]	left-arc	I $\leftarrow$ spotted
4. [root spotted]	[you with binoculars]	shift	no-relation-added
5. [root spotted you]	[with binoculars]	shift	no-relation-added
6. [root spotted you with]	[binoculars]	shift	no-relation-added
7. [root spotted you with binoculars]	[]	right-arc	with $\rightarrow$ binoculars
8. [root spotted you with]	[]	right-arc	you $\rightarrow$ with
9. [root spotted you]	[]	right-arc	spotted $\rightarrow$ you
10. [root spotted]	[]	right-arc	root $\rightarrow$ spotted
11. [root]	[]	parsing ends	

Need Classification

Decision making

- Constituency Parsing
  - Shift
  - Reduce

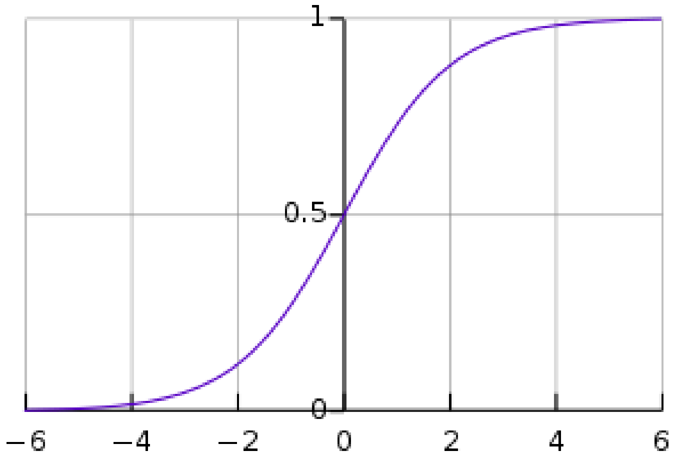
- Dependency Parsing
  - Shift
  - Right Arc
  - Left Arc

2-class: Sigmoid or Logit function

$$y = \frac{1}{1+e^{-x}}$$

$$\frac{dy}{dx} = y(1-y)$$

Sigmoid function



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

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

$$\begin{aligned}
\frac{df(x)}{dx} &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) \\
&= \frac{e^{-x}}{(1+e^{-x})^{-2}} \\
&= \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right) \\
&= f(x).(1 - f(x))
\end{aligned}$$

Decision making under sigmoid

- Output of sigmod is between 0-1

- Look upon this value as probability of Class-1 (C_1)

- *1-sigmoid(x)* is the probability of Class-2 (C_2)
- Decide C_1, if $P(C_1) > P(C_2)$, else C_2

multiclass: SOFTMAX

- 2-class → multi-class (C classes)
- Sigmoid → softmax
- i^{th} input, c^{th} class (small c), k varies over classes
- In softmax, decide for that class which has the highest probability

What is softmax

- Turns a vector of K real values into a vector of K real values that sum to 1
- Input values can be positive, negative, zero, or greater than one
- But softmax transforms them into values between 0 and 1
- so that they can be interpreted as probabilities.

Mathematical form

$$\sigma(\bar{Z})_i = \frac{e^{Z_i}}{\sum_{j=1}^{K} e^{Z_j}}$$

- σ is the **softmax** function
- Z is the input vector of size K
- The RHS gives the i^{th} component of the output vector
- Input to softmax and output of softmax are of the same dimension

Example

$$\bar{Z} = <1, 2, 3>$$

$$Z_1 = 1,\ Z_2 = 2,\ Z_3 = 3$$

$$e^1 = 2.72,\ e^2 = 7.39,\ e^3 = 20.09$$

$$\sigma(\bar{Z}) = <\frac{2.72}{2.72 + 7.39 + 20.09}, \frac{7.39}{2.72 + 7.39 + 20.09}, \frac{20.09}{2.72 + 7.39 + 20.09}>$$

$$= <.09,\ 0.24,\ 0.67>$$

Softmax and Cross Entropy

- Intimate connection between softmax and cross entropy
- Softmax gives a vector of probabilities
- Winner-take-all strategy will give a classification decision

Winner-take-all with softmax

- Consider the softmax vector obtained from the example where the softmax vector is <0.09, 0.24, 0.65>
- These values correspond to 3 classes
  - For example, - *positive (+)*, *negative (-)* and *neutral* (0) sentiments, given an input sentence like
  - *(a) I like the story line of the movie (+). (b) However the acting is weak (-). (c) The protagonist is a sports coach (0)*

Sentence vs. Sentiment

(a) *I like the story line of the movie (+).*
(b) *However the acting is weak (-).*
(c) *The protagonist is a sports coach (0)*

Sentence vs. Sentiment	**Positive**	**Negative**	**Neutral**
Sent (a)	1 *(P_{max} from softmax)*	0	0
Sentence (b)	0	1 *(P_{max} from softmax)*	0
Sentence (C)	0	0`	1 *(P_{max} from softmax)*

Training data

- *(a) I like the story line of the movie (+).*
- *(b) However the acting is weak (-).*
- *(c) The protagonist is a sports coach (0)*

Input	Output
(a)	<1,0,0>
(b)	<0,1,0>
(c)	<0,0,1>

Finding the error

- Difference between target (T) and obtained (Y)
- Difference is called **LOSS**
- Options:
  - Total Sum Square Loss (TSS)
  - Cross Entropy *(measures difference between two probability distributions)*
- Softmax goes with cross entropy

Cross Entropy Function

$$H(P,Q) = -\sum_{x} P(x)\log_2 Q(x)$$

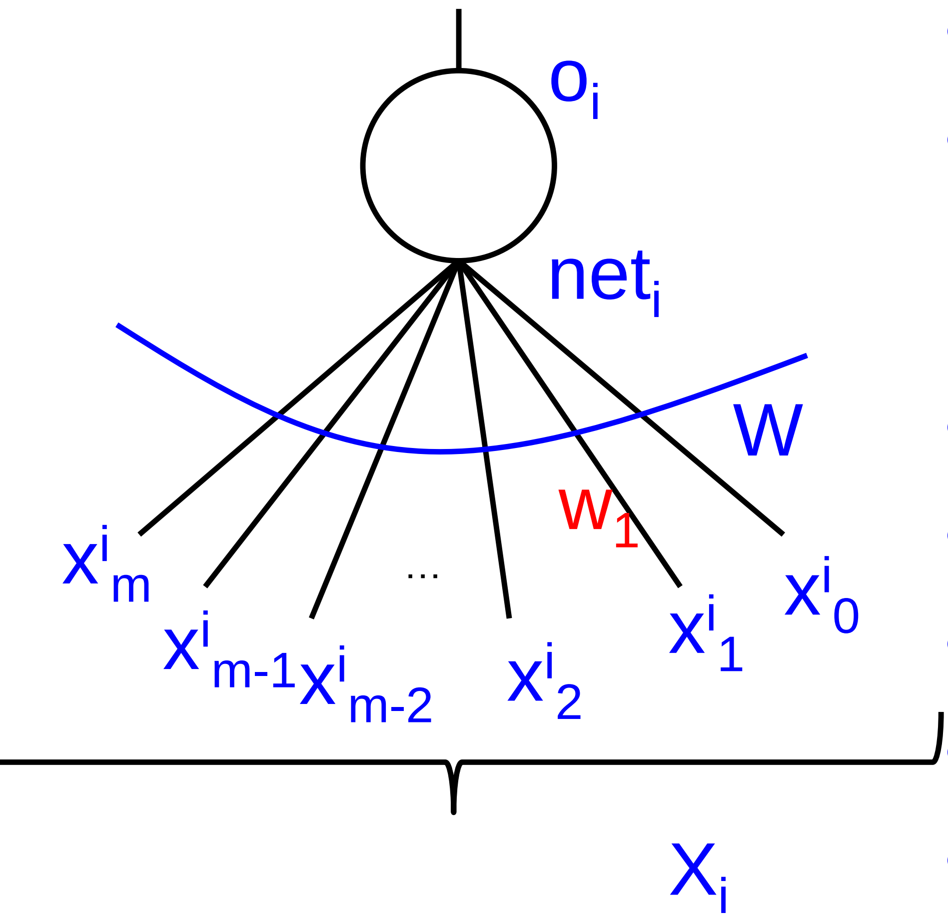
P is target distribution
Q is observed distribution

How to minimize loss

- Gradient descent approach
- Backpropagation Algorithm
- Involves derivative of the input out function for each neuron
- FFNN with BP is one of the most important TECHNIQUEs

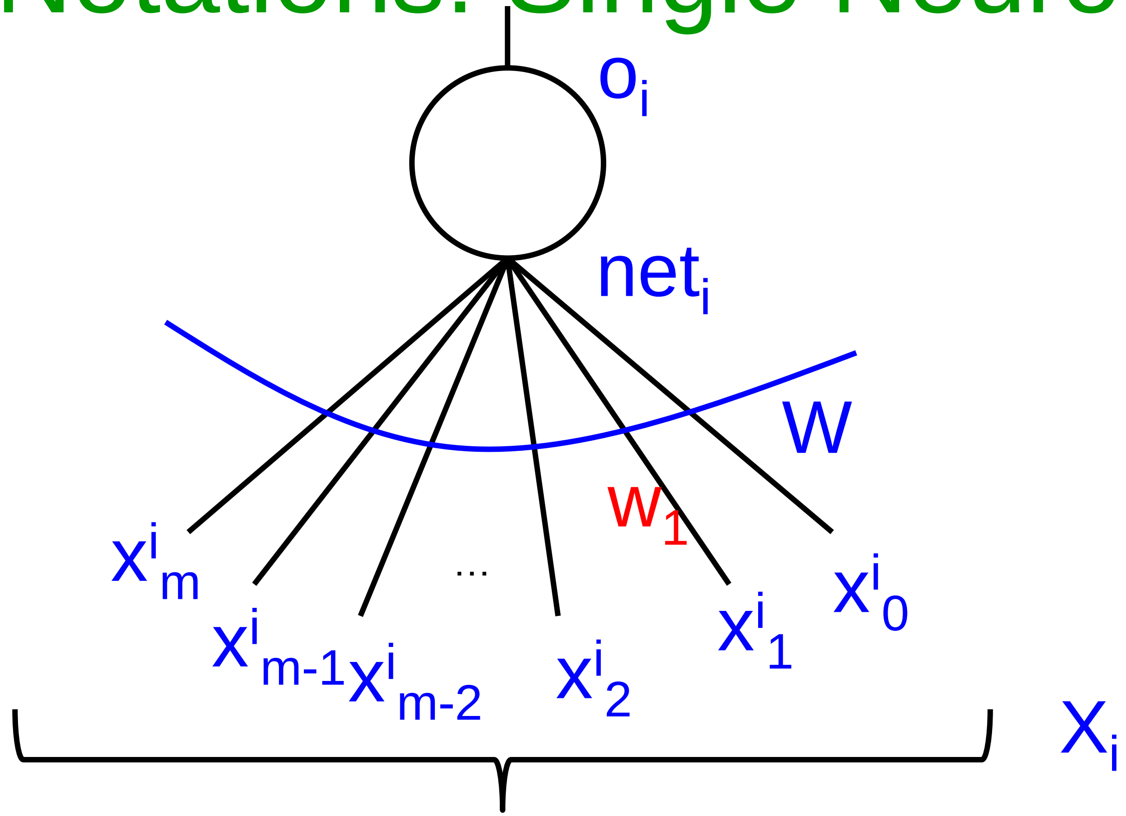
Sigmoid and Softmax Neurons

Fix Notations: Single Neuron (1/2)



- Capital letter for vectors
- Small letter for scalars (therefore for vector components
- X_i: ith input vector
- o_i: output (scalar)
- W: weight vector
- net_i: $W.X_i$
- There are *n* input-output observations

Fix Notations: Single Neuron (2/2)



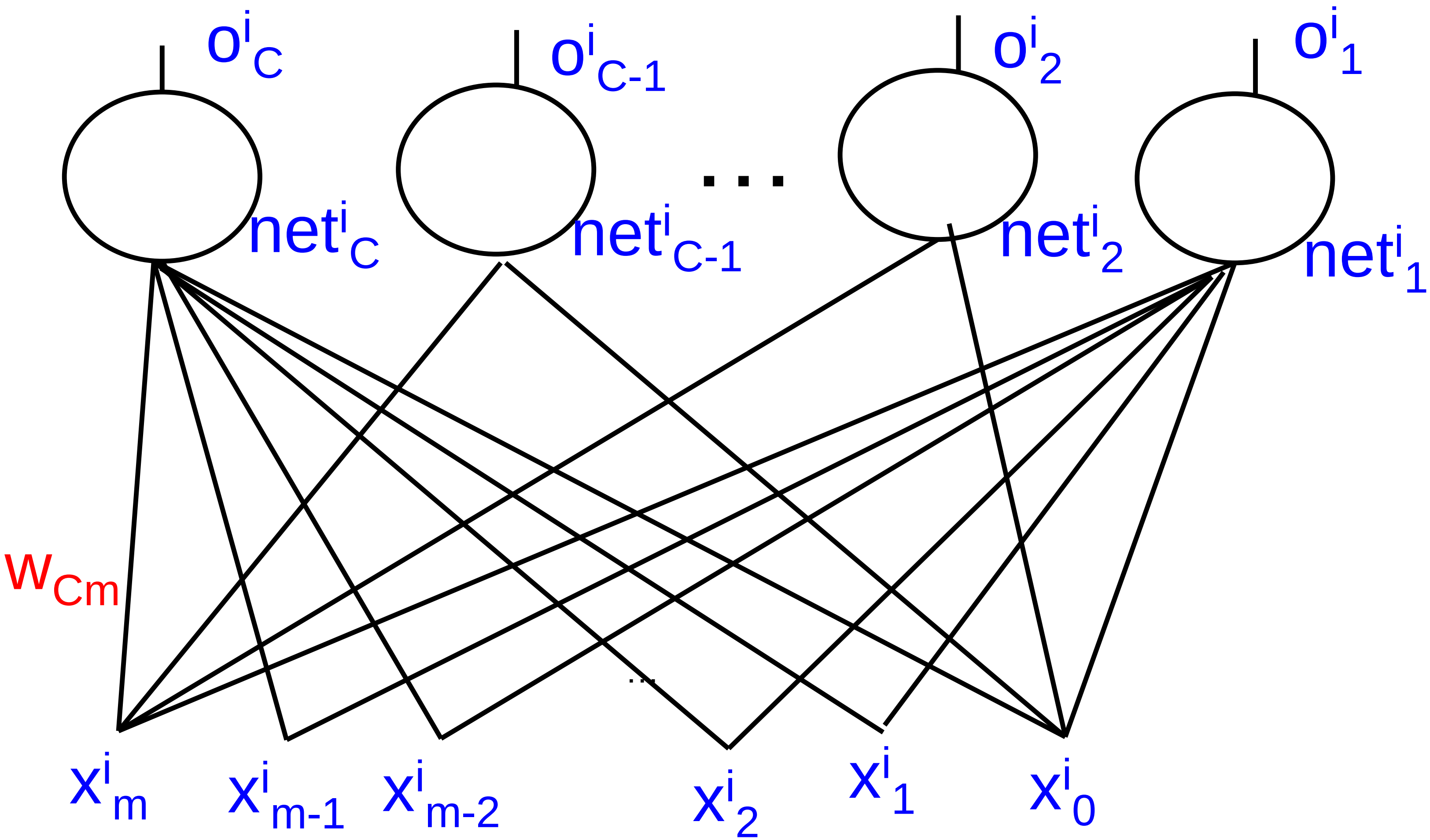
W and each X_i has *m* components

W:$<w_m, w_{m-1}, \ldots, w_2, w_0>$

Xi:$<x^i_m, x^i_{m-1}, \ldots, x^i_2, x^i_0>$

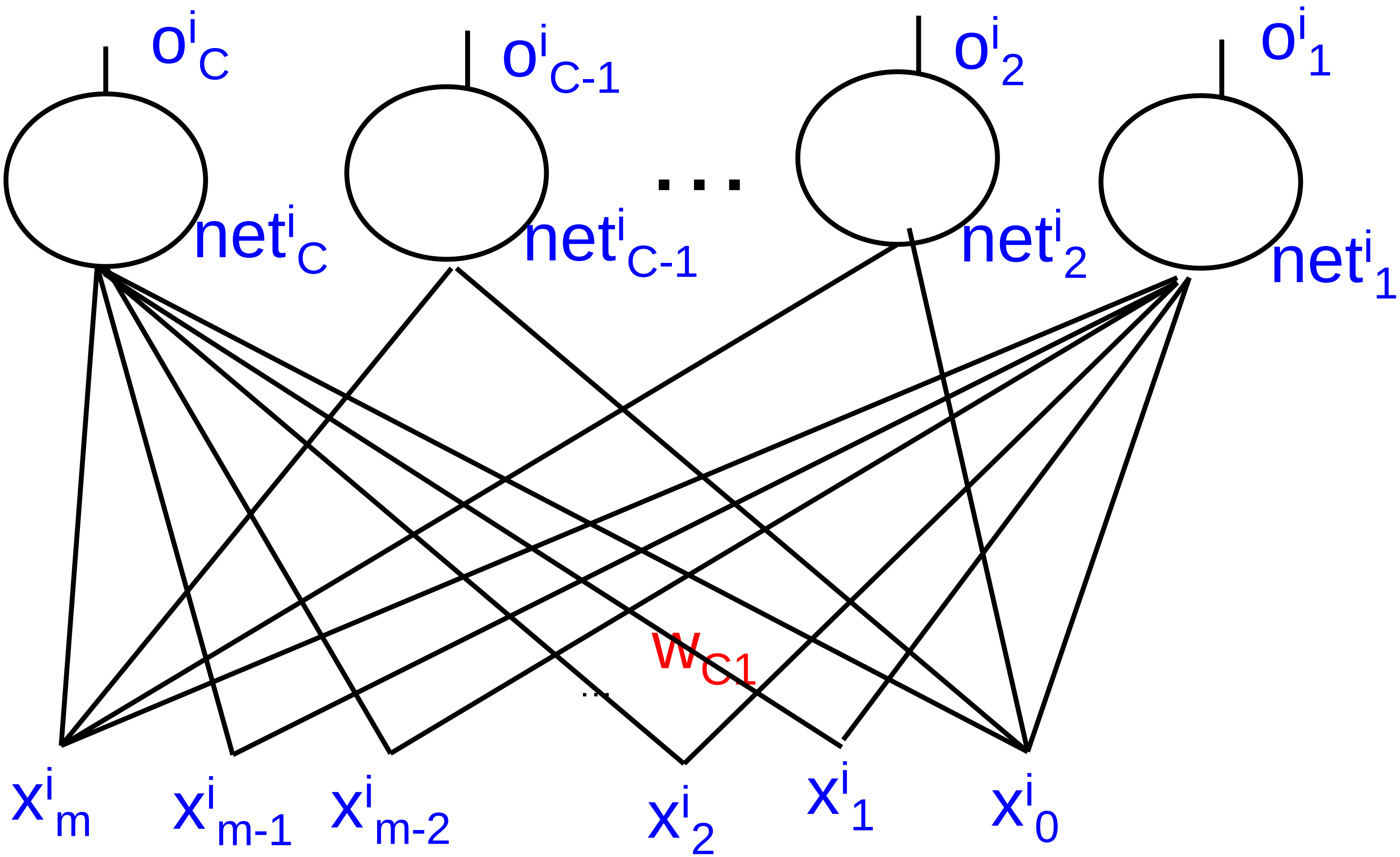
Upper suffix *i* indicates i^{th} input

Fixing Notations: Multiple neurons in o/p layer



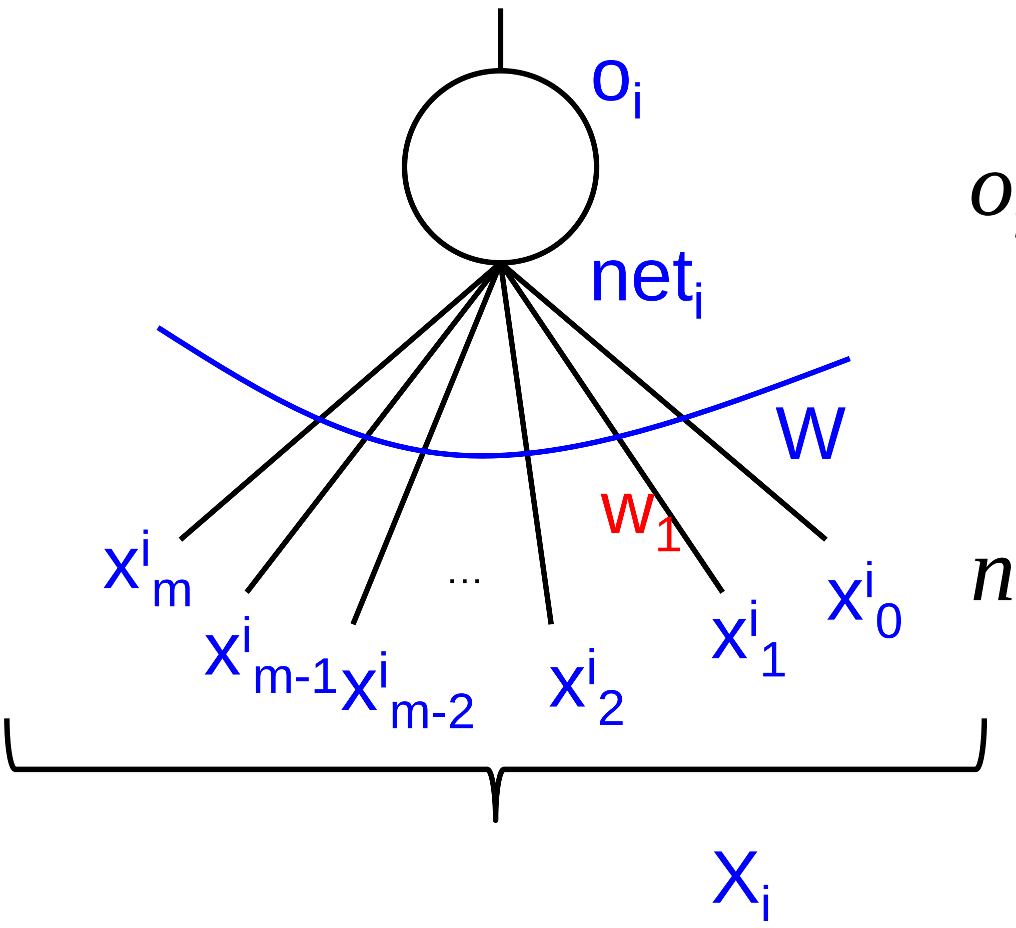
Now, O_i and NET_i are vectors for i^{th} input
W_k is the weight vector for k^{th} output neuron, k=1..C

Fixing Notations



Target Vector, T_i: $<t^i_C\ t^i_{C-1} \ldots t^i_2\ t^i_1>$, $i \rightarrow$ for i^{th} input. Only one of these C componets is 1, rest are 0

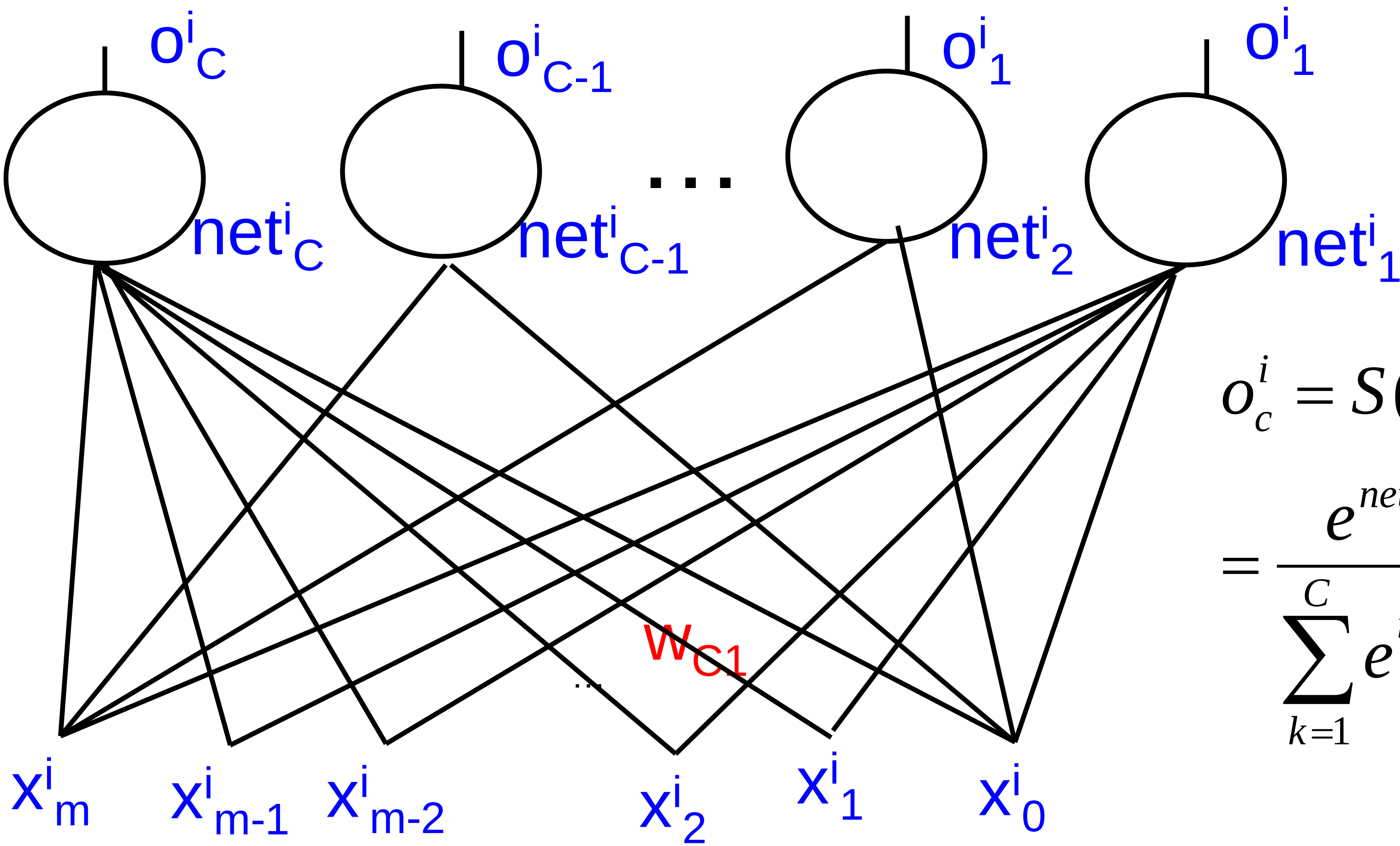
Sigmoid neuron



$$o_i = \frac{1}{1+e^{-net_i}}$$

$$net_i = W.X_i = \sum_{j=0}^{m} w_j x_j^i$$

Softmax Neuron



$$o_c^i = S(\overline{NET}_i)_c$$

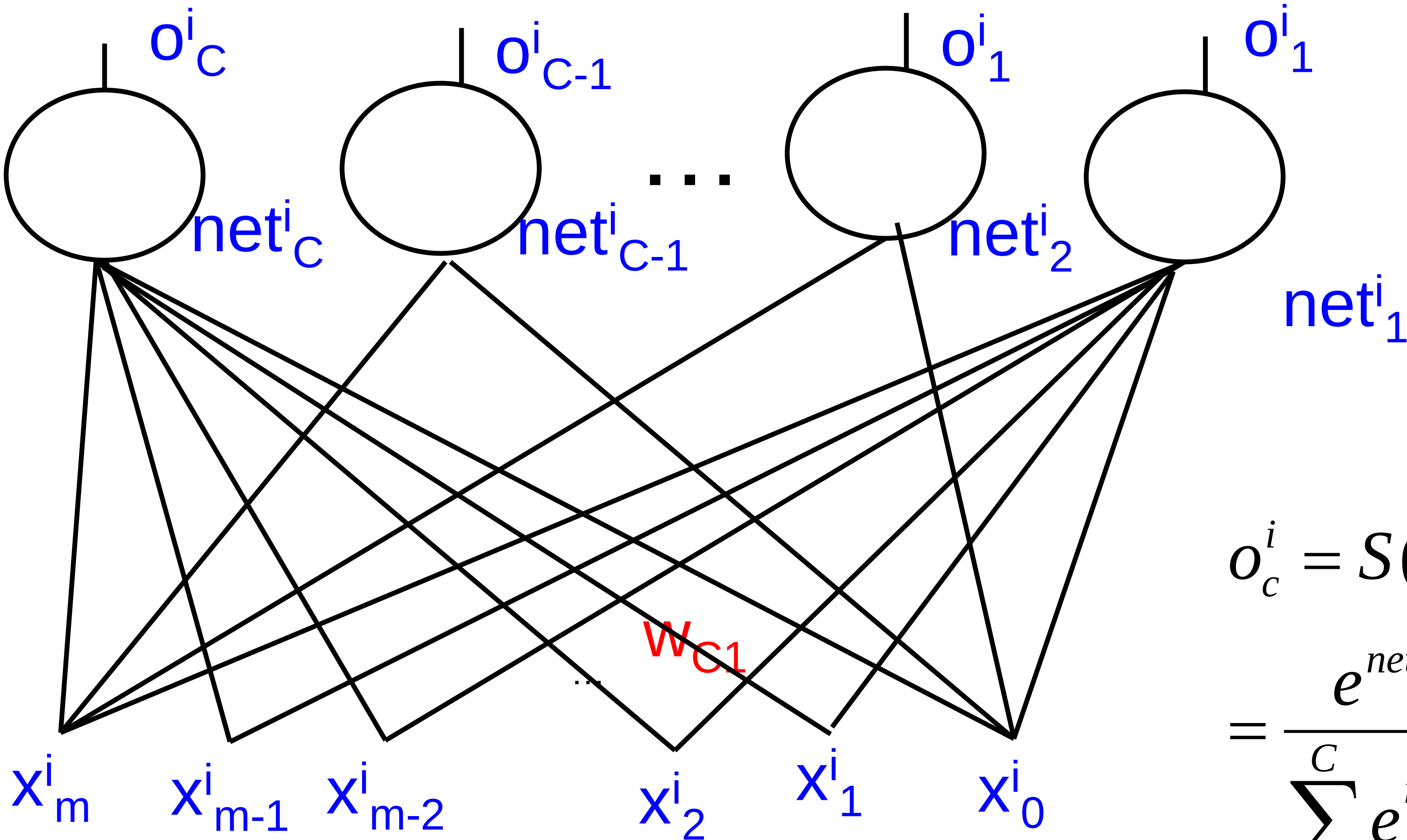
$$= \frac{e^{net_c^i}}{\sum_{k=1}^{C} e^{net_k^i}},$$

Output for class c (small c), c:1 to C

Notation Again

- $i=1..N$, N i-o pairs, i runs over training data
- $j=0\ldots m$, m components in the input vector, j runs over the input dimension (also weight vector dimension)
- $k=1\ldots C$, C classes (C components in the output vector)

Softmax Neuron



$$o^i_c = S(\bar{NET}_i)_c = \frac{e^{net^i_c}}{\sum_{k=1}^{C} e^{net^i_k}},$$

Target Vector, T_i: $<t^i_C\ t^i_{C-1} \ldots t^i_2\ t^i_1>$, $i \rightarrow$ for i^{th} input.

Only one of these C componets is 1, rest are 0.

Compare and contrast Sigmoid and Softmax

$$sigmoid : o_i = \frac{1}{1+e^{-net_i}}, \ for \ i^{th} \ input$$

$$soft\max : o_c^i = \frac{e^{net_c^i}}{\sum_{k=1}^{C} e^{net_k^i}},$$

ith input, *c*th class (small c), *k* varies over classes 1 to *C*

Interpreting o^i_c

- o^i_c value is between 0 and 1
- Interpreted as probability
- Multi-class situation
- o^i_c value is the probability of the class being *'c'* for the i^{th} input
- That is,

  $$P(Class\ of\ i^{th}\ input=c)=o^i_c$$

Derivatives

Derivative of sigmoid

$$o_i = \frac{1}{1+e^{-net_i}},\ for\ i^{th}\ input$$

$$\ln o_i = -\ln(1+e^{-net_i})$$

$$\frac{1}{o_i}\frac{\partial o_i}{\partial net_i} = -\frac{1}{1+e^{-net_i}}.-e^{-net_i} = \frac{e^{-net_i}}{1+e^{-net_i}} = (1-o_i)$$

$$\Rightarrow \frac{\partial o_i}{\partial net_i} = o_i(1-o_i)$$

Derivative of Softmax

$$o_c^i = \frac{e^{net_c^i}}{\sum_{k=1}^{C} e^{net_k^i}}, \; i^{th} \; input \; pattern$$

Derivative of Softmax: Case-1, class *c* for O and NET same

$$\ln o_c^i = net_c^i - \ln\left(\sum_{k=1}^{C} e^{net_k^i}\right)$$

$$\frac{1}{o_c^i}\frac{\partial o_c^i}{\partial net_c^i} = 1 - \frac{1}{\sum_{k=1}^{C} e^{net_k^i}}.e^{net_c^i} = 1 - o_c^i$$

$$\Rightarrow \frac{\partial o_c^i}{\partial net_c^i} = o_c^i(1 - o_c^i)$$

Derivative of Softmax: Case-2, class *c'* in $net^i_{c'}$ different from class c of O

$$\ln o^i_c = net^i_c - \ln(\sum_{k=1}^{C} e^{net^i_k})$$

$$\frac{1}{o^i_c}\frac{\partial o^i_c}{\partial net^i_{c'}} = 0 - \frac{1}{\sum_{k=1}^{C} e^{net^i_k}}.e^{net^i_{c'}} = -o^i_{c'}$$

$$\Rightarrow \frac{\partial O^i_k}{\partial net^i_{c'}} = -o^i_c o^i_{c'}$$