Subject Name: Natural Language Processing

Unit No:3

Unit Name: Syntax Analysis

Faculty Name :Dr. Shilpa Shinde

Index -

D Y PATIL RAMRAO ADIK INSTITUTE OF TECHNOLOGY NAVI MUMBAI

Unit No: 3 Syntax Analysis

Lecture No: 16 Parsing with CFG

Parsing

- Two basic search strategies:
  - Top-down: start at the root of the tree
  - Bottom-up: start at the leaves

Top-down Parsing

- Top-down parsing is goal-directed.
- A top-down parser starts with a list of constituents to be built.
- It rewrites the goals in the goal list by matching one against the LHS of the grammar rules, and expanding it with the RHS, ...attempting to match the sentence to be derived.
- If a goal can be rewritten in several ways, then there is a choice of which rule to apply (search problem)
- Can use depth-first or breadth-first search, and goal ordering.

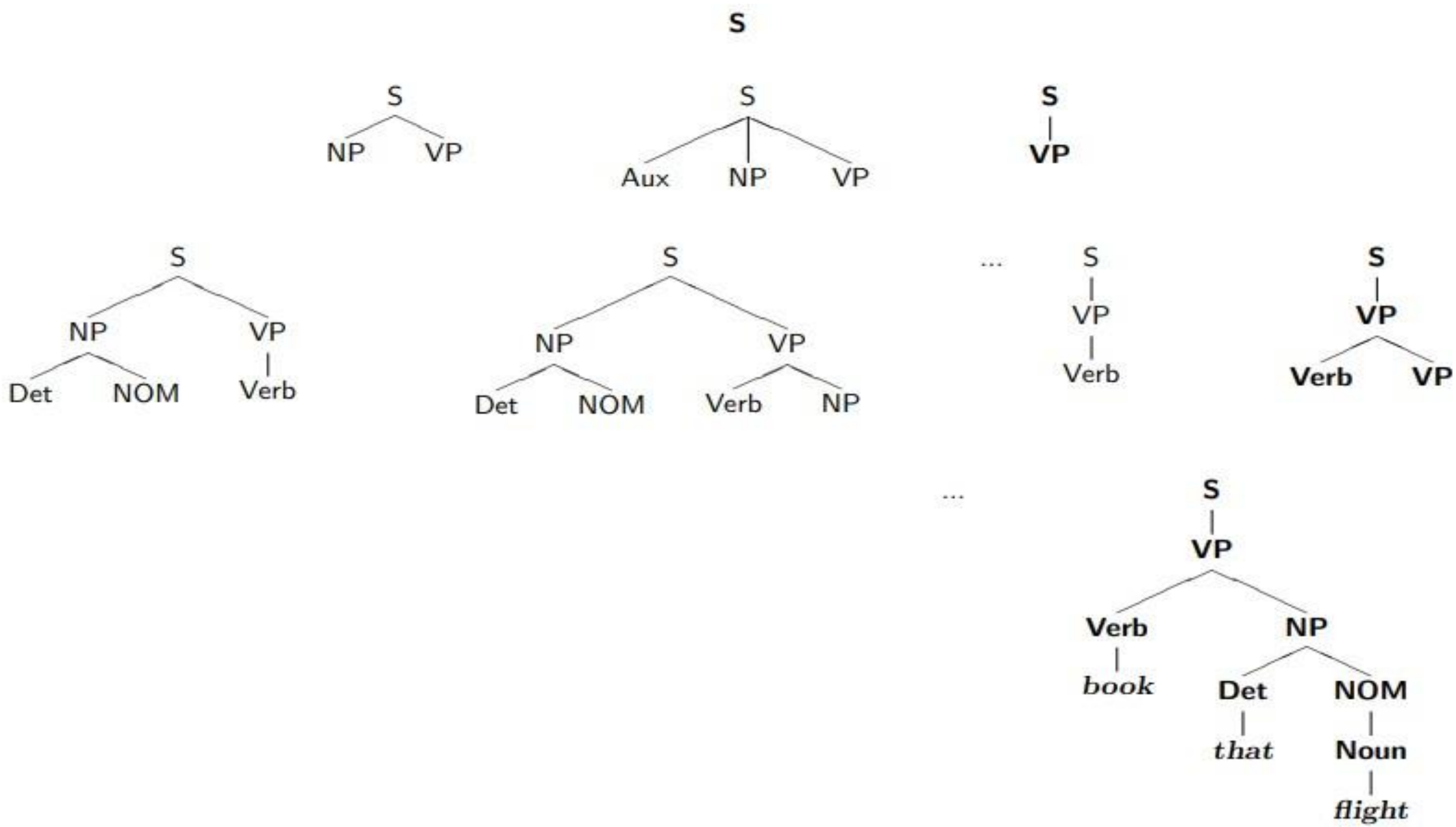
- **Potential problems:**
  - Uses rules that could never match the input
  - May loop on recursive rules: VP → VP PP

Example of Top-down Parsing

S → NP VP	Det → *that* | *this* | *a* | *the*
S → Aux NP VP	Noun → *book* | *flight* | *meal* | *man*
S → VP	Verb → *book* | *include* | *read*
NP → Det NOM	Aux → *does*
NOM → Noun	
NOM → Noun NOM	
VP → Verb	
VP → Verb NP	

Book that flight.

Example of Top-down Parsing



Bottom-up parsing

- Top-down parsing is data-directed.
- The initial goal list of a bottom-up parser is the string to be parsed.
- If a sequence in the goal list matches the RHS of a rule, then this sequence may be replaced by the LHS of the rule.
- Parsing is finished when the goal list contains just the start symbol.
- If the RHS of several rules match the goal list, then there is a choice of which rule to apply (search problem)
- Can use depth-first or breadth-first search, and goal ordering.
- The standard presentation is as shift-reduce parsing.

- **Potential problems**
  - Builds structures that could never be in a tree
  - May loop on epsilon productions: NP $\rightarrow \varepsilon$

Shift Reduce Parser

- Start with the sentence to be parsed in an input buffer.
- a ”shift” action correponds to pushing the next input symbol from the buffer onto the stack
- a ”reduce” action occurrs when we have a rule’s RHS on top of the stack. To perform the reduction, we pop the rule’s RHS off the stack and replace it with the terminal on the LHS of the corresponding rule.
- (When either ”shift” or ”reduce” is possible, choose one arbitrarily.)
- If you end up with only the Start symbol on the stack, then success!
- If you don’t, and you cannot and no ”shift” or ”reduce” actions are possible, backtrack.

Shift-reduce parsing

Stack	Input remaining	Action
()	Book that flight	shift
(Book)	that flight	reduce, Verb → book, (Choice #1 of 2)
(Verb)	that flight	shift
(Verb that)	flight	reduce, Det → that
(Verb Det)	flight	shift
(Verb Det flight)		reduce, Noun → flight
(Verb Det Noun)		reduce, NOM → Noun
(Verb Det NOM)		reduce, NP → Det NOM
(Verb NP)		reduce, VP → Verb NP
(Verb)		reduce, S → V
(S)		SUCCESS!

Top-Down VS. Bottom-Up

Top-down

- – Only searches for trees that can be answers
- – But suggests trees that are not consistent with the words
- – Guarantees that tree starts with S as root
- – Does not guarantee that tree will match input words

Bottom-up

- – Only forms trees consistent with the words
- – Suggest trees that make no sense globally
- – Guarantees that tree matches input words
- – Does not guarantee that parse tree will lead to S as a root

- Combine the advantages of the two by doing a search constrained from both sides (top and bottom)

Context Free Grammars

It is Also known as Phrase structure grammars and Backus-Naur form.

- Consist of Rules:
  - Terminals
  - Non-terminals

Terminals

words

Non-Terminals

The constituents in a language Such as noun phrases, verb phrases and sentences

Rules

Rules are equations that consist of a single non-terminal on the left and any number of terminals and nonterminals on the right.

Context Free Grammars

- Each rule has a left-hand side, which identifies a syntactic category, and a right-hand side, which defines its alternative component parts, reading from left to right.
- Grammatical relations are a formalization of ideas from traditional grammar about SUBJECTS and OBJECTS.
- In the sentence: She ate a mammoth breakfast. The noun phrase **She** is the **SUBJECT** and a **mammoth breakfast** is the **OBJECT**

Context Free Grammars

$G = \langle T, N, S, R \rangle$

- T is set of terminals (lexicon)
- N is set of non-terminals For NLP, we usually distinguish out a set $P \subset N$ of *preterminals* which always rewrite as terminals.
- S is start symbol (one of the nonterminals)
- R is rules/productions of the form $X \rightarrow \gamma$, where X is a nonterminal and γ is a sequence of terminals and nonterminals (may be empty).
- A grammar G generates a language L.

An example context-free grammar

$G = \langle T, N, S, R \rangle$

$T = \{that,\ this,\ a,\ the,\ man,\ book,\ flight,\ meal,\ include,\ read,\ does\}$

$N = \{\text{S, NP, NOM, VP, Det, Noun, Verb, Aux}\}$

$S = S$

$R = \{$

S → NP VP
S → Aux NP VP
S → VP
NP → Det NOM
NOM → Noun
NOM → Noun NOM
VP → Verb
VP → Verb NP

Det → *that* | *this* | *a* | *the*
Noun → *book* | *flight* | *meal* | *man*
Verb → *book* | *include* | *read*
Aux → *does*

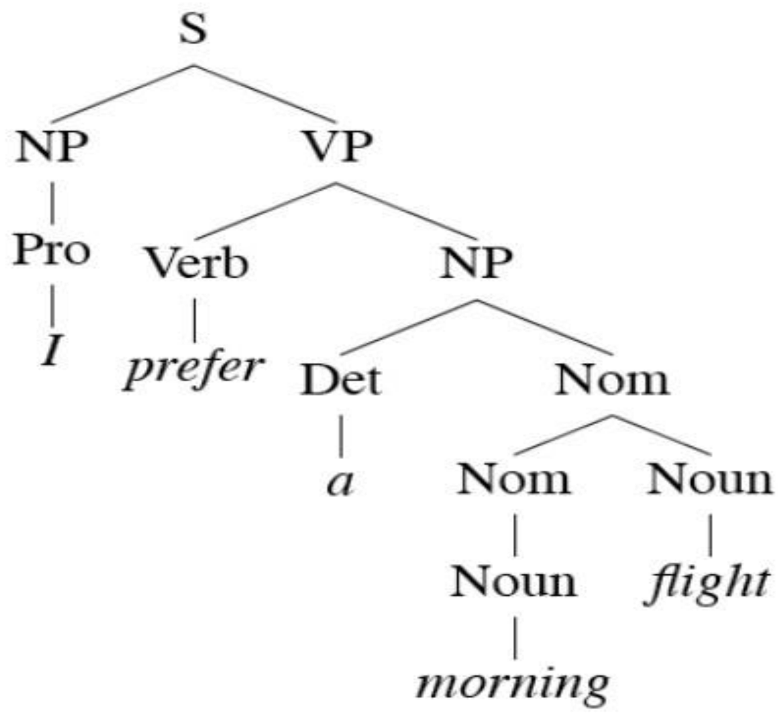
$\}$

Grammer Rules

Grammar Rules			Examples
S	$\rightarrow$	$NP\ VP$	I + want a morning flight
NP	$\rightarrow$	$Pronoun$	I
	$\mid$	$Proper\text{-}Noun$	Los Angeles
	$\mid$	$Det\ Nominal$	a + flight
$Nominal$	$\rightarrow$	$Nominal\ Noun$	morning + flight
	$\mid$	$Noun$	flights
VP	$\rightarrow$	$Verb$	do
	$\mid$	$Verb\ NP$	want + a flight
	$\mid$	$Verb\ NP\ PP$	leave + Boston + in the morning
	$\mid$	$Verb\ PP$	leaving + on Thursday
PP	$\rightarrow$	$Preposition\ NP$	from + Los Angeles

Derivation

- A "derivation" is a sequence of rules applied to a string that accounts for that string.



Context Free Grammars: Example

s → np vp

np → det n

vp → tv np

→ iv

det → the

→ a

→ an

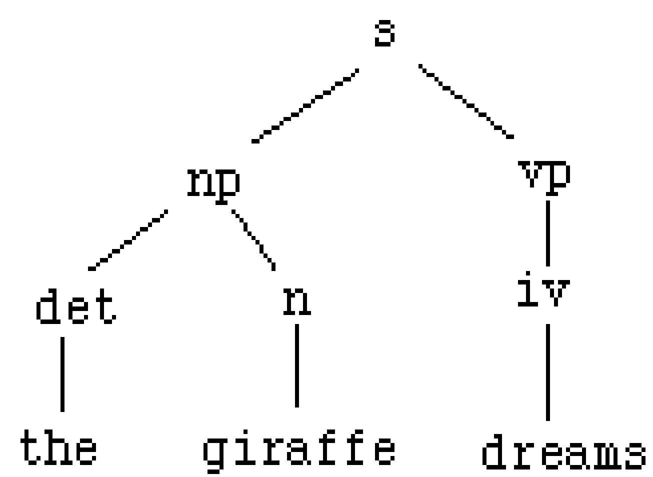
n → giraffe

→ apple

iv → dreams

tv → eats

→ dreams



Potential Problems in Context Free Grammars

- While context-free grammar can account for much of the syntactic structure of English, it is not a perfect solution.

- Agreement
- Subcategorization
- Movement

Number Agreement

- Constraints that hold among various constituents.
- For example, in English, determiners and the head nouns in NPs have to agree in their number.
- Which of the following cannot be parsed by the rule
- NP --→ Det Nominal ?
- This rule does not handle agreement! (The rule does not detect whether the agreement is correct or not.)
  - (O) This flight
  - (O) Those flights
  - (X) This flights
  - (X) Those flight

- **And verbs agree in number with their subjects:**
  - ✓ What flights leave in the morning?
  - ✓ *What flight leave in the morning?

Number Agreement

Expand our grammar with multiple sets of rules?

Grammar rule	Example
S → NP[sg] VP[sg]	this flight + leaves on Monday
NP[sg] → Det[sg] Nom[sg]	this + flight
VP[sg] → Verb[sg] PP	leaves + on Monday
NP[pl] → Det[pl] NP[pl]	these + flights

While this approach is sound, it is practically infeasible: the grammars get too large

Subcategorization

- English VPs consist of a main verb along with zero or more constituents that we can call arguments

Grammar rule	Example
VP → Verb	sleep
VP → Verb NP	want + a flight
VP → Verb NP PP	leave + Boston + in the morning
VP → Verb PP	leave + on Thursday

But not all verbs are allowed to participate in all of those rules: we need to subcategorize them

D Y PATIL RAMRAO ADIK INSTITUTE OF TECHNOLOGY NAVI MUMBAI

Subcategorization Frames

Modern grammars may have several hundreds of subcategories.
Examples:

Verb	Example
sleep	John slept.
find + NP	Please find [a flight to New York].
give + NP + NP	Give [me] [a cheaper fare].
help + NP + PP	Can you help [me] [with a flight]?
prefer + TO-VP	I prefer [to leave earlier].
told + S	I was told [United has a flight].

Movement

- *I gave __to the driver.*
- *I gave some money to the driver.*

- $5 [*I gave __to the driver*], (*and $1 I gave to the porter*).
- *He asked how much* [*I gave __to the driver*].
- *I forgot about the money which* [*I gave __to the driver*].

- *How much did you think* [*I gave __to the driver*]?
- *How much did you think he claimed* [*I gave __to the driver*]?
- *How much did you think he claimedthat I said* [*I gave __to the driv*
- . . .

Lecture No: 20Sequence labelling: Hidden Markov Model (HMM)

Sequence labelling

- In natural language processing, it is a common task to extract words or
  phrases of particular types from a given sentence or paragraph.
- For example, when performing analysis of a corpus of news articles, we may want to know which countries are mentioned in the articles, and how many articles are related to each of these countries.
- This is actually a special case of sequence labelling in NLP (others include POS tagging and Chunking), in which the goal is to assign a label to each member in the sequence.
- input = ["Paris", "is", "the", "capital", "of", "France"]
- output = ["I", "I", "I", "I", "I", "C"]

Sequence labelling

- A simple, though sometimes quite useful, approach is to prepare a dictionary of country names, and look for these names in each of the sentences in the corpus.
- Hidden Markov Model (HMM)
- Maximum Entropy
- Conditional Random Field (CRF)

HMM Model

- We cannot determine the exact sequence of tags that generated and calculate using t = argmax P(w, t) and it is based on the Markovian assumption that the current tag depends only on the previous n tags.
- Use transition probability(i.e. forward tag and backward tags).
- P (ti/wi) = P (ti/ti-1). P (ti+1/ti). P (wi/ti)........... (6)
- P (ti/ti-1) is the probability of current tag given previous tag.
- P (ti+1/ti) is the probability of future tag given current tag.
- P (wi/ti) Probability of word given current tag

HMM Model

- Bigram tagger Make predictions based on the preceding tag .The basic unit
  is the preceding tag and the current tag

- Trigram tagger We would expect more accurate predictions if more context is taken into account

- RB(adverb) VBD(past tense) Vs RB VBN(past participle) ? Ex) “clearly marked” Is clearly marked : P(BEZ RB VBN) > P(BEZ RB VBD) He clearly marked : P(PN RB VBD) > P(PN RB VBN)

HMM Model

- Assigns each word to its most common tag and consider one word at a a time.
- P (ti/wi) = freq (wi/ti)/freq (wi)
- Here Probability of tag given word is computed by frequency count of word given tag divided by frequency count of that particular word

HMM Model

- It based on preceding tag i.e. it take two tags: the preceding tag and current tag into account.

- P (ti/wi) = P (wi/ti). P (ti/ti-1)

Here P (wi/ti) is the probability of current word given current tag P (ti/ti-1) is the probability of a current tag given the previous tag

HMM Model

- It based on previous two tags.
- P (ti/wi) = P (wi/ti). P (ti/ti-2, ti-1)
- Where ti denotes tag sequence and wi denote word sequence.
- P (wi/ti) is the probability of current word given current tag.
- Here, P(ti|ti-2, ti-1)is the probability of a current tag given the previous two tags

An Example

- Secretariat/NNP is/VBZ expected/VBN to/TO race/VB tomorrow/NN
- People/NNS continue/VBP to/TO inquire/VB the DT
- reason/NN for/IN the/DT race/NN for/IN outer/JJ
- space/NN
- to/TO race/???
- the/DT race/???
- ti = argmaxj P(tj|ti-1)P(wi|tj) max[P(VB|TO)P(race|VB) , P(NN|TO)P(race|NN)]
- Brown: P(NN|TO) = .021 × P(race|NN) = .00041 = .000007 P(VB|TO) = .34 × P(race|VB) = .00003 = .00001

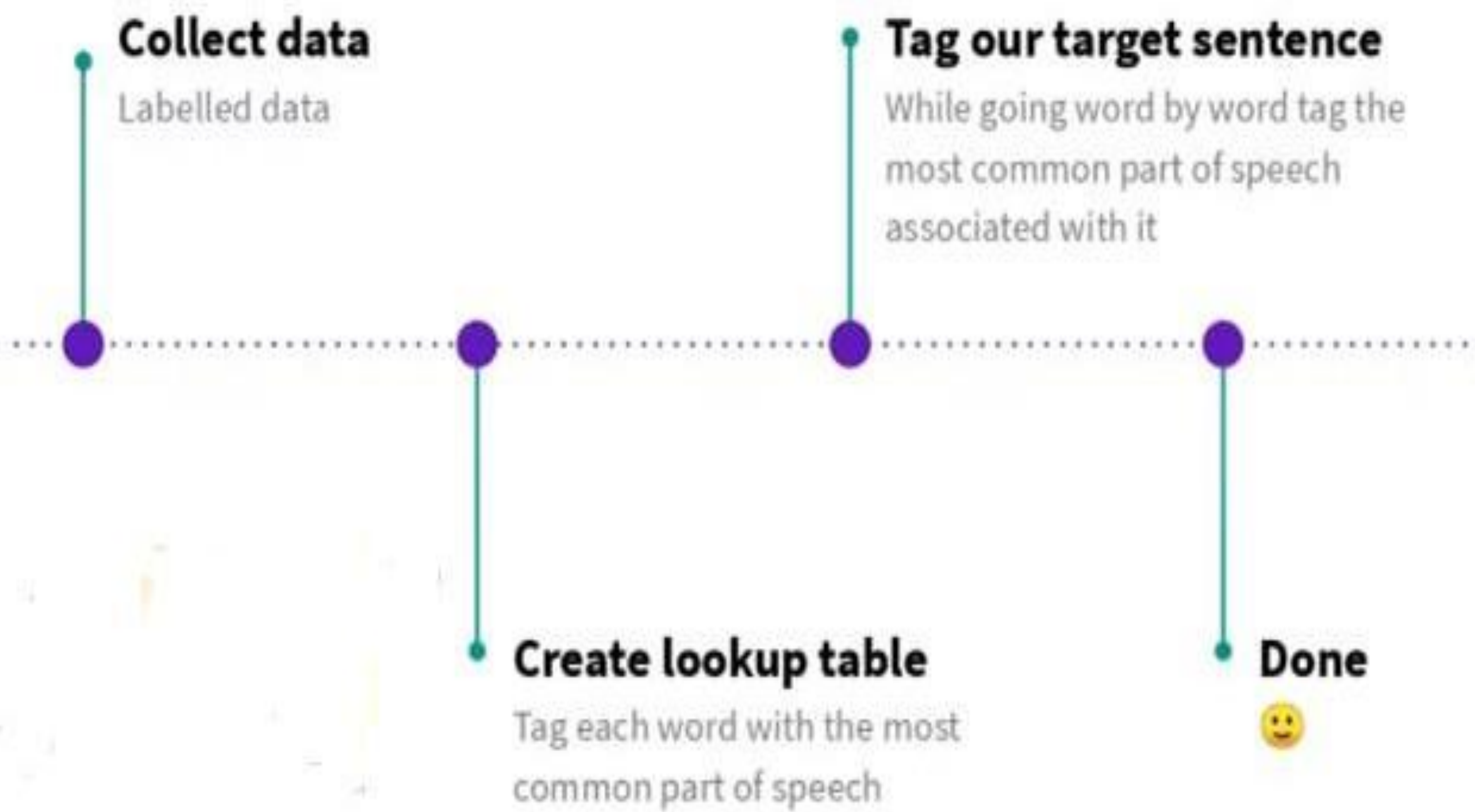
Lecture No: 21 Sequence labelling: Hidden Markov Model (HMM)

HMM Tagger Example

Mary saw Will.

Jane	saw	Will	Marry	saw	Jane
noun	verb	noun	noun	verb	noun

HMM Tagger



Collect data
Labelled data
Create lookup table
Tag each word with the most common part of speech
Tag our target sentence
While going word by word tag the most common part of speech associated with it
Done

HMM Tagger Example

Lookup Table

Mary saw Will.

Jane	saw	Will
noun	*verb*	*noun*
Marry	saw	Jane
noun	*verb*	*noun*

Mary	saw	Will
noun	*verb*	*noun*

	N	V
Mary	1	0
saw	0	2
Jane	2	0
Will	1	0

Problem

Lookup Table

Marry	will	see	will
?	?	?	?

Our data!

Mary	will	see	Jane
noun	*modal*	*verb*	*noun*
Will	will	see	Mary
noun	*modal*	*verb*	*noun*
Jane	will	see	Will
noun	*modal*	*verb*	*noun*

	N	V	M
Mary	2	0	0
see	0	3	0
Jane	2	0	0
Will	2	0	3

HMM Tagger

- **Tigram** is based on previous two tags.

$$P\ (ti/wi) = P\ (wi/ti).\ P\ (ti/ti\text{-}2,\ ti\text{-}1)$$

Where ti denotes tag sequence and wi denote word sequence.

- P (wi/ti) is the probability of current word given current tag.
- Here, P(ti|ti-2, ti-1)is the probability of a current tag given the previous two tags.

Using Bi Grams

Lookup Table

Marry	will	see	will
noun	*modal*	*verb*	*noun*

Our data!

Mary	will	see	Jane
noun	*modal*	*verb*	*noun*
Will	will	see	Mary
noun	*modal*	*verb*	*noun*
Jane	will	see	Will
noun	*modal*	*verb*	*noun*

BIGRAMS

	N-M	M-V	V-N
mary-will	1	0	0
will-see	0	3	0
see-jane	0	0	1
will-will	1	0	0
see-mary	0	0	1
jane-will	1	0	0
see-will	0	0	1

Problem with Bigrams

Our data

Mary Jane can see Will

Spot will see Mary

Will Jane spot Marry?

Marry will pat Spot

Jane	will	spot	will
?	?	?	?

	N-M	M-V	V-N	etc...
mary-jane				
jane-can				
can-see				
see-will				
spot-will				
will-see				
see-mary				
will-jane				
jane-spot				
spot-mary				
mary-will				
will-pat				
pat-spot				

HMM

Jane	will	spot	will
noun	*modal*	*verb*	*noun*

We will need two things

Two probabilities

Transition Probabilities

How likely is it that…
Noun is followed by a modal
which is followed by a verb and
than a noun

Emission probabilities

How likely is that Jane will be a noun, will be a modal,…

…

🙂

Emission Probabilities

Emission Probabilities

	N	M	V
Mary	4/9	0	0
Jane	2/9	0	0
Will	1/9	3/4	0
Spot	2/9	0	1/4
Can	0	1/4	0
See	0	0	1/2
Pat	0	0	1/4

N N M V N
Mary Jane can see Will.

N M V N
Spot will see Mary.

M N V N
Will Jane spot Mary?

N M V N
Mary will pat Spot

	N	M	V
Mary	4	0	0
Jane	2	0	0
Will	1	3	0
Spot	2	0	1
Can	0	1	0
See	0	0	2
Pat	0	0	1

Transition Probabilities

Transition Probabilities

	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	1/3	1/9	4/9
M	1/4	0	3/4	0
V	1	0	0	0

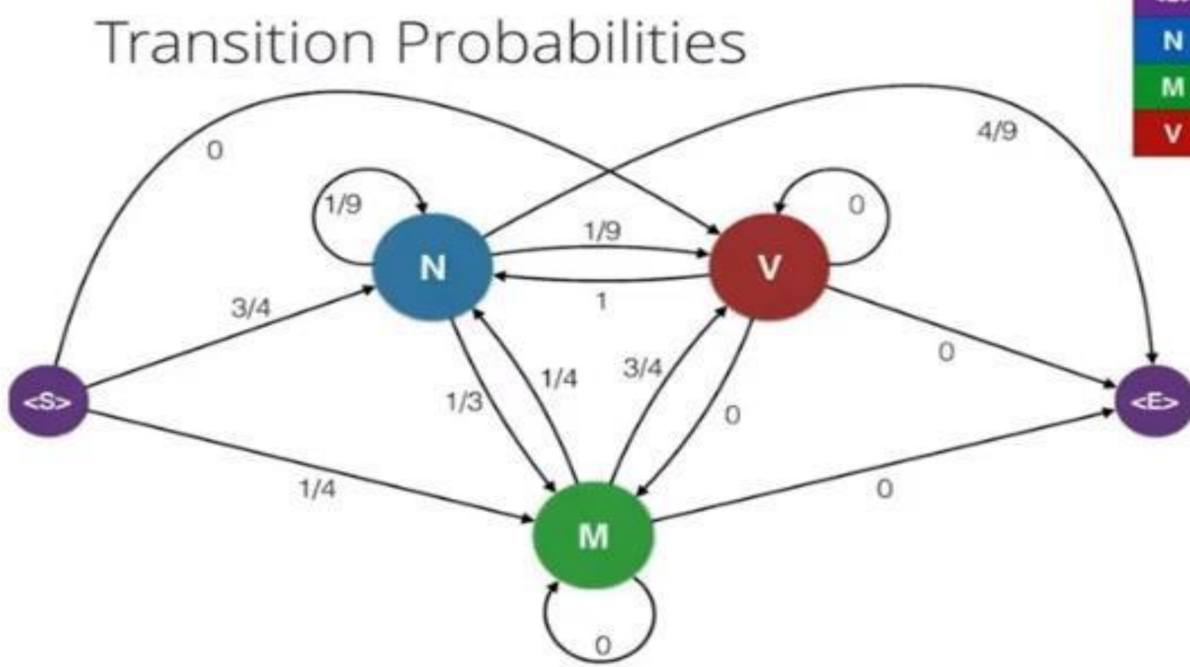
<S> N N M V N <E>
Mary Jane can see Will.

<S> N M V N <E>
Spot will see Mary.

<S> M N V N <E>
Will Jane spot Mary?

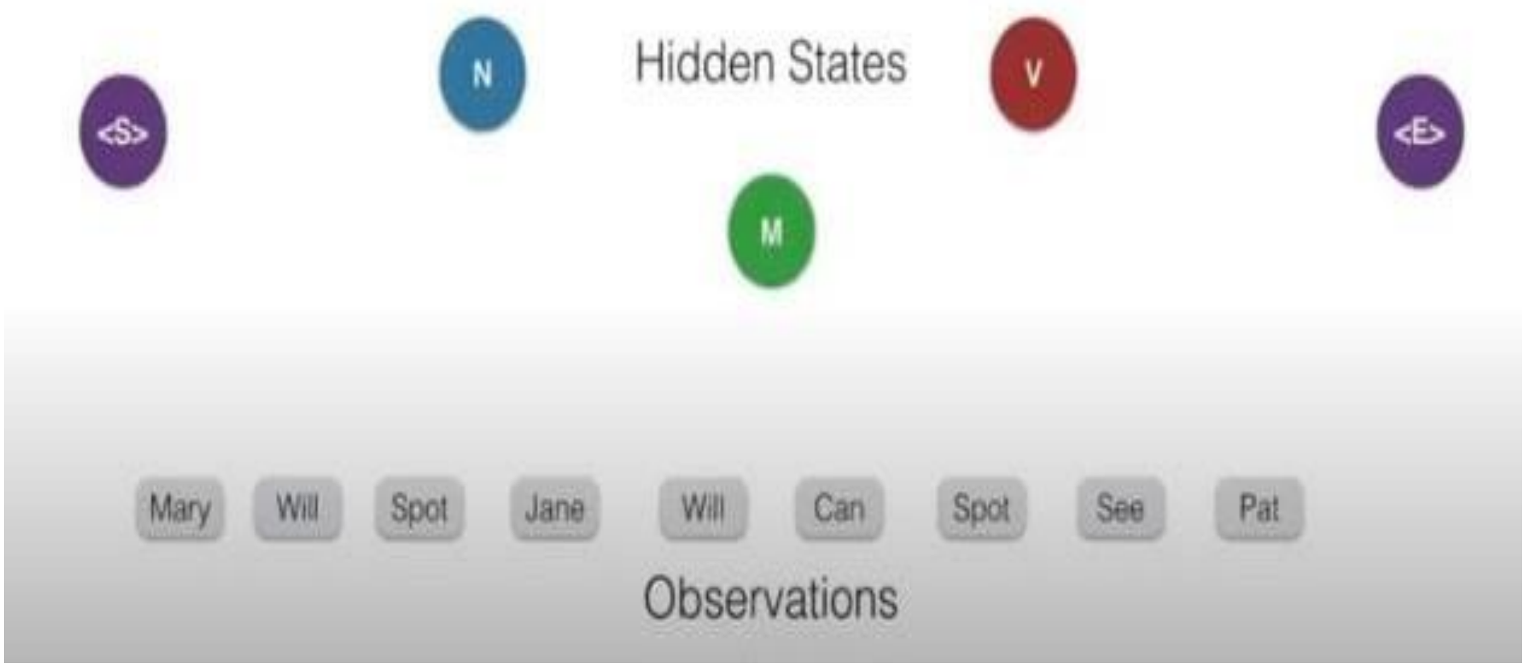
<S> N M V N <E>
Mary will pat Spot

Transition Probabilities



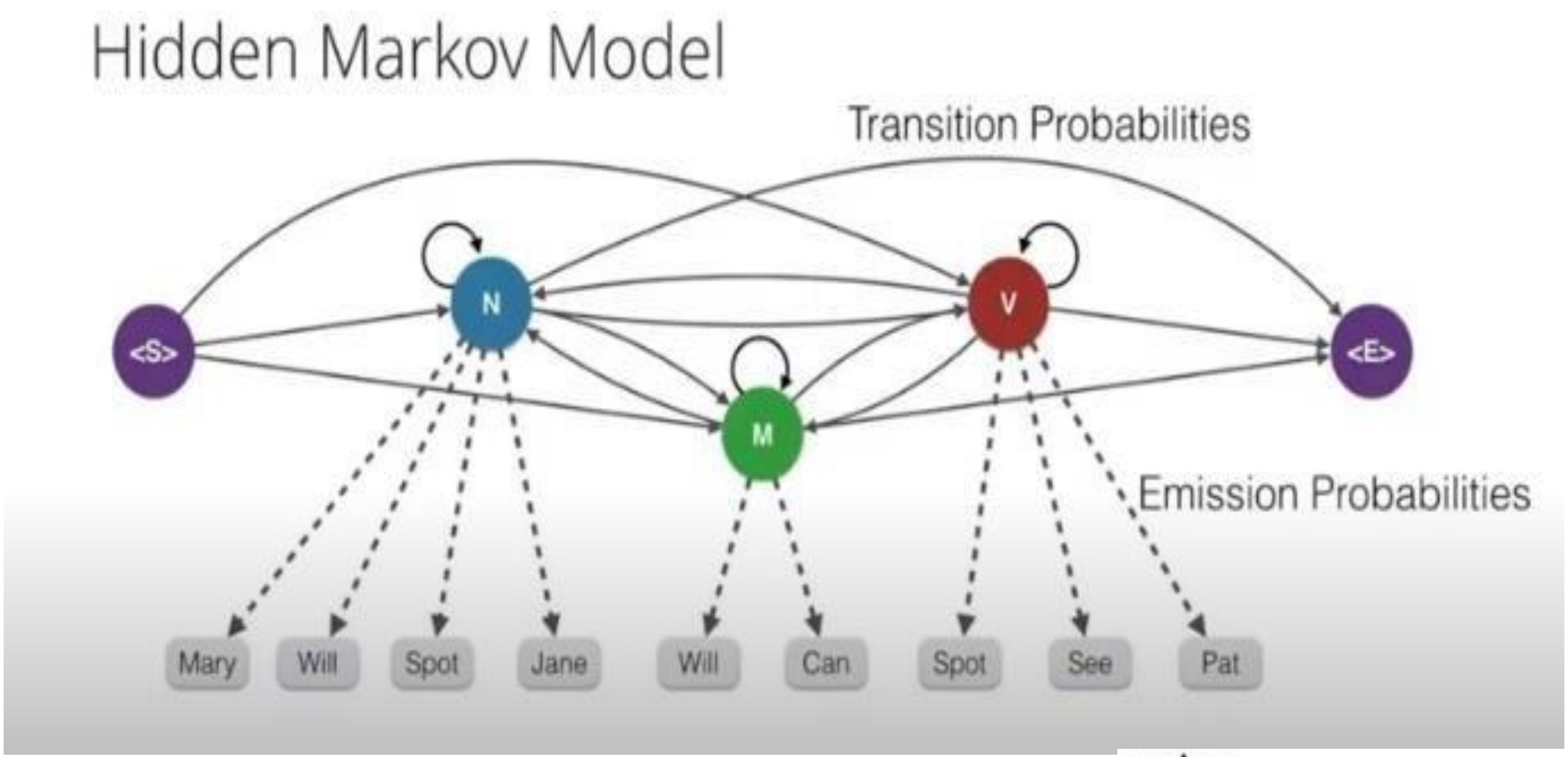
	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	1/3	1/9	4/9
M	1/4	0	3/4	0
V	1	0	0	0

Hidden States



D Y PATIL DEEMED TO BE UNIVERSITY RAMRAO ADIK INSTITUTE OF TECHNOLOGY NAVI MUMBAI

HMM



D Y PATIL DEEMED TO BE UNIVERSITY RAMRAO ADIK INSTITUTE OF TECHNOLOGY NAVI MUMBAI

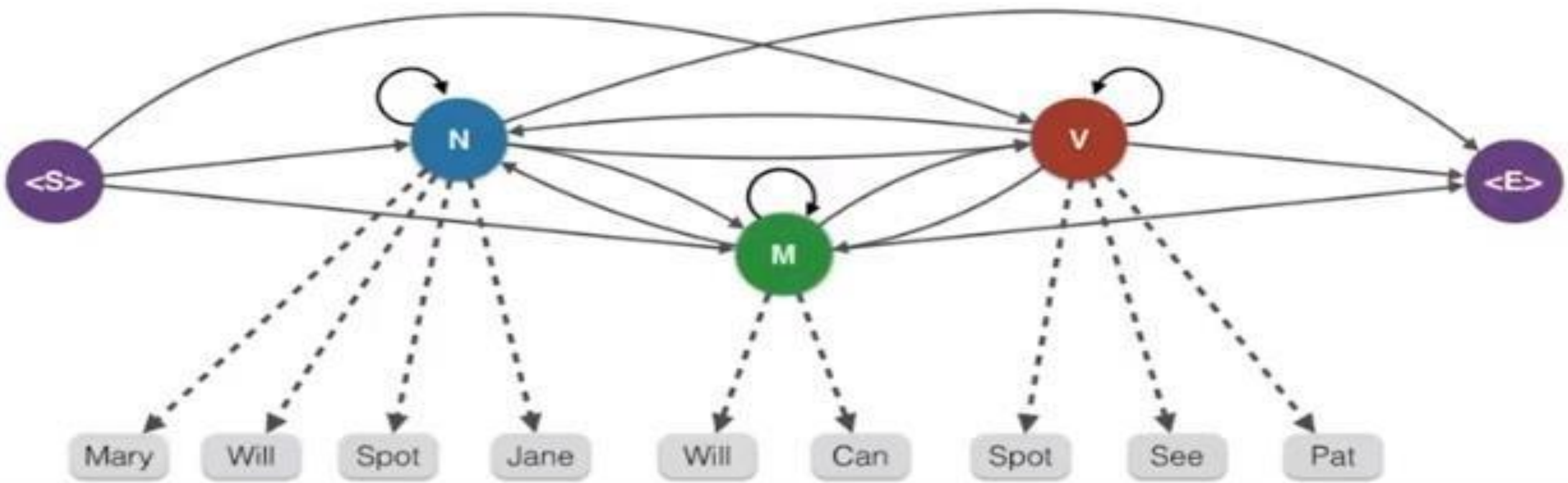
HMM

Emission Probabilities

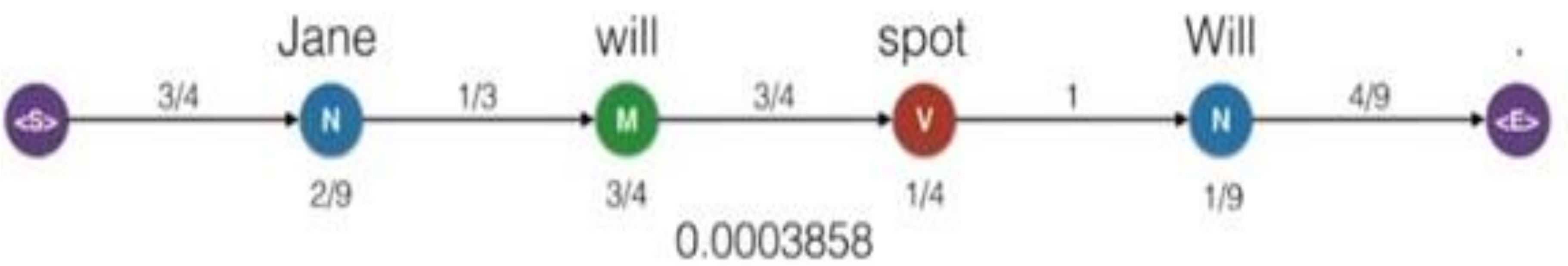
	N	M	V
Mary	4/9	0	0
Jane	2/9	0	0
Will	1/9	3/4	0
Spot	2/9	0	1/4
Can	0	1/4	0
See	0	0	1/2
Pat	0	0	1/4

Transition Probabilities

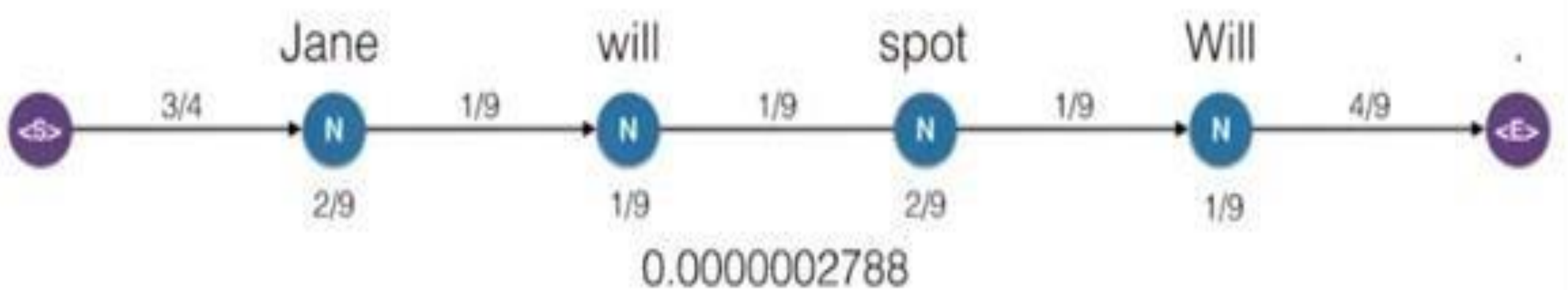
	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	1/3	1/9	4/9
M	1/4	0	3/4	0
V	1	0	0	0



HMM



Now how we know this is correct ?? So consider that the probability of hidden state is noun we get the probability as



Possibilities??

Try this : Will can spot Mary' be tagged as-

Video

- https://www.youtube.com/watch?v=68hmUltbPnw

Thank You

D Y PATIL
RAMRAO ADIK INSTITUTE OF TECHNOLOGY
NAVI MUMBAI