

The Eternal Relevance of Part of Speech Tagging

Pushpak Bhattacharyya
CSE Dept,
IIT Patna and Bombay

Overview, POS tagging

Road Map

- Introduction
 - NLP, ML, NLP-ML
 - Ambiguity
- POS tagging
 - Linguistics
 - Challenges
- HMM Based
- CRF Based
- Neural Net Based
- Deep Learning Based

What is NLP?

- Make computers understand and produce natural language
- Natural languages are languages used by humans for day to day living (Hindi, English, Bangla, Nepali, French, Portugese, Sinhala, Tahiti, Swahili and so on)
- (**Wikipedia**) “There are roughly **6,500 spoken languages** in the world today. However, about 2,000 of those languages have fewer than 1,000 speakers. The most popular language in the world is Mandarin Chinese. There are **1,213,000,000** people in the world that speak that language.”

Need for NLP

- Humongous amount of language data in electronic form
- Unstructured data (like free flowing text) will grow to 40 zetabytes (1 zettabyte= 10^{21} bytes) by 2020.
- How to make sense of this huge data?
- Example-1: e-commerce companies need to know **sentiment** of online users, sifting through 1 lakh e-opinions per week: needs NLP
- Example-2: **Translation** industry to grow to \$37 billion business by 2020

What is Machine Learning

- Automatically learning rules and concepts from data



Learning the concept of table.

What is “tableness”

Rule: a flat surface with 4 legs (approx.: to be refined gradually)

Machine Learning (contd)

- Positive examples and negative examples of a concept: Tables and non-tables
- Labelled data- supervised learning (classification)
- Unlabelled data- unsupervised learning (clustering)

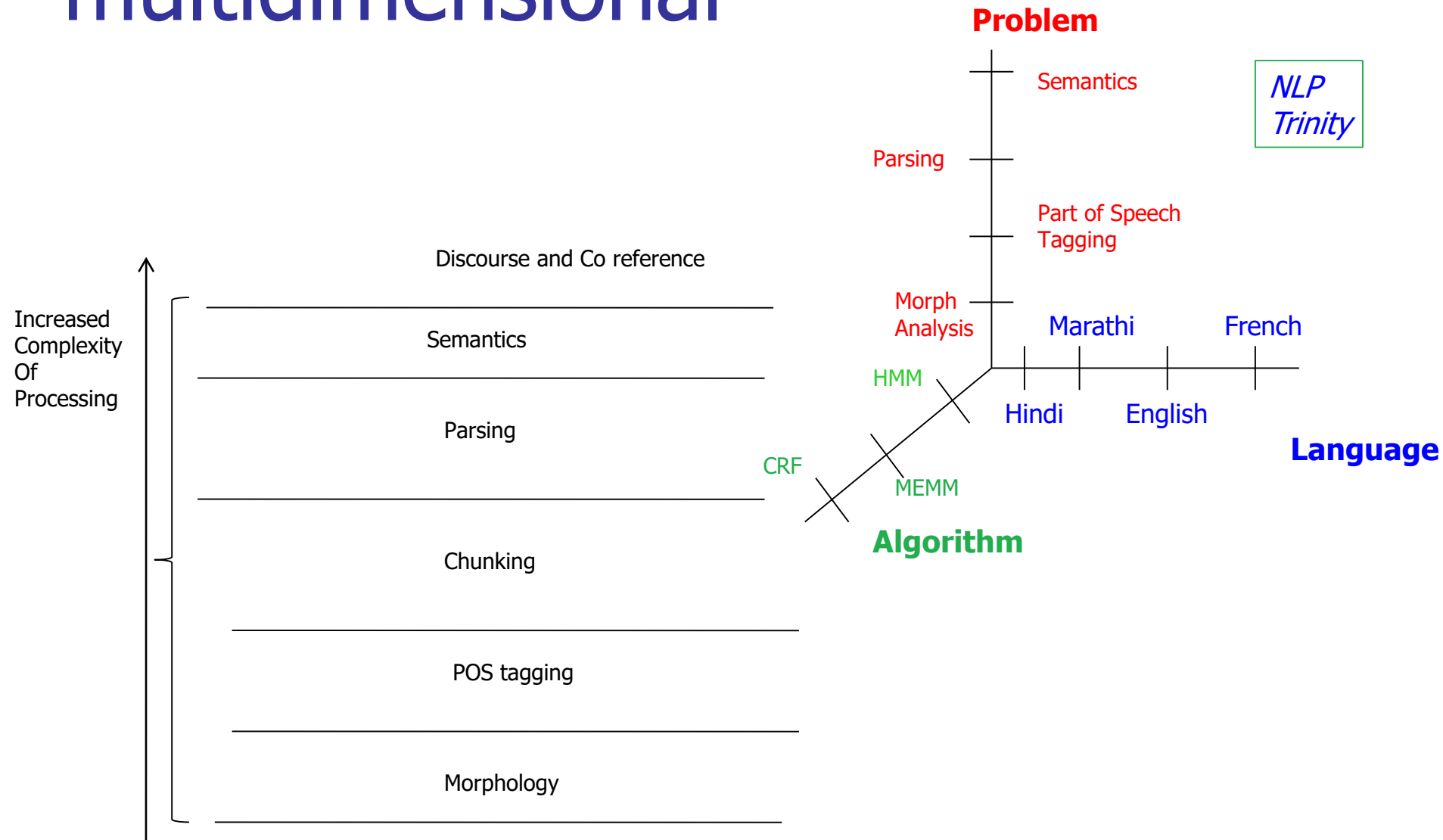
Why NLP and ML?

- Impossible for humans (single or a team) to makes sense of and analyse humongous text data
- Many processing steps in NLP
- Impossible to give correct-consistent-complete rules covering each and every situation

NLP-ML marriage



NLP: layered, multidimensional



NLP= Ambiguity Processing

- Lexical Ambiguity
- Structural Ambiguity
- Semantic Ambiguity
- Pragmatic Ambiguity

Lexical Disambiguation

First step: *part of Speech Disambiguation*

- *Dog* as a *noun* (animal)
- *Dog* as a verb (*to pursue*)

Sense Disambiguation

- *Dog* (as *animal*)
- *Dog* (as *a very detestable person*)

Needs word relationships in a context

- *The chair emphasised the need for adult education*

Very common in day to day communications

Satellite Channel Ad: *Watch what you want, when you want* (two senses of watch)

e.g., Ground breaking ceremony/research

Ambiguity of Multiwords

- *The grandfather kicked the bucket after suffering from cancer.*
- *This job is a piece of cake*
- *Put the sweater on*
- *He is the dark horse of the match*

Google Translations of above sentences:

दादा कैंसर से पीड़ित होने के बाद बाल्टी लात मारी.

इस काम के केक का एक टुकड़ा है.

स्वेटर पर रखो.

वह मैच के अंधेरे घोड़ा है.

Ambiguity of Named Entities

- Bengali: চঞ্চল সরকার বাড়িতে আছে
English: *Government is restless at home. (*)*
Chanchal Sarkar is at home
- Hindi: दैनिक दबंग दुनिया
English: everyday bold world
Actually name of a Hindi newspaper in Indore

A big challenge- ambiguity of function words

- I saw the boy with a telescope
 - *Who has the telescope?*
- मुर्गियां अंडो के कारण है (murgiyaan ando ke kaaran hai)
 - *A chicken and egg problem indeed!*
- आपको मुझे मिठाई खिलानी पड़ेगी (aapako mujhe mithaee khilaanee padegee)
 - *Who gives sweets to whom?*
- युद्ध आर्थिक दूरवश्र कारण (Yud'dha ārthika durabasthāra kāraṇa) (war is the cause of financial distress)
 - No ambiguity! Case marker attached to the word

Use of Ambiguity: Humour

1. If money does not grow on trees, why do banks have **branches**?

2. Person-1: Have you heard of the heavy **firing**?

Person-2: No, which **sector**?

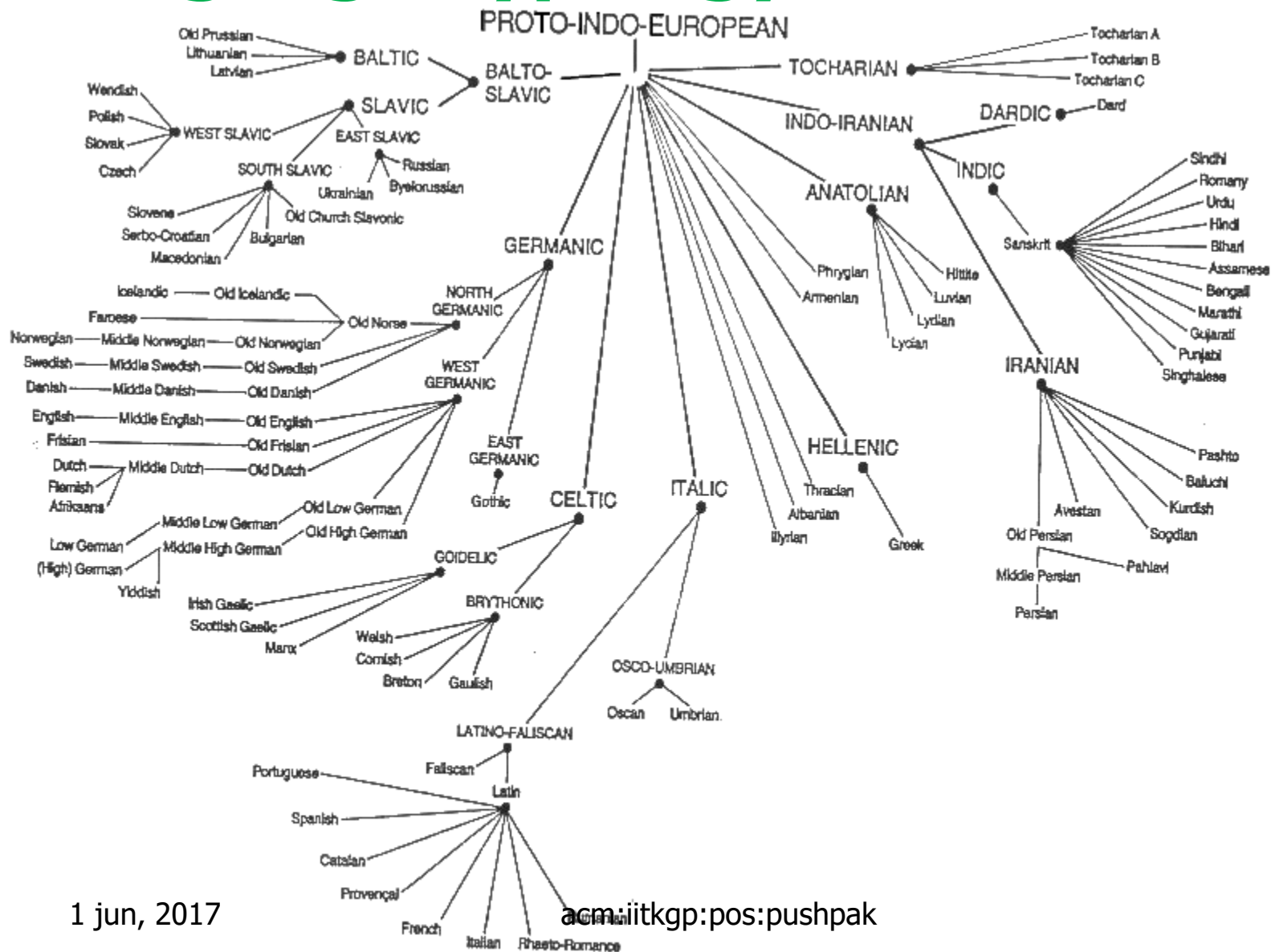
Person-1: IT **Sector**

3. Person-1: I have got a new car **for** my unemployed son

Person-2: That is a great exchange!

NLP: deal with multilinguality

Language Typology



Rules: when and when not

- When the phenomenon is understood AND expressed, rules are the way to go
- “Do not learn when you know!!”
- When the phenomenon “seems arbitrary” at the current state of knowledge, DATA is the only handle!
 - *Why do we say “Many Thanks” and not “Several Thanks”!*
 - *Impossible to give a rule*

Impact of probability: Language modeling

Probabilities computed in the context of corpora

1. $P(\text{"The sun rises in the east"})$
2. $P(\text{"The sun rise in the east"})$
 - Less probable because of grammatical mistake.
3. $P(\text{The svn rises in the east})$
 - Less probable because of lexical mistake.
4. $P(\text{The sun rises in the west})$
 - Less probable because of semantic mistake.

Power of Data- Automatic image labeling

(Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan, 2014)



Automatically captioned: "Two pizzas sitting on top of a stove top oven"

Automatic image labeling (cntd)

Describes without errors



A person riding a motorcycle on a dirt road.



A group of young people playing a game of frisbee.



A herd of elephants walking across a dry grass field.

Describes with minor errors



Two dogs play in the grass.



Two hockey players are fighting over the puck.



A close up of a cat laying on a couch.

Somewhat related to the image



A skateboarder does a trick on a ramp.



A little girl in a pink hat is blowing bubbles.



A red motorcycle parked on the side of the road.

Unrelated to the image



A dog is jumping to catch a frisbee.



A refrigerator filled with lots of food and drinks.



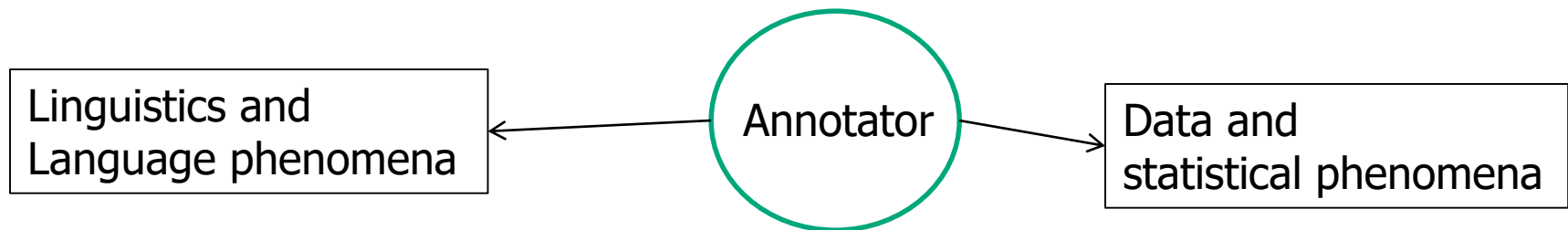
A yellow school bus parked in a parking lot.

Main methodology

- Object A: extract parts and features
- Object B which is in correspondence with A: extract parts and features
- LEARN mappings of these features and parts
- Use in NEW situations: called DECODING

Linguistics-Computation Interaction

- Need to understand BOTH language phenomena and the data
- An annotation designer has to understand BOTH linguistics and statistics!



Part of Speech Tagging

With Hidden Markov Model

NLP Layer

What a gripping movie was Dangal!

What/WP a/DT gripping/JJmovie/NN was/VBD Dangal/NNP !/.

Parse

```
(ROOT
  (FRAG
    (SBAR
      (WHNP
        (WP What))
      (S
        (NP
          (DT a)
          (JJ gripping)
          (NN movie)
        )
        (VP
          (VBD was)
          (NP
            (NNP Dangal))))))
    )
  )
)
```

Universal dependencies

```
dobj(Dangal-6, What-1)
det(movie-4, a-2)
amod(movie-4, gripping-3)
nsubj(Dangal-6, movie-4)
cop(Dangal-6, was-5)
root(ROOT-0, Dangal-6)
```


Part of Speech Tagging

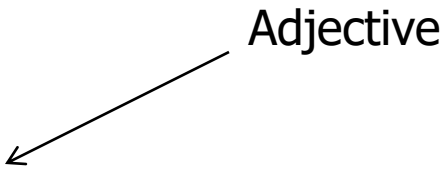
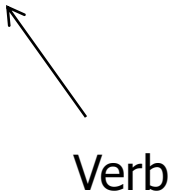
- POS Tagging: attaches to each word in a sentence a part of speech tag from a given set of tags called the **Tag-Set**
- Standard Tag-set : Penn Treebank (for English).

POS ambiguity instances

best ADJ ADV NP V
better ADJ ADV V DET
close ADV ADJ V N
cut V N VN VD
even ADV DET ADJ V
grant NP N V –
hit V VD VN N
lay ADJ V NP VD
left VD ADJ N VN
like CNJ V ADJ P –
near P ADV ADJ DET
open ADJ V N ADV
past N ADJ DET P
present ADJ ADV V N
read V VN VD NP
right ADJ N DET ADV
second NUM ADV DET N
set VN V VD N –
that CNJ V WH DET

Part-of-speech tag

- A word can have more than one POS tags.

- E.g.
 1. *What a **gripping** movie was Dangal!*
Adjective
 2. *He is **gripping** it firm.*
Verb

Linguistic fundamentals

- A word can have two roles
 - Grammatical role (Dictionary POS tag)
 - Functional role (Contextual POS tag)
- E.g. **Golf stick**
- POS tag of “*Golf*”
 - Grammatical: Noun
 - Functional: Adjective (+ al)

The “al” rule!

- If a word has different functional POS tag than its grammatical pos then add “**al**” to the functional POS tag

- E.g. *Golf stick*

Adjective + al
└──────────┘
Adjectival

Noun	+ al	= Nominal
Verb	+ al	= Verbal
Adjective	+ al	= Adjectival
Adverb	+ al	= Adverbial

The “al” rule cntd.

- Examples:

- Nominal

- *Many don't understand the problem of **hungry**.*

- Adverbial

- *Come **quick**.*

- Verbal

POS tagging as an ML problem

■ Question

- Is one instance of example enough for ML?
- E.g. Known example of “people”

People → Noun

POS Ambiguity

- But it can be verb as well

People → Verb (to populate)

■ Answer

- We need at least as many instances as number of different labels (POS tags)-1 to make decision.

Disambiguation of POS tag

- If no ambiguity, learn a table of words and its corresponding tags.
- If ambiguity, then look for the contextual information, i.e., look-back or look-ahead.

Rule based disambiguation: instances of “present”

1. He gifted me the/a/this/that **present_NN**.
2. They **present_VB** innovative ideas.
3. He was **present_JJ** in the class.

Rules for disambiguating “present”

- For Present_NN (look-back)
 - If present is preceded by determiner (the/a) or demonstrative (this/that), then POS tag will be noun.
 - Does this rule guarantee 100% precision and 100% recall?
 - False positive:
 - *The **present_ADJ** case is not convincing.*
Adjective preceded by “the”
 - False negative:
 - ***Present** foretells future.*

Noun but not preceded by “the”

Rules for disambiguating “present” (Look back + look ahead)

- For Present_NN (look-back and look ahead)
 - If present is preceded by determiner (the/a) or demonstrative (this/that) or followed by a verb, then POS tag will be noun.
 - E.g.
 - **Present_NN** will tell the future.
 - **Present_NN** foretells the future.
 - Does this rule guarantee 100% precision and 100% recall?

Answer: P and R improved, but...

- The new rule has higher precision and recall, i.e., lower false positive and false negative
- But not really 100% full proof
- “Present will”- mutually dependent for decision
- “Will”-VX (auxiliary verb) makes “Present” NN
- “Will”-NN (noun) makes “Present” JJ!
- Circular dependency

How to break the circularity?

- Maintain competing hypothesis
- Retain/discard with more information
- Where does more information come from?
- More look back-
- More look ahead- “Present will will replace any previous one”

More information? enter PROBABILITY!

- “will will” is MOST PROBABLY *not* a NN
NN combination
- Reduplication is very very rare in
English
 - Linguistics understanding
 - Frequentist experience
 - (linguistics + probability = Empirical NLP)

Lets remember

- Half the problem of POS tagging in English originate in the peculiar behaviour of the language that
 - *Almost all nouns are also verbs!*
- *Damage control by introducing ARTICLE*
- Whoever introduced article must have been a genius!
- 's' for BOTH 3rd person singular number present tense AND plural noun has perennially been a headache!

Article and 's'

- Article is of supreme importance in POS tagging
- Like 'capitalization' in English NER
- Ablate 'the', performance drops drastically, just like dropping capitalization in English NER
- Complementary, 's' has caused infinite grief to POS taggers!

The curious case of 'TO'

Language Phenomena

```
graph TD; A[Language Phenomena] --> B["To (infinitive, complementizer, Preposition)"]; A --> C["NNS & VBZ"];
```

To (infinitive, complementizer, Preposition)

1. *He wants to dance*
2. *To dance was his passion*
3. *I went to dance parties*

NNS & VBZ

1. Most English nouns can act as verbs
2. Noun plurals have the same form as 3p1n verbs

'to' as a preposition is kept separate from all other prepositions

Christopher D. Manning. 2011. Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?

In Alexander Gelbukh (ed.), *Computational Linguistics and Intelligent Text Processing, 12th International Conference, CICLing 2011, Proceedings, Part I*. Lecture Notes in Computer Science 6608, pp. 171--189.

NLP: compulsory Inter layer interaction (1/2)

Text-1: "I saw the boy with a telescope which he dropped accidentally"

Text-2: "I saw the boy with a telescope which I dropped accidentally"

Text-1:

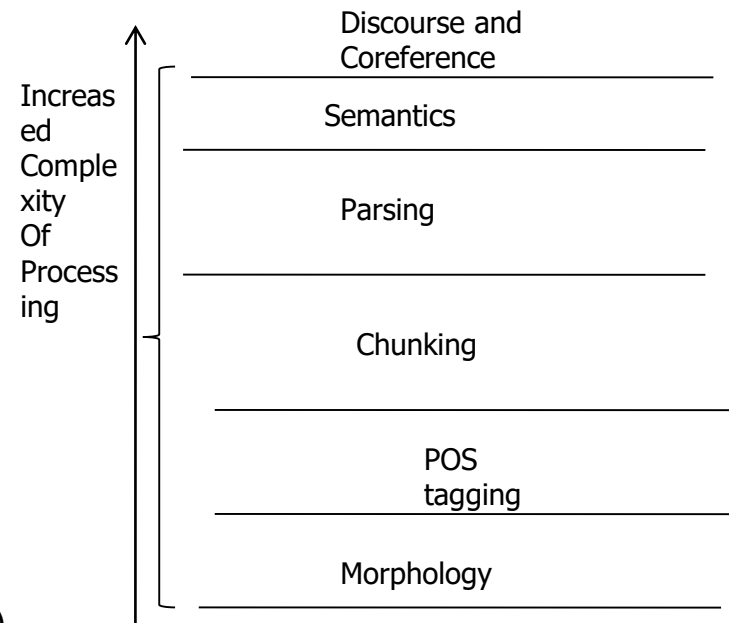
(S
 (NP (PRP I))
 (VP
 (VBD saw)
 (NP (DT the) (NN boy))
 (**PP (IN with)** (NP (NP (DT a) (NN telescope))
 (SBAR (WHNP (WDT which)) (S (NP (PRP I))
 (VP (VBD dropped)
 (ADVP (RB accidentally)))))) (. .)))

Text-2:

(S
 (NP (PRP I))
 (VP
 (VBD saw)
 (NP (DT the) (NN boy))
 (**PP (IN with)** (NP (NP (DT a) (NN telescope))
 (SBAR (WHNP (WDT which)) (S (NP (PRP he))
 (VP (VBD dropped) (ADVP (RB accidentally)))))) (. .)))

1 jun, 2017

acm:iitkgp:pos:pushpak



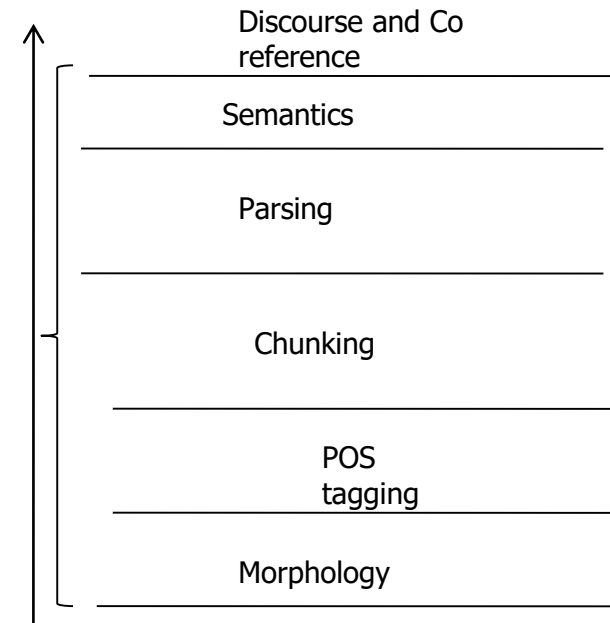
Inter layer interaction (2/2)

Text-1: *"I saw the boy with a telescope which he dropped accidentally"*

Text-2: *"I saw the boy with a telescope which I dropped accidentally"*

nsubj(saw-2, I-1)
root(ROOT-0, saw-2)
det(boy-4, the-3)
dobj(saw-2, boy-4)
det(telescope-7, a-6)
prep_with(saw-2, telescope-7)
dobj(dropped-10, telescope-7)
nsubj(dropped-10, I-9)
rcmod(telescope-7, dropped-10)
advmod(dropped-10, accidentally-11)

nsubj(saw-2, I-1)
root(ROOT-0, saw-2)
det(boy-4, the-3)
dobj(saw-2, boy-4)
det(telescope-7, a-6)
prep_with(saw-2, telescope-7)
dobj(dropped-10, telescope-7)
nsubj(dropped-10, he-9)
rcmod(telescope-7, dropped-10)
advmod(dropped-10, accidentally-11)



Morphology and POS tagging

- Finnish: "istahtaisinkohan"
- English: "I wonder if I should sit down for a while"

Analysis:

- ist + "sit", verb stem
- ahta + verb derivation morpheme, "to do something for a while"
- isi + conditional affix
- n + 1st person singular suffix
- ko + question particle
- han a particle for things like reminder (with declaratives) or "softening" (with questions and imperatives)

Need for ML in POS tagging

- New examples break rules, so we need a robust system.
- Machine learning based POS tagging:
 - HMM (Accuracy increased by 10-20% against rule based systems)
 - Jelinek's work

Important summary from language phenomena analysis

- POS tagging needs

1. Look back and Look ahead

'the' $\leftarrow \rightarrow$ 'present' $\leftarrow \rightarrow$ 'will'

2. Up and down transition across NLP Layers

'the present will will...' (pos \rightarrow chunk \rightarrow pos \rightarrow chunk \rightarrow ...)

What machine/technique can provide this facility?

ML technique repository (1/4)

■ Generative

- *Training phase*- Maximize the Likelihood of observations
- Thereby fix POS probabilities given the words
- *Testing phase*- Search in the hypothesis space of possible POS tag sequences
- In this case, generate POS tag *sequences* and score them
- E.g., Hidden Markov Model

ML technique repository (2/4)

■ Discriminative

- *Training phase*- Maximize entropy of probability distribution subject to the constraints imposed by data
- Thereby fix POS probabilities
- *Testing phase*- discriminate amongst hypotheses by scoring them
- In this case, discriminate amongst POS tag *sequences*
- E.g., MEMM, CRF

ML technique repository (3/4)

■ Classificatory

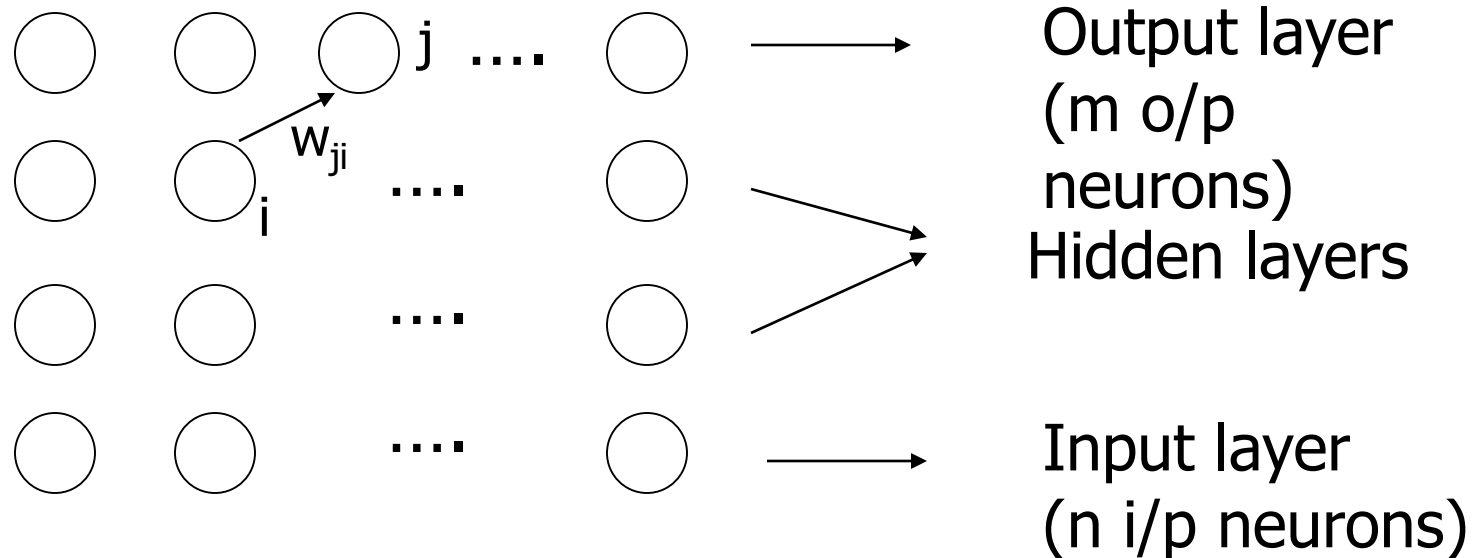
- *Training phase*- Minimise loss function (Total sum square error, cross entropy-soft_max)
- Thereby fix machine parameters
- *Testing phase*- classify the input INTO one of the hypotheses classes
- In this case, classify into one of the POS tag *sequences*
- E.g., SVM, Neural Net

ML technique repository (4/4)

■ Deep Neural Net

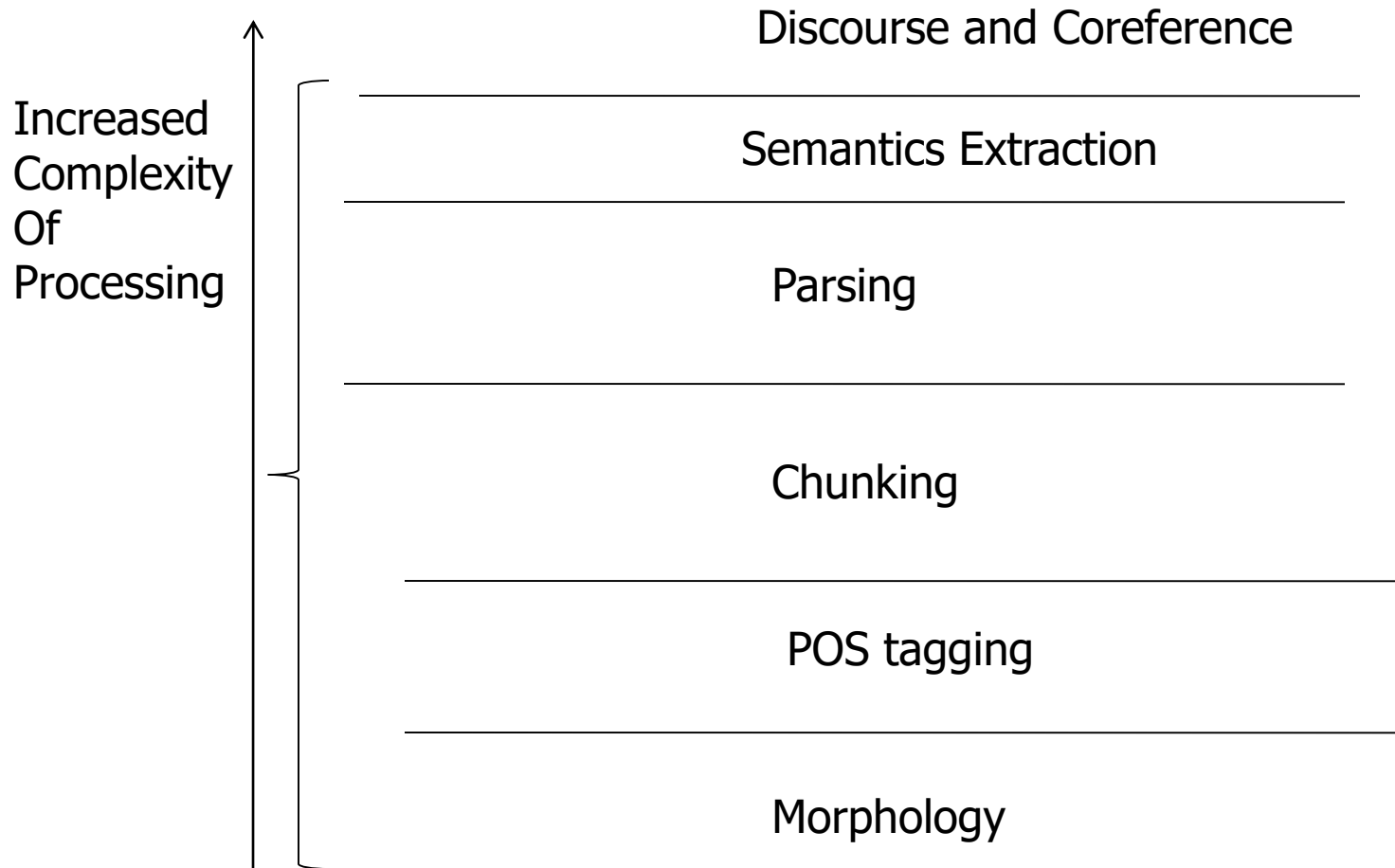
- *Training phase*- Like classificatory: Minimise loss function (Total sum square error, cross entropy-soft_max)
- But do feature discovery
- *Deep Learning= Feature Discovery+Classification*
- In this case, classify into one of the POS tag sequences, while discovering required features like morphological features on the way

Multilayer neural net

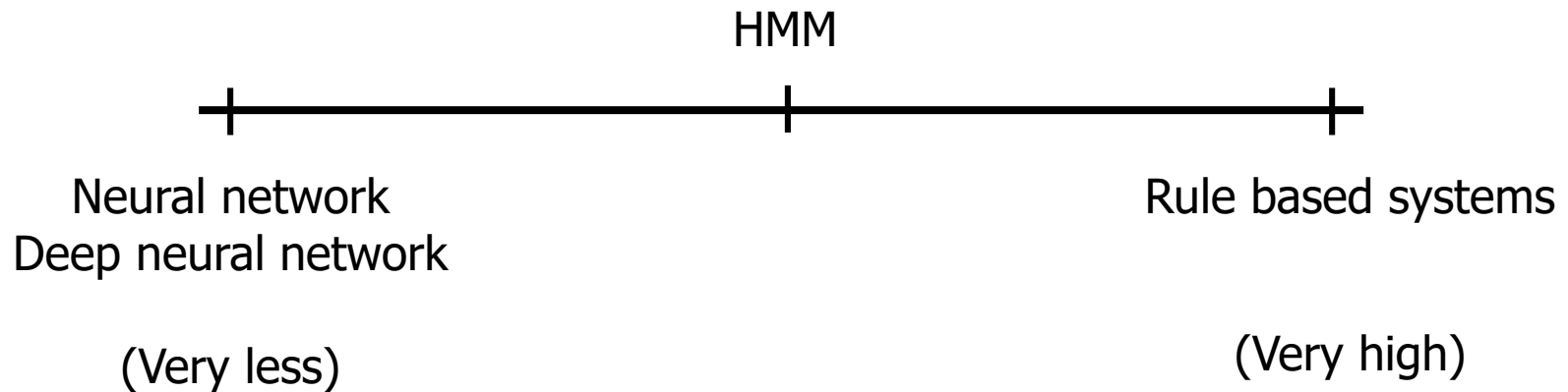


- NLP pipeline \leftrightarrow NN layers
- Discover bigger structures bottom up, starting from character?
- Words, POS, Parse, Sentence, Discourse?

NLP Layer/Pipeline



Understandability spectrum



Combinatorics of POS tagging

- Remember?- only ONE appearance sufficient
- More than one data point needed due to AMBIGUITY!
- Problem formulation

Let

- N: Number of words in vocabulary
- T: Number of tags
- A: Maximum degree of ambiguity of POS

Example scenario

- Given

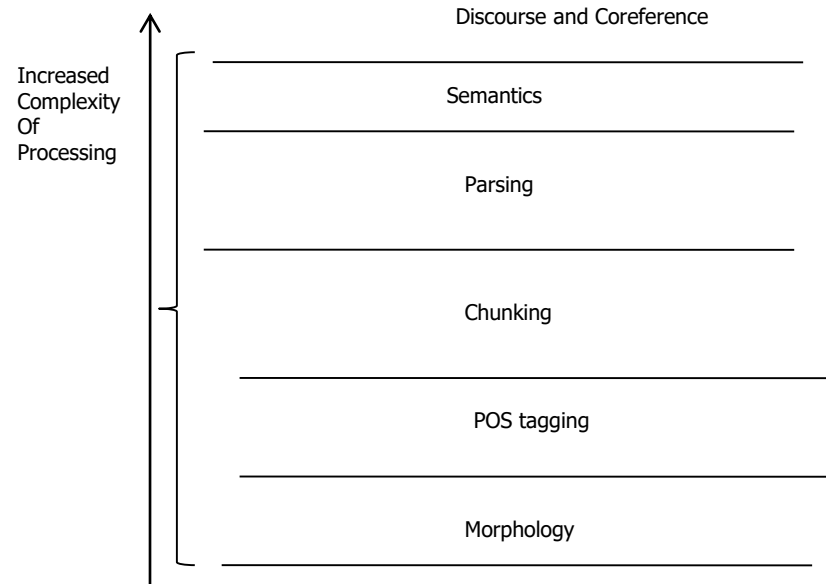
- $N = 2,00,000$ words in English
- $T = 40$ (Penn tag set)
- Maximum degree of ambiguity i.e. $A = 4$
- Average degree of ambiguity = 2
- Demanded accuracy = 90% (atleast)

- a) What is the maximum amount of data that we need?
- b) What is the average amount of data that we need?

Mathematics of POS tagging

Best Possible Tag Sequence

- Many factors are at play: morphology, syntax, semantics
- Computation in presence of uncertainty
- What is the best that can be done?



Best tag sequence
= T^*
= $\operatorname{argmax} P(T | W)$
= $\operatorname{argmax} P(T)P(W | T)$

Another example to emphasise the point

- DEM-PRON dilemma in Hindi POS tagging
- 'जो' can be both DEM and PRON
- जो_DEM बच्चे रोज स्कूल जाते हैं, वे अच्छे बच्चे कहलाते हैं
- जो_PRON रोज स्कूल जाते हैं, वे अच्छे बच्चे कहलाते हैं

Difficult to for rules!

- 'jo' followed by NN, makes it DEM
- Fails: जो_DEM स्कूल रोज जाते हैं, वे अच्छे बच्चे कहलाते हैं
- Succeeds: जो_PRON स्कूल रोज होम वर्क नहीं देते हैं, वे अच्छे स्कूल कहलाते हैं!
- Clue can be far apart!
- Clue can be before too!
- वे बच्चे जो_PRON स्कूल रोज जाते हैं, अच्छे कहलाते हैं

Argmax computation (1/2)

Best tag sequence

$$= T^*$$

$$= \operatorname{argmax} P(T|W)$$

$$= \operatorname{argmax} P(T)P(W|T) \quad (\text{by Baye's Theorem})$$

$$\begin{aligned} P(T) &= P(t_0 = \wedge t_1 t_2 \dots t_{n+1} = .) \\ &= P(t_0)P(t_1|t_0)P(t_2|t_1 t_0)P(t_3|t_2 t_1 t_0) \dots \\ &\quad P(t_n|t_{n-1} t_{n-2} \dots t_0)P(t_{n+1}|t_n t_{n-1} \dots t_0) \\ &= P(t_0)P(t_1|t_0)P(t_2|t_1) \dots P(t_n|t_{n-1})P(t_{n+1}|t_n) \end{aligned}$$

$$= \prod_{i=0}^{N+1} P(t_i|t_{i-1})$$

Bigram Assumption

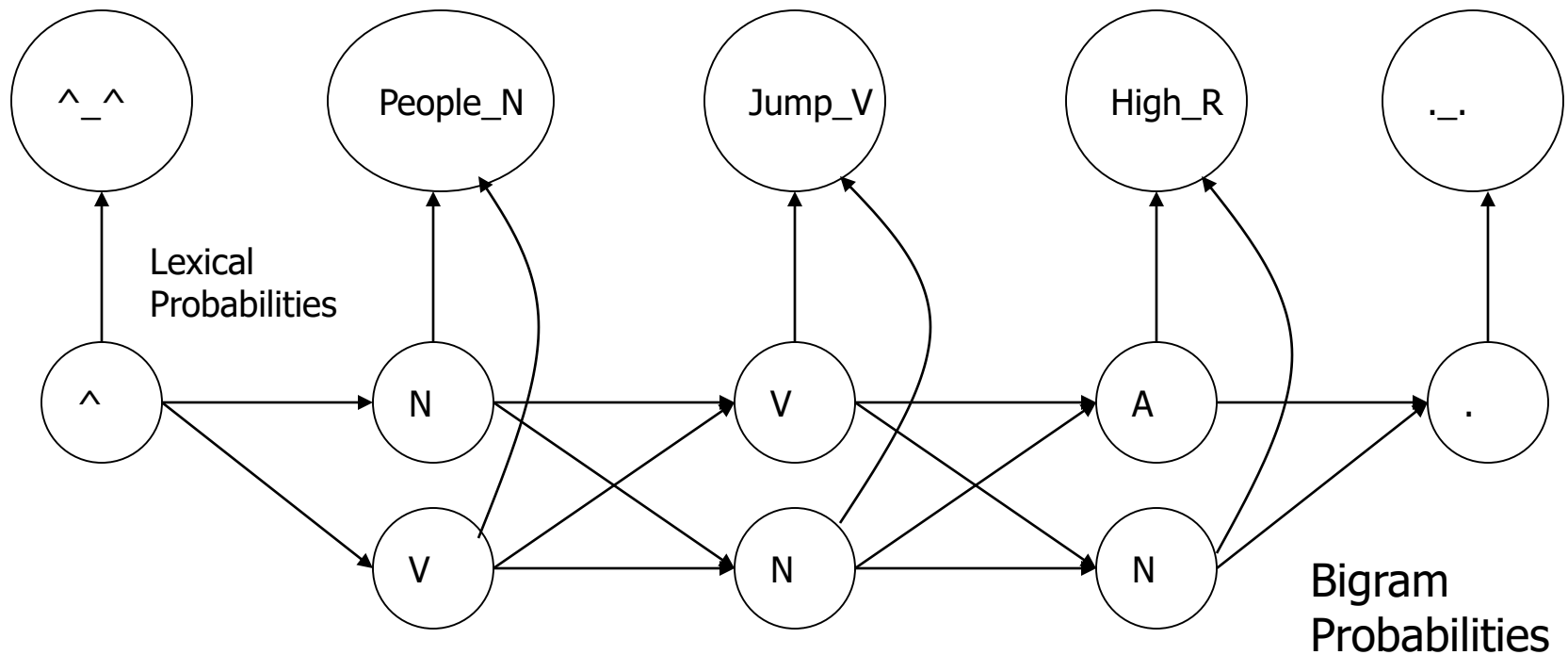
Argmax computation (2/2)

$$P(W|T) = P(w_0|t_0-t_{n+1})P(w_1|w_0t_0-t_{n+1})P(w_2|w_1w_0t_0-t_{n+1}) \dots \\ P(w_n|w_0-w_{n-1}t_0-t_{n+1})P(w_{n+1}|w_0-w_nt_0-t_{n+1})$$

Assumption: A word is determined completely by its tag. This is inspired by speech recognition

$$\begin{aligned} &= P(w_0|t_0)P(w_1|t_1) \dots P(w_{n+1}|t_{n+1}) \\ &= \prod_{i=0}^{n+1} P(w_i|t_i) \\ &= \prod_{i=1}^{n+1} P(w_i|t_i) \quad (\text{Lexical Probability Assumption}) \end{aligned}$$

Generative Model



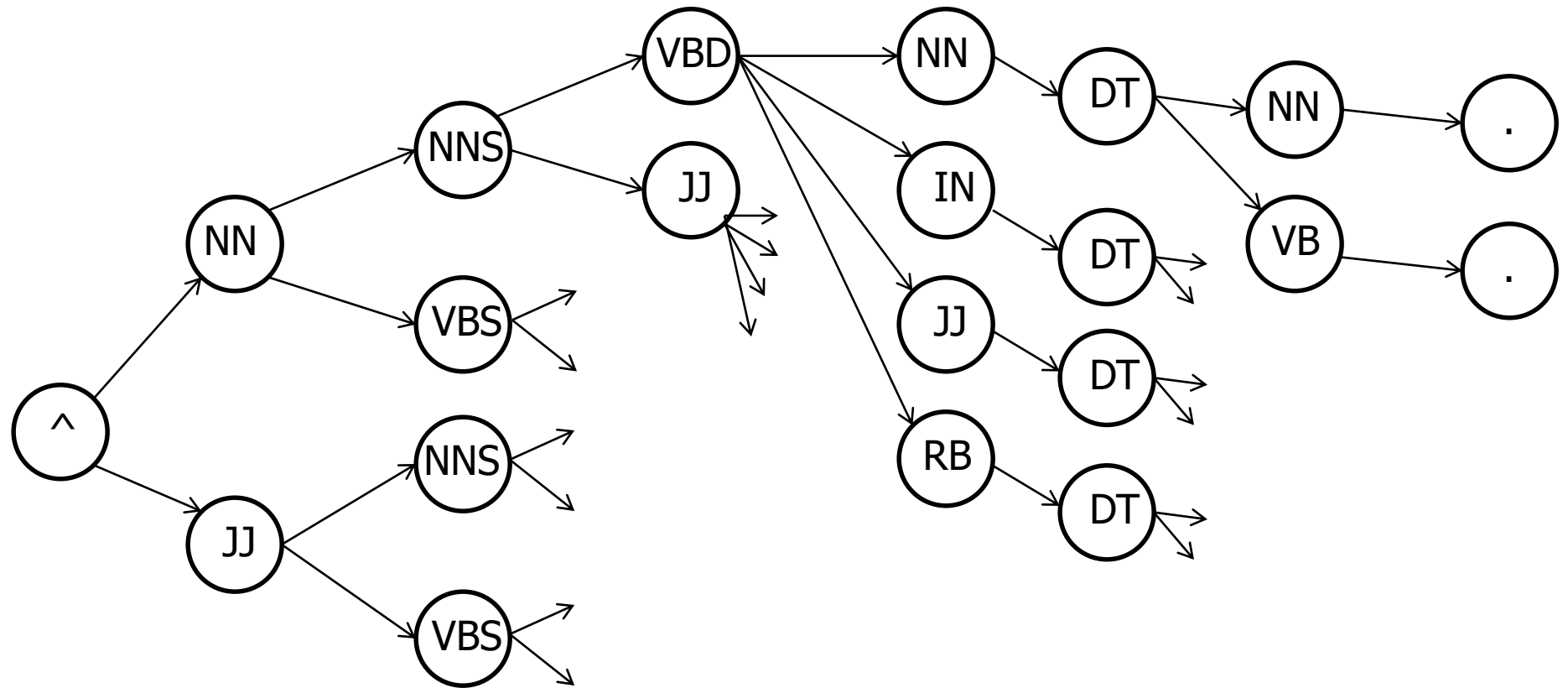
This model is called Generative model.
Here words are observed from tags as states.
This is similar to HMM.

Typical POS tag steps

- Implementation of Viterbi – Unigram, Bigram.
- Five Fold Evaluation.
- Per POS Accuracy.
- Confusion Matrix.

Computation of POS tags

W:	^	Brown	foxes	jumped	over	the	fence	.
T:	^	JJ	NNS	VBD	NN	DT	NN	.
		NN	VBS	JJ	IN		VB	
					JJ			
					RB			



^

Brown

1 jun, 2017

foxes

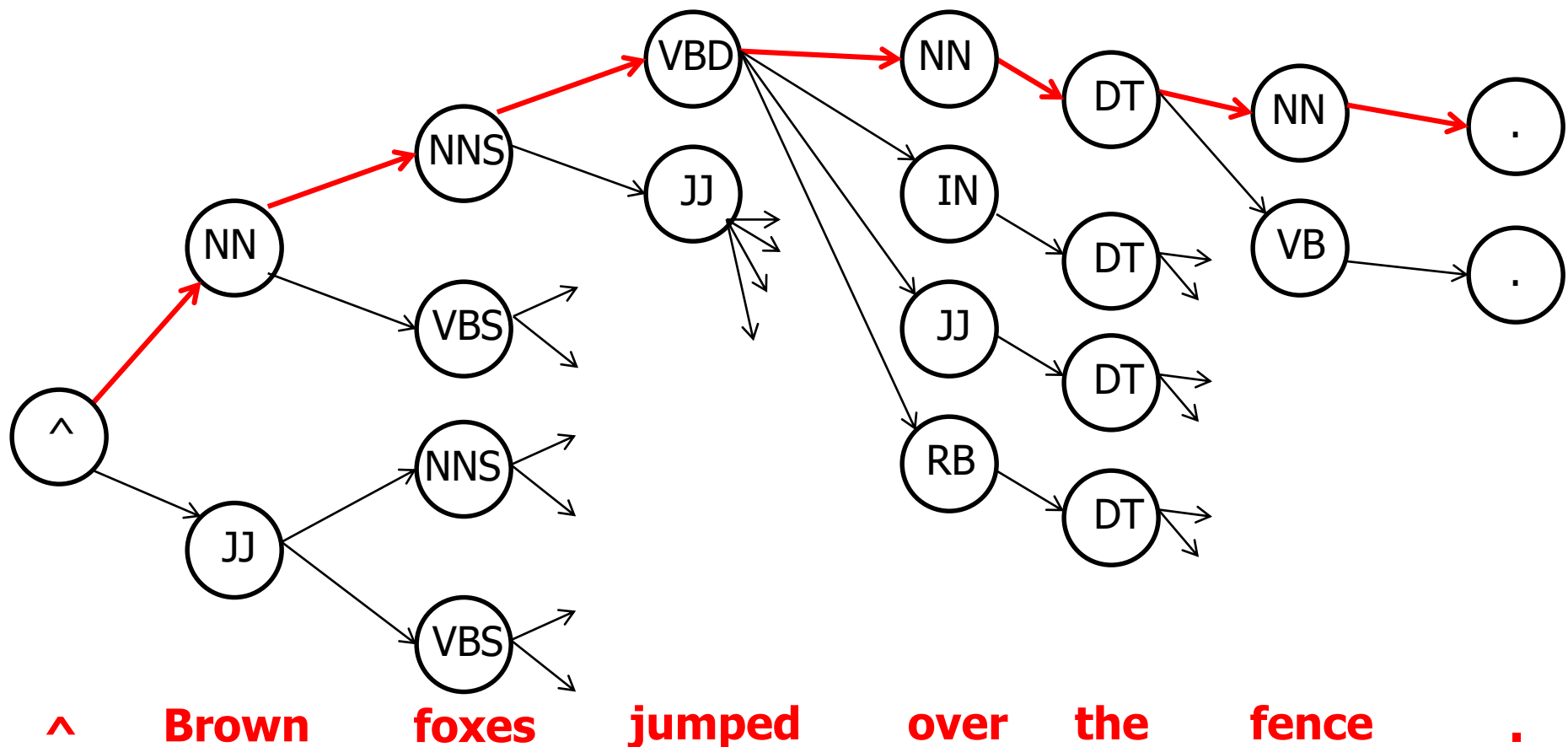
jumped

acm:iitkgp:pos:pushpak

over

the

fence



Probability of a path (e.g. Top most path) = $P(T) * P(W | T)$

$$P(\wedge) \cdot P(NN/\wedge) \cdot P(NNS/NN) \cdot P(VBD/NNS) \cdot P(NN/VBD) \cdot P(DT/NN) \cdot P(NN/DT) \cdot P(./NN) \cdot P(./) *$$

$$P(^/^) \cdot P(\text{brown}/NN) \cdot P(\text{foxes}/NNS) \cdot P(\text{jumped}/VBD) \cdot P(\text{over}/NN) \cdot P(\text{the}/DT) \cdot P(\text{fence}/NN) \cdot P(.|.)$$

Questions?

- Where do tags come from?
 - Tag set
- How to get probability values i.e. $P()$?
 - Annotated corpora

After modeling of the problem,
emphasis should be on the corpus.

Computing P() values

Lets suppose annotated corpus has following sentence

I	have	a	brown	bag	.
PRN	VB	DT	JJ	NN	.

$$P(NN | JJ) = \frac{\text{Number_of_times_JJ_followed_by_NN}}{\text{Number_of_times_JJ_appeared}}$$

$$P(\text{Brown} | JJ) = \frac{\text{Number_of_times_Brown_tagged_as_JJ}}{\text{Number_of_times_JJ_appeared}}$$

Next question?

- How to decode efficiently?

- E.g.

- T: Tags
- W: Words
- Two special symbol: '^' and '.'

Find out number of paths in the tree given word sequence.

Exponential *w.r.t.* number of words

Number of path = Number of leaves in the tree.

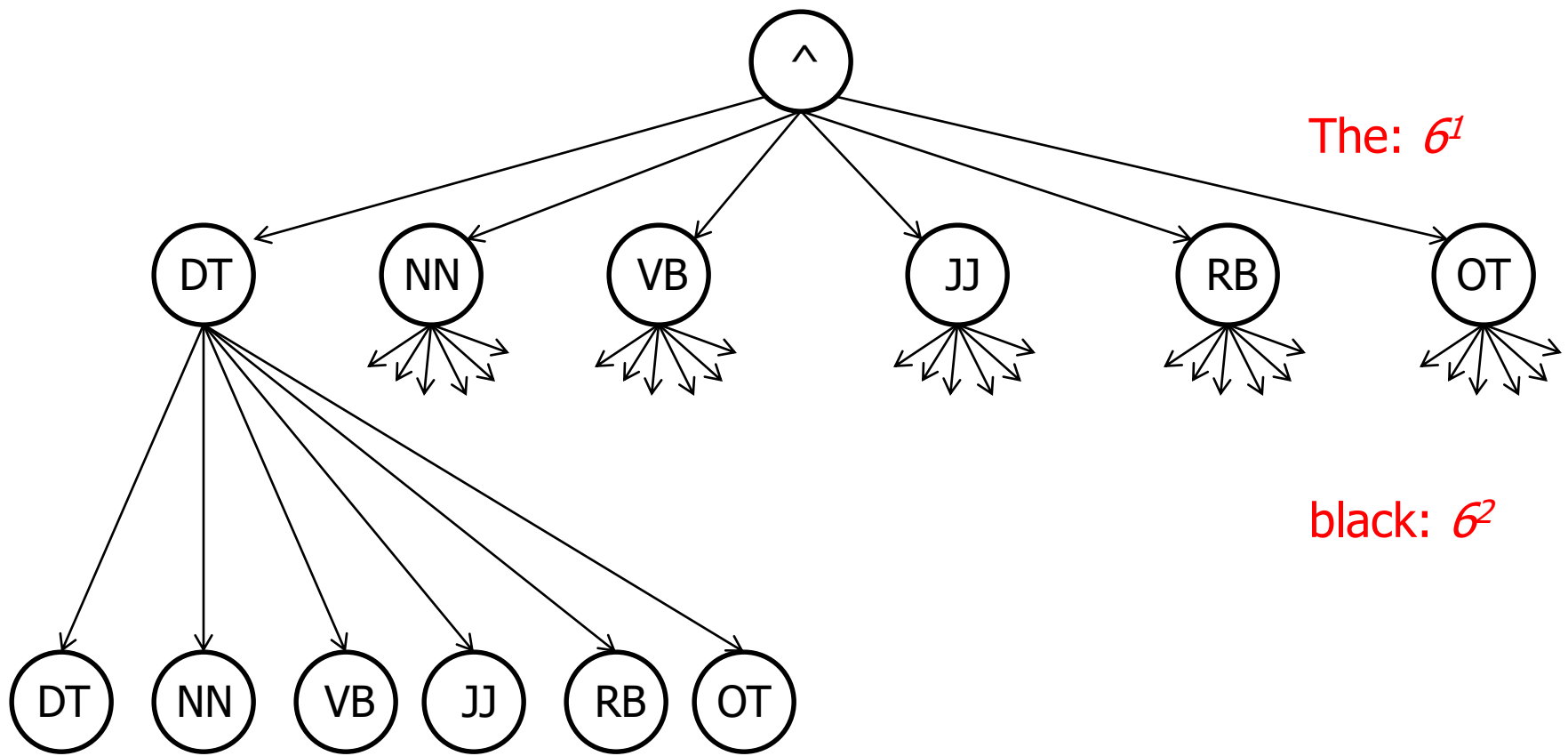
$$O(T^n)$$

We do not need exponential work!

- Suppose our tags are
 - DT, NN, VB, JJ, RB and OT
- E.g.

^	The	black	dog	barks	.
^	DT	DT	DT	DT	.
	NN	NN	NN	NN	.
	VB	VB	VB	VB	.
	JJ	JJ	JJ	JJ	.
	RB	RB	RB	RB	.
	OT	OT	OT	OT	.

→ Possible tags



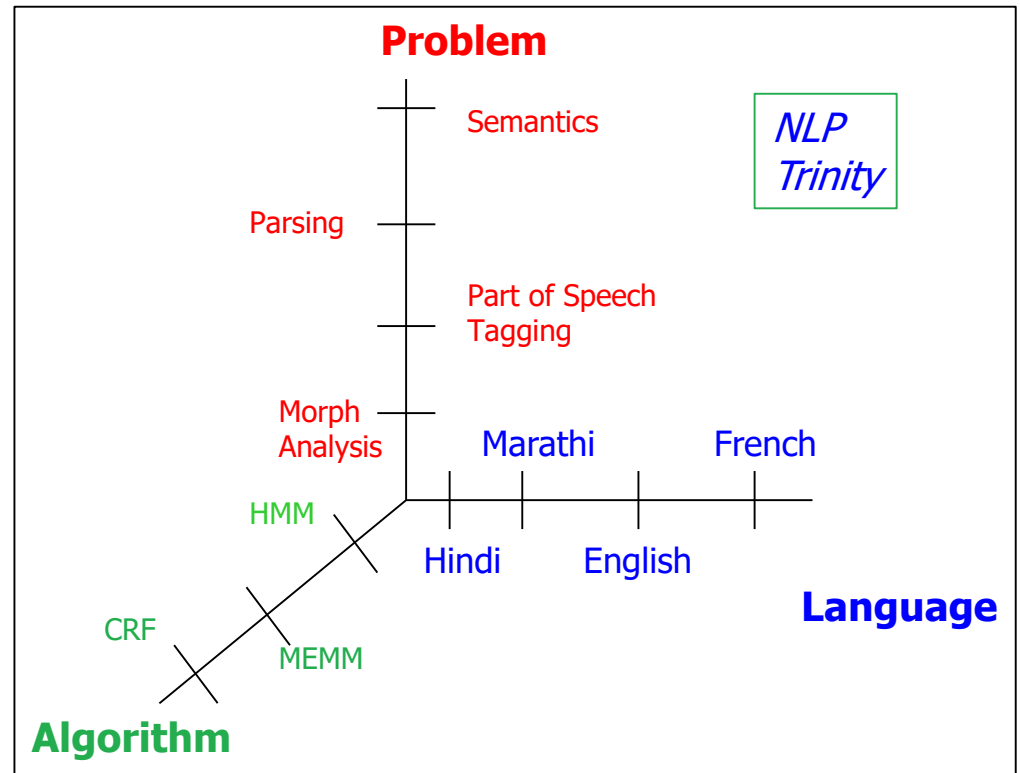
Total 6^4 paths

- Now consider the paths that end in NN after seeing input “The black”

^	Th e	blac k	
^	DT	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{DT} \wedge) \cdot P(\text{NN} \text{DT}) \cdot P(\text{The} \text{DT}) \cdot P(\text{Black} \text{NN})$
^	NN	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{NN} \wedge) \cdot P(\text{NN} \text{NN}) \cdot P(\text{The} \text{NN}) \cdot P(\text{Black} \text{NN})$
^	VB	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{VB} \wedge) \cdot P(\text{NN} \text{VB}) \cdot P(\text{The} \text{VB}) \cdot P(\text{Black} \text{NN})$
^	JJ	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{JJ} \wedge) \cdot P(\text{NN} \text{JJ}) \cdot P(\text{The} \text{JJ}) \cdot P(\text{Black} \text{NN})$
^	RB	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{RB} \wedge) \cdot P(\text{NN} \text{RB}) \cdot P(\text{The} \text{RB}) \cdot P(\text{Black} \text{NN})$
^	OT	NN	$P(\mathbf{T}).P(\mathbf{W} \mathbf{T}) = P(\text{OT} \wedge) \cdot P(\text{NN} \text{OT}) \cdot P(\text{The} \text{OT}) \cdot P(\text{Black} \text{NN})$

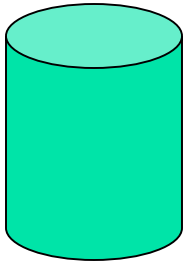
Complexity = $W_n * T$ For each tag, only path with highest probability value are **retained**, others are simply **discarded**.

HMM



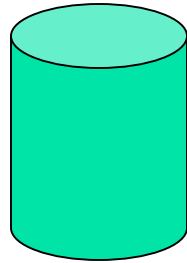
A Motivating Example

Colored Ball choosing



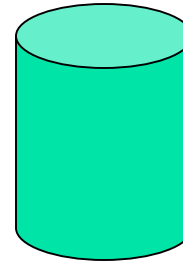
Urn 1

of Red = 30
of Green = 50
of Blue = 20



Urn 2

of Red = 10
of Green = 40
of Blue = 50



Urn 3

of Red = 60
of Green = 10
of Blue = 30

Example (contd.)

Given :

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Transition probability table

and

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

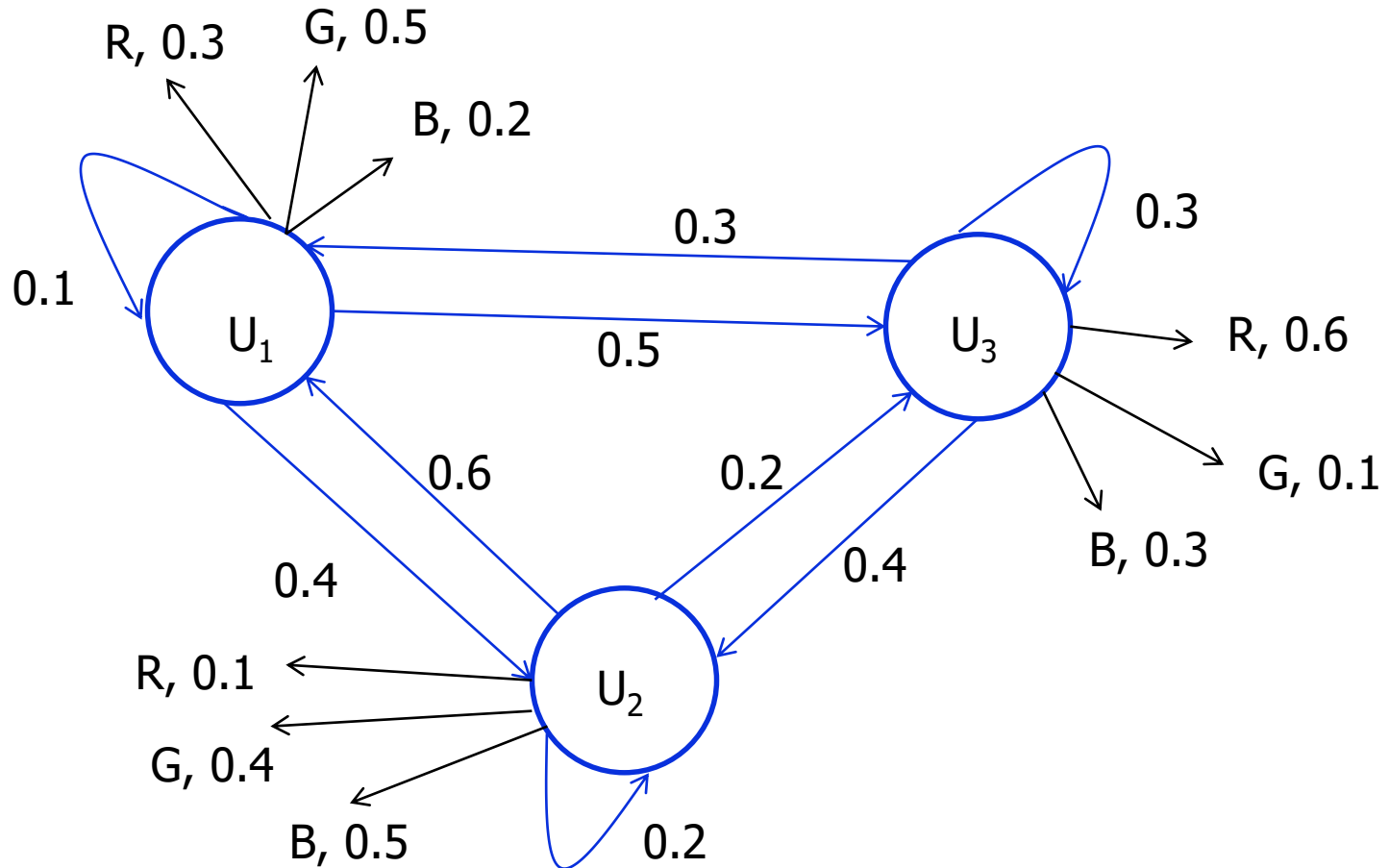
Emission probability table

Observation : RRGGBRGR

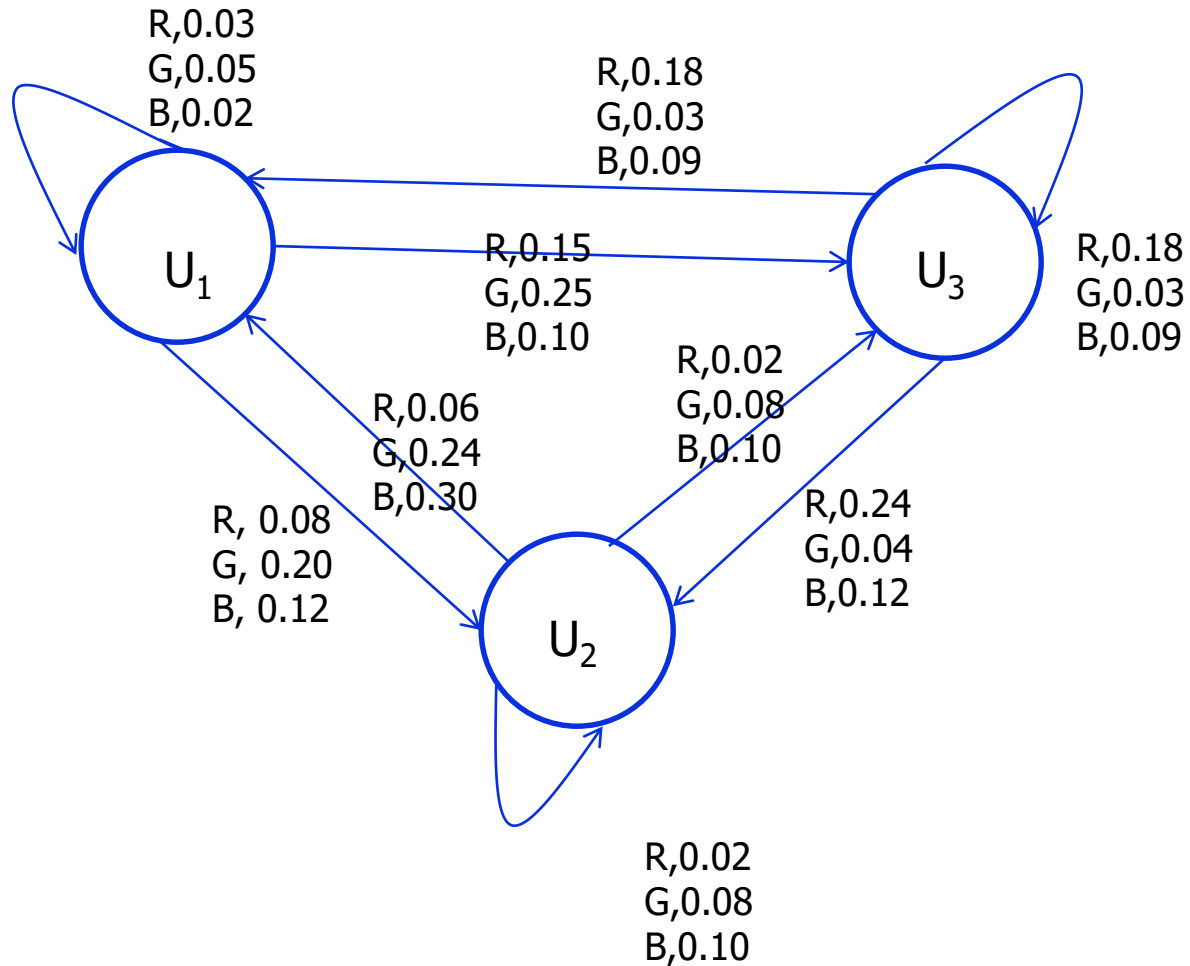
State Sequence : ??

Not so Easily Computable.

Diagrammatic representation (1/2)



Diagrammatic representation (2/2)



Classic problems with respect to HMM

1. Given the observation sequence, find the possible state sequences- Viterbi
2. Given the observation sequence, find its probability- forward/backward algorithm
3. Given the observation sequence find the HMM parameters.- Baum-Welch algorithm

Baye's Theorem

$$P(A | B) = P(A).P(B | A) / P(B)$$

$P(A)$ -: Prior

$P(B|A)$ -: Likelihood

$$\arg \max_s P(S | O) = \arg \max_s P(S).P(O | S)$$

State Transitions Probability

$$P(S) = P(S_{1-8})$$

$$P(S) = P(S_1).P(S_2 | S_1).P(S_3 | S_{1-2}).P(S_4 | S_{1-3})...P(S_8 | S_{1-7})$$

By Markov Assumption (k=1)

$$P(S) = P(S_1).P(S_2 | S_1).P(S_3 | S_2).P(S_4 | S_3)...P(S_8 | S_7)$$

Observation Sequence probability

$$P(O | S) = P(O_1 | S_{1-8}).P(O_2 | O_1, S_{1-8}).P(O_3 | O_{1-2}, S_{1-8})...P(O_8 | O_{1-7}, S_{1-8})$$

Assumption that ball drawn depends only on the Urn chosen

$$P(O | S) = P(O_1 | S_1).P(O_2 | S_2).P(O_3 | S_3)...P(O_8 | S_8)$$

$$P(S | O) = P(S).P(O | S)$$

$$P(S | O) = P(S_1).P(S_2 | S_1).P(S_3 | S_2).P(S_4 | S_3)...P(S_8 | S_7).$$

$$P(O_1 | S_1).P(O_2 | S_2).P(O_3 | S_3)...P(O_8 | S_8)$$

Grouping terms

	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ϵ	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9

$$\begin{aligned}
 &P(S).P(O|S) \\
 = &[P(O_0|S_0).P(S_1|S_0)]. \\
 &[P(O_1|S_1).P(S_2|S_1)]. \\
 &[P(O_2|S_2).P(S_3|S_2)]. \\
 &[P(O_3|S_3).P(S_4|S_3)]. \\
 &[P(O_4|S_4).P(S_5|S_4)]. \\
 &[P(O_5|S_5).P(S_6|S_5)]. \\
 &[P(O_6|S_6).P(S_7|S_6)]. \\
 &[P(O_7|S_7).P(S_8|S_7)]. \\
 &[P(O_8|S_8).P(S_9|S_8)].
 \end{aligned}$$

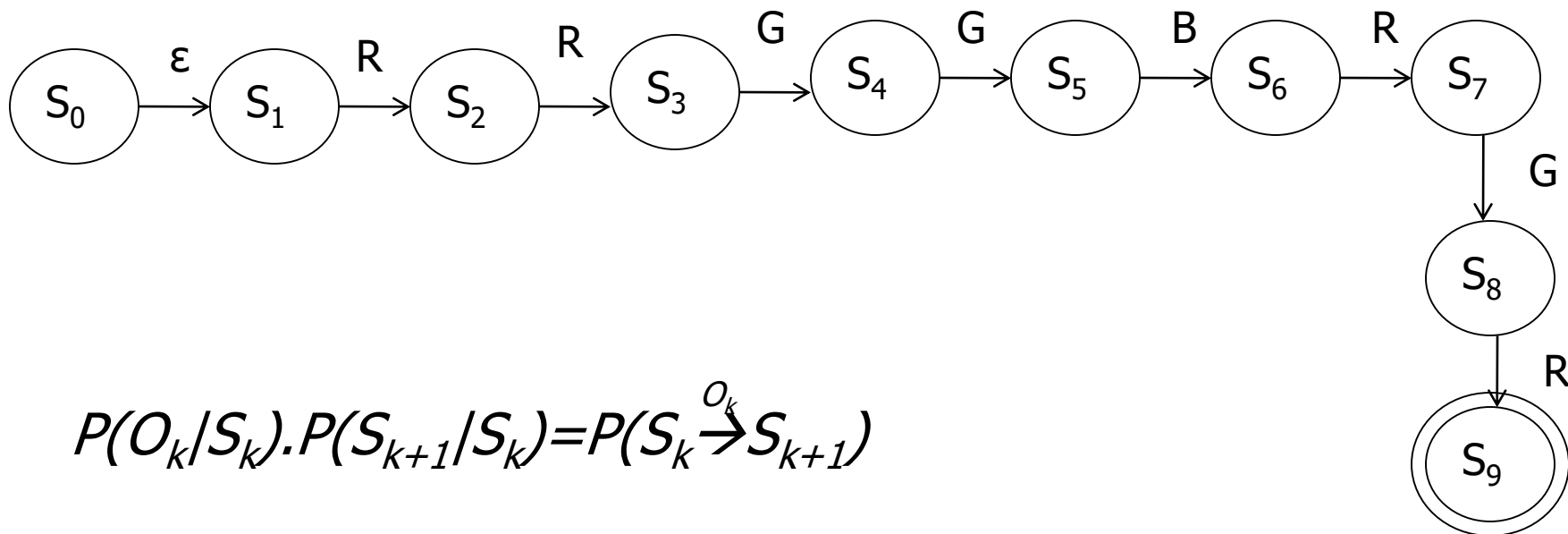
We introduce the states S_0 and S_9 as initial and final states respectively.

After S_8 the next state is S_9 with probability 1, i.e., $P(S_9|S_8)=1$

O_0 is ϵ -transition

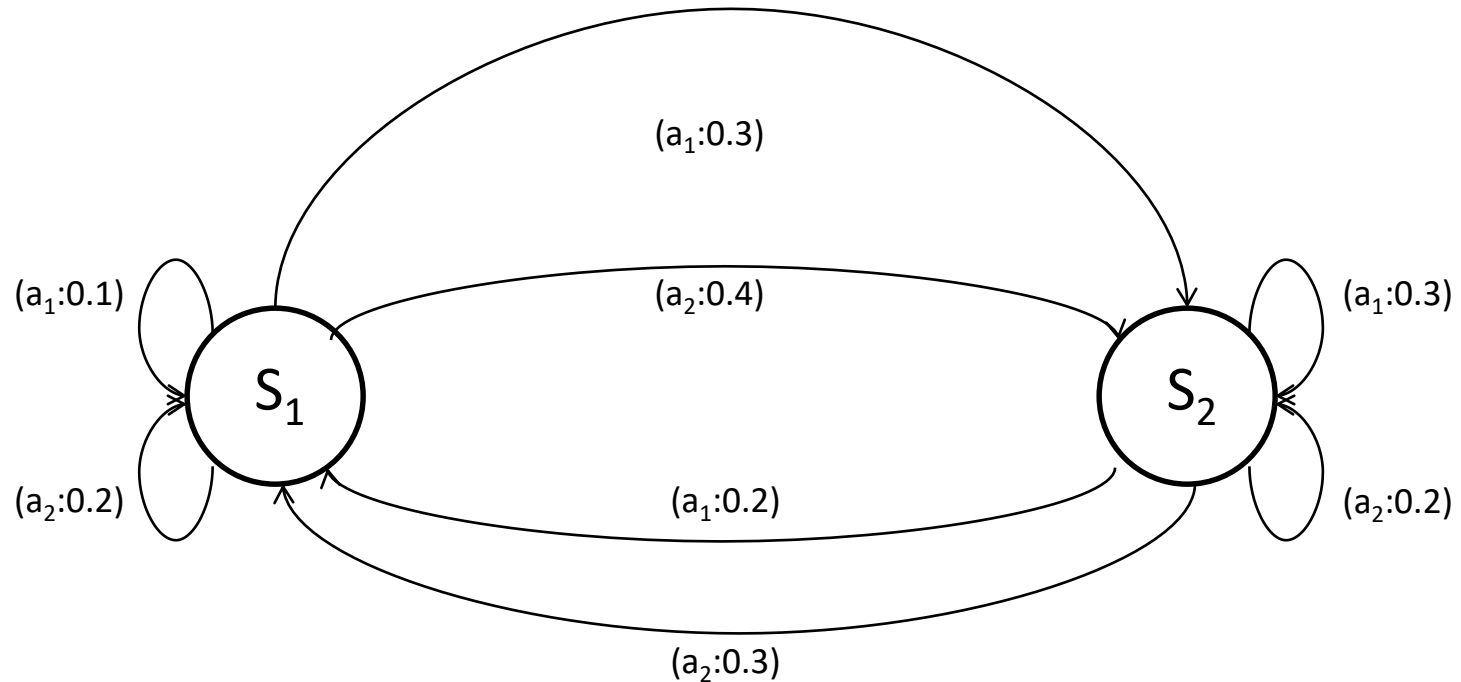
Introducing useful notation

	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ϵ	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9



$$P(O_k/S_k) \cdot P(S_{k+1}/S_k) = P(S_k \xrightarrow{O_k} S_{k+1})$$

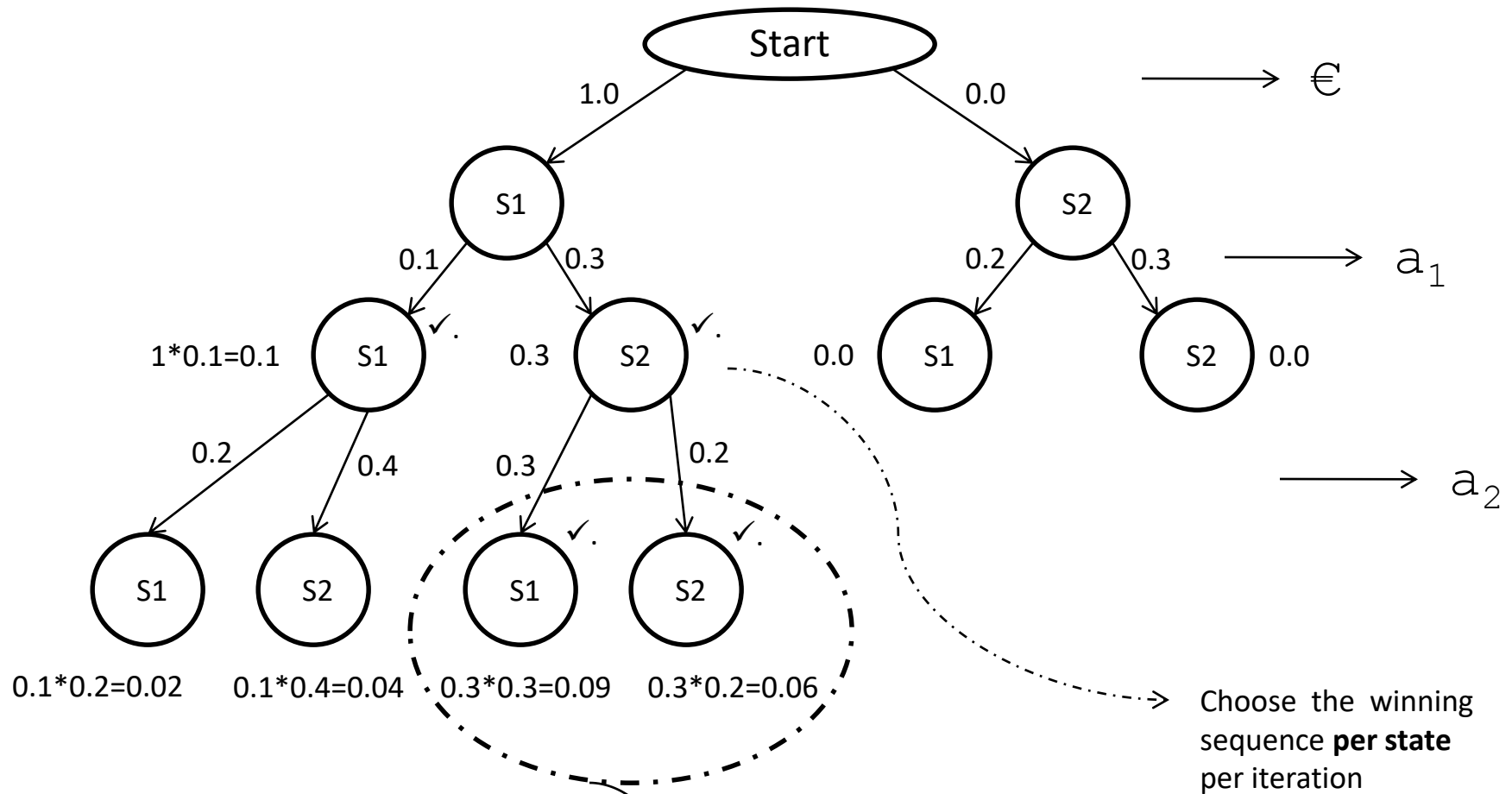
Probabilistic FSM



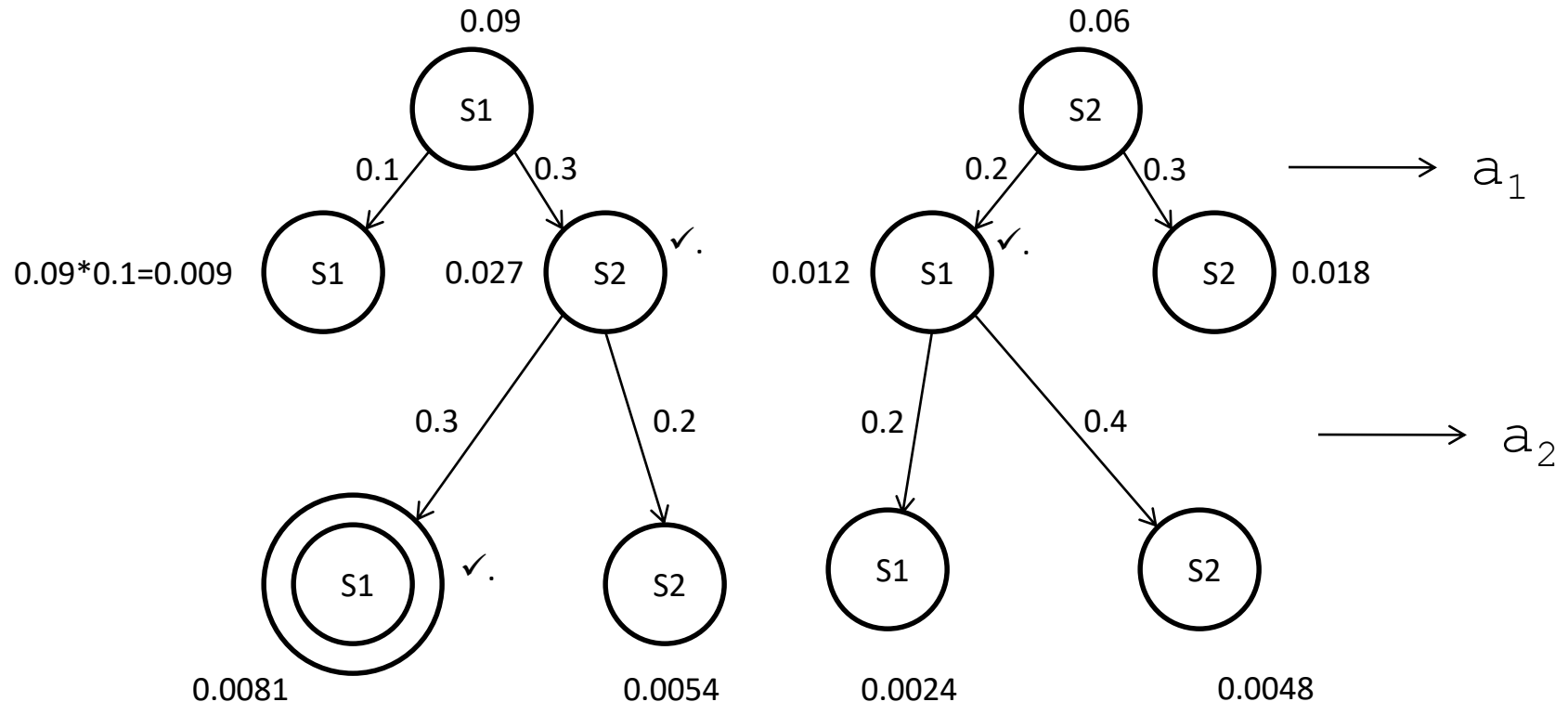
The question here is:

“what is the most likely state sequence given the output sequence seen”

Developing the tree

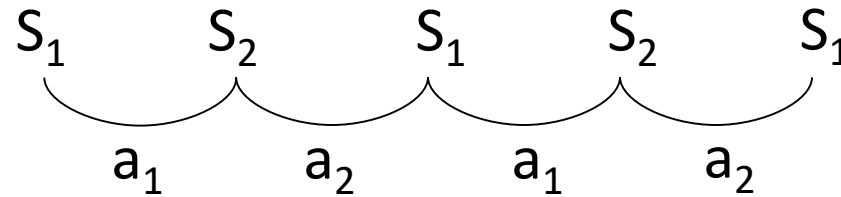


Tree structure contd...



The problem being addressed by this tree is $S^* = \arg \max_s P(S \mid a_1 - a_2 - a_1 - a_2, \mu)$
 $a_1 - a_2 - a_1 - a_2$ is the output sequence and μ the model or the machine

Path found:
(working backward)

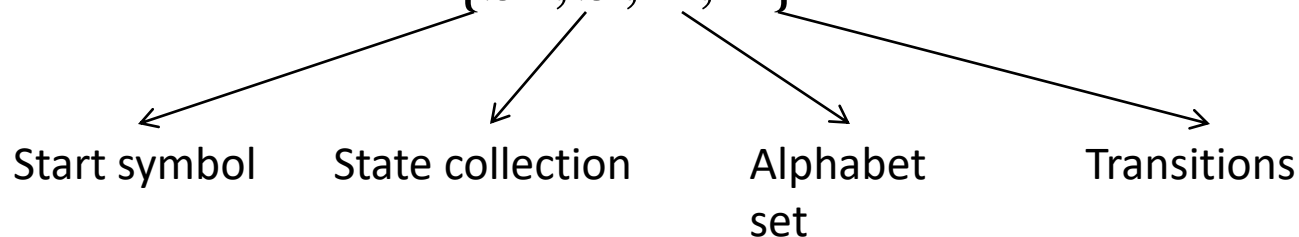


Problem statement: Find the best possible sequence

$$S^* = \arg \max_s P(S \mid O, \mu)$$

where, $S \rightarrow$ State Seq, $O \rightarrow$ Output Seq, $\mu \rightarrow$ Model or Machine

Model or Machine = $\{S_0, S, A, T\}$



T is defined as $P(S_i \xrightarrow{a_k} S_j) \quad \forall i, j, k$

POS tagging with CRF

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{\Psi_A \in G} \exp \left\{ \sum_{k=1}^{K(A)} \lambda_{Ak} f_{Ak}(\mathbf{y}_A, \mathbf{x}_A) \right\} .$$

Y: tag sequence

X: word sequence

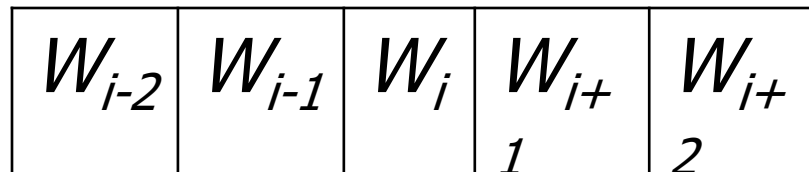
Accuracy ranging from 70s to 90s reported for many languages

Neural Net implementation

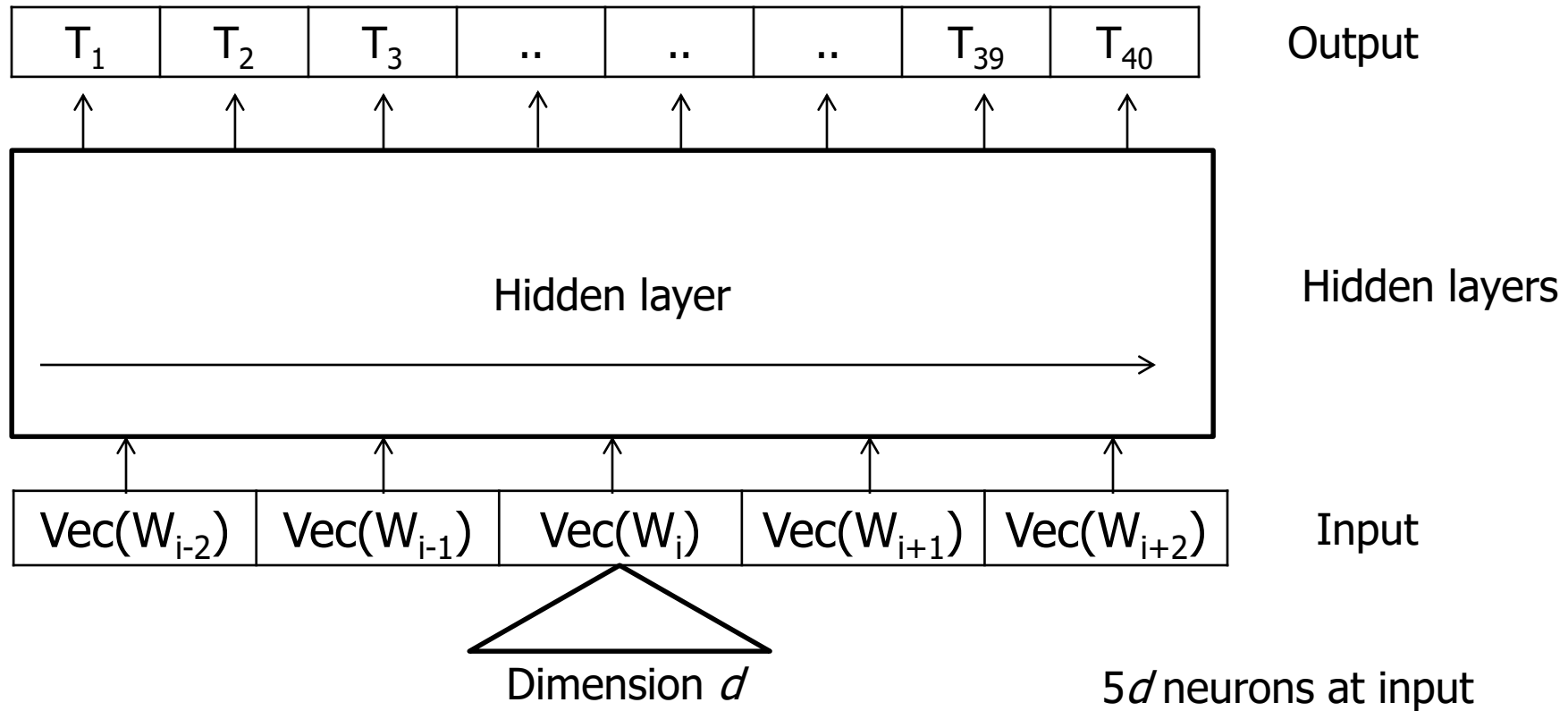
- Design and implement a neural network for POS tagging.

Input Sequence: $W_1 \quad W_2 \quad W_3 \quad .. \quad .. \quad W_n$
Output sequence: $T_1 \quad T_2 \quad T_3 \quad .. \quad .. \quad T_n$

- Context window of size 2 for word W_i



NN-POS tagger (contd)



POS tagging accuracy on WSJ corpus

Sys	Acc (%)
(Toutanova et al., 2003)	97.24
(Huang et al., 2012)	97.35
(Collobert et al., 2011) NN	96.36
(Collobert et al., 2011) NN+WE	97.20
BLSTM-RNN	96.61
BLSTM-RNN+WE(10m)	96.61
BLSTM-RNN+WE(100m)	97.10
BLSTM-RNN+WE(all)	97.26
BLSTM-RNN+WE(all)+suffix2	97.40

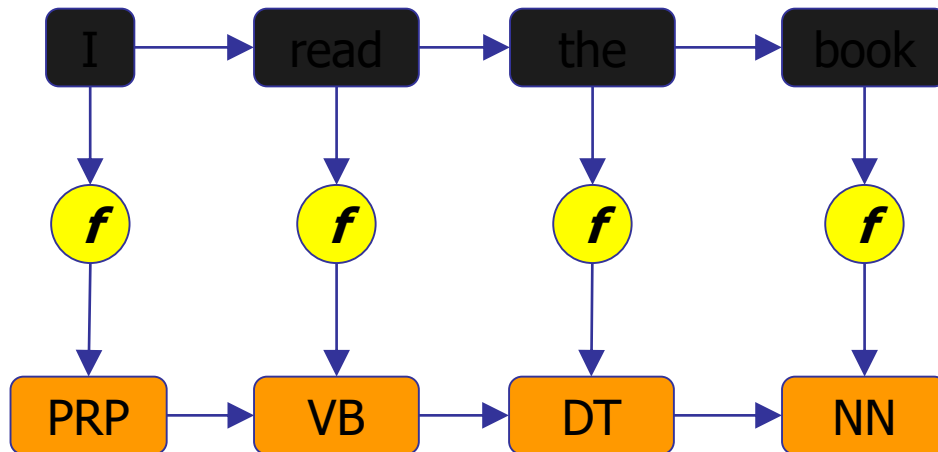
Peilu Wang, Yao Qian, Frank K. Soong, Lei He, Hai Zhao, Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network, Arxiv, Oct 2015

Deep Learning Based Seq2Seq Models and POS Tagging

Acknowledgement: Anoop Kunchukuttan, PhD Scholar, IIT Bombay

*So far we are seen POS tagging as a **sequence labelling** task*

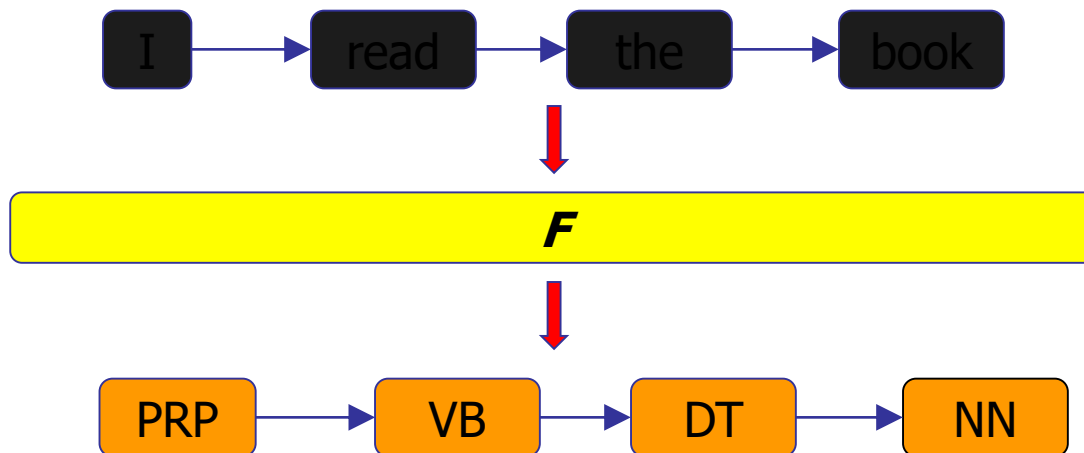
For every element, predict the tag/label (using function f)



- Length of output sequence is same as input sequence
- Prediction of tag at time t can use only the words seen till time t

We can also look at POS tagging as a *sequence to sequence transformation problem*

Read the entire sequence and predict the output sequence (using function F)



- Length of output sequence need not be the same as input sequence
- Prediction at any time step t has access to the entire input
- A more general framework than sequence labelling

Sequence to Sequence transformation is a more general framework than sequence labelling

- Many other problems can be expressed as sequence to sequence transformation
 - *e.g. machine translation, summarization, question answering, dialog*
- Adds more capabilities which can be useful for problems like MT:
 - many \rightarrow many mappings: insertion/deletion to words, one-one mappings
 - non-monotone mappings: reordering of words
- For POS tagging, these capabilities are not required

How does a sequence to sequence model work? Let's see two paradigms

Encode - Decode Paradigm

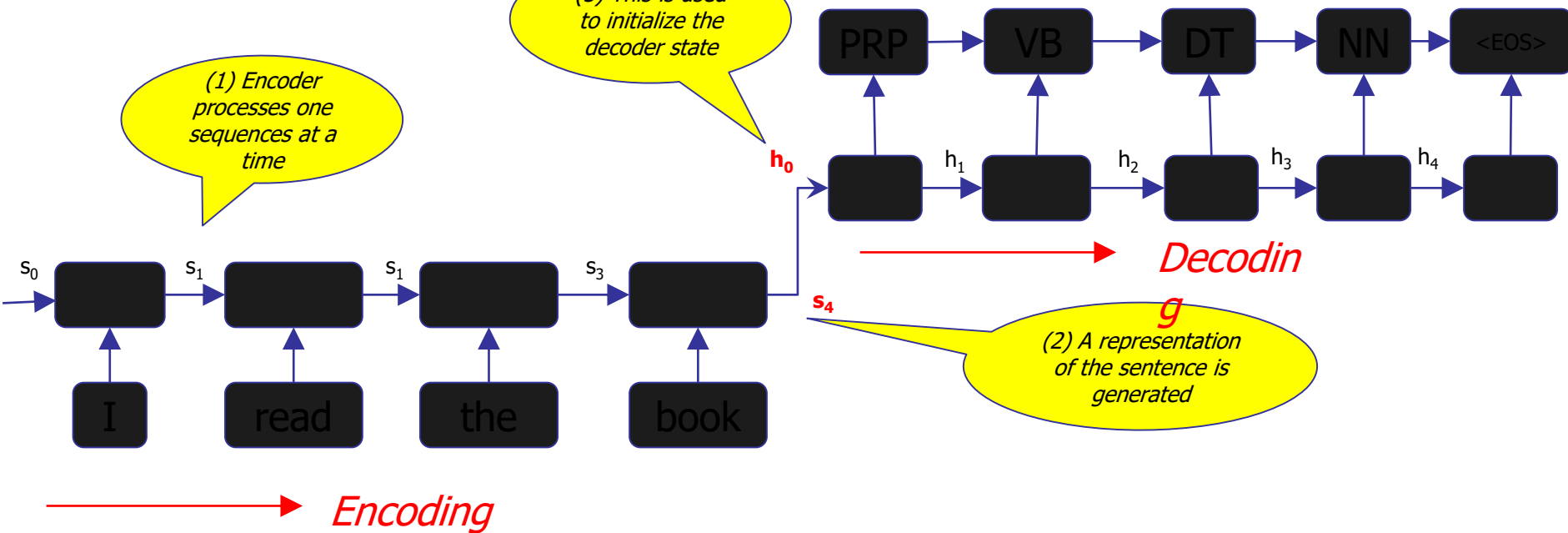
Use two RNN networks: the encoder and the decoder

(1) Encoder processes one sequence at a time

(3) This is used to initialize the decoder state

(4) Decoder generates one element at a time

(5)... continue till end of sequence tag is generated



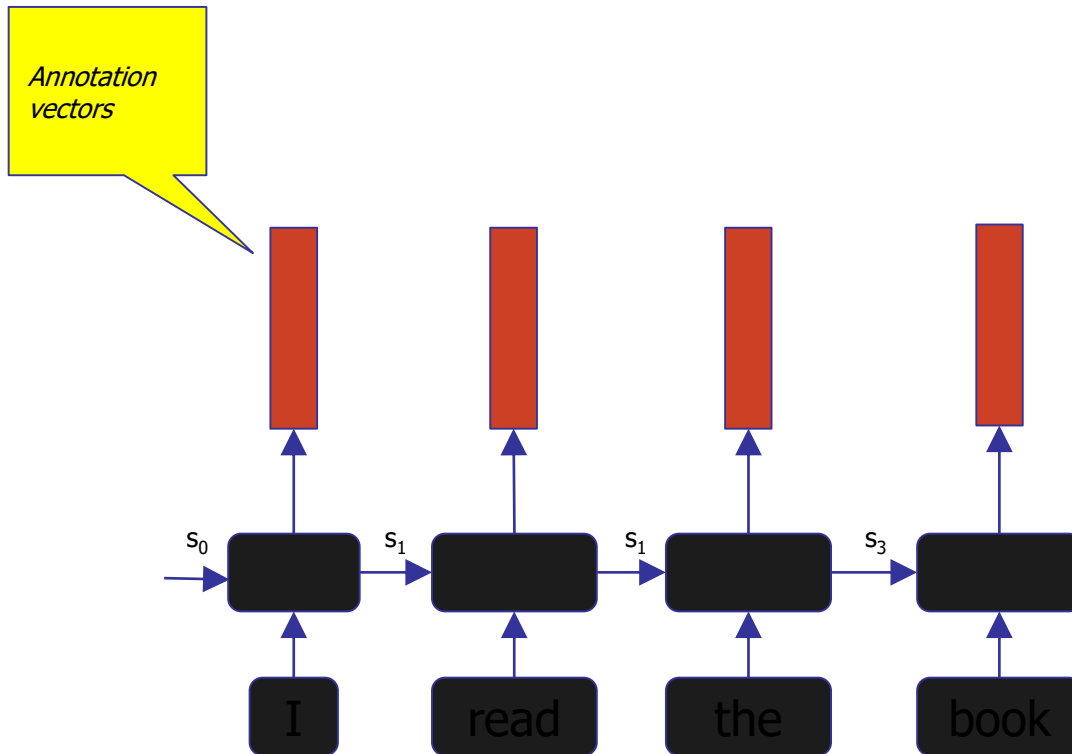
This approach reduces the entire sentence representation to a single vector

Two problems with this design choice:

- This is not sufficient to represent to capture all the syntactic and semantic complexities of a sentence
 - *Solution: Use a richer representation for the sentences*
- Problem of capturing long term dependencies: The decoder RNN will not be able to make use of source sentence representation after a few time steps
 - *Solution: Make source sentence information when making the next prediction*
 - *Even better, make **RELEVANT** source sentence information available*

These solutions motivate the next paradigm

Encode - Attend - Decode Paradigm



Represent the source sentence by the **set of output vectors** from the encoder

Each output vector at time t is a contextual representation of the input at time t

s_4 *Note: in the encoder-decode paradigm, we ignore the encoder outputs*

Let's call these encoder output vectors **annotation vectors**

How should the decoder use the set of annotation vectors while predicting the next character?

Key Insight:

- (1) Not all annotation vectors are equally important for prediction of the next element
- (2) The annotation vector to use next depends on what has been generated so far by the decoder

eg. To generate the 3rd POS tag, the 3rd annotation vector (hence 3rd word) is most important

One way to achieve this:

Take a weighted average of the annotation vectors, with more weight to annotation vectors which need more **focus or attention**

This averaged **context vector** is an input to the decoder

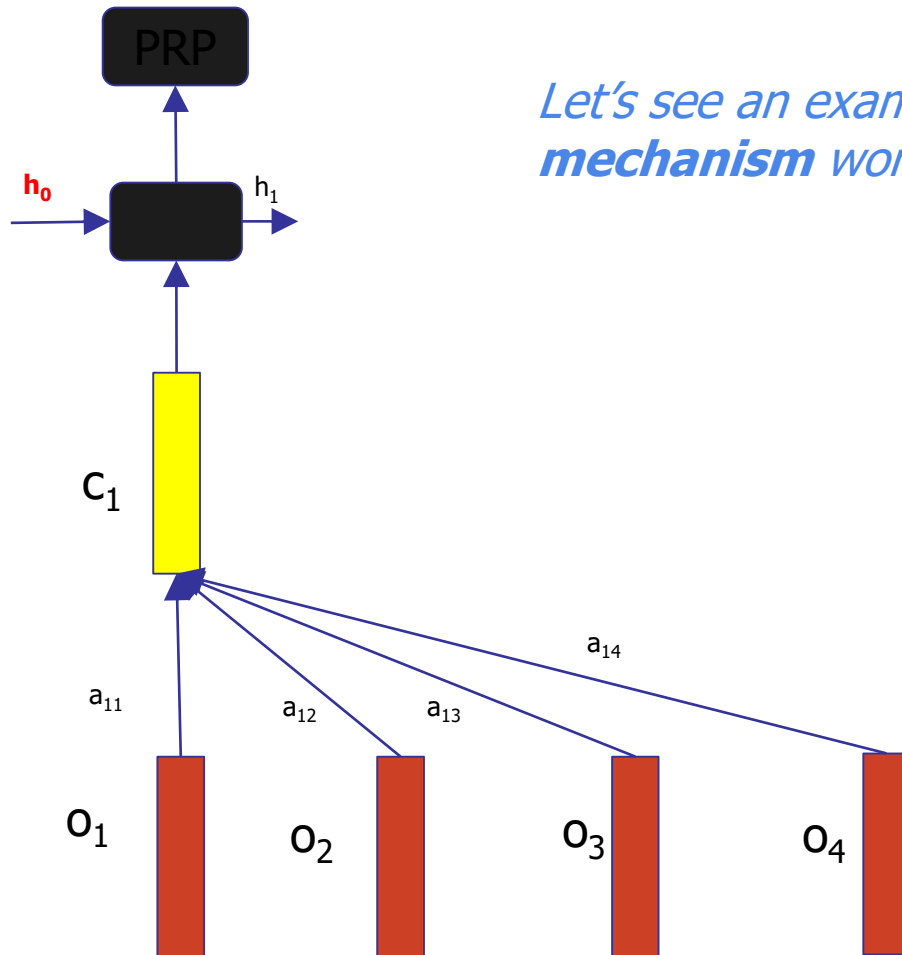
$$c_i = \sum_{j=1}^n a_{ij} o_j$$

For generation of i^{th} output character:

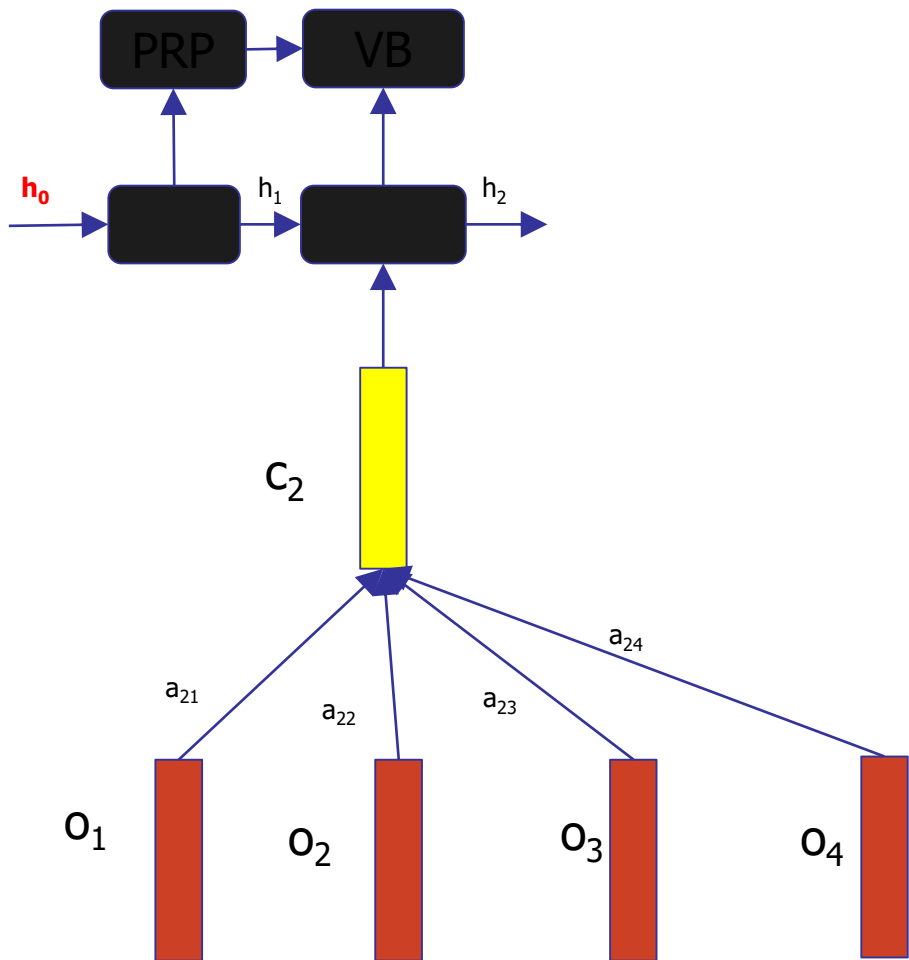
c_i : context vector

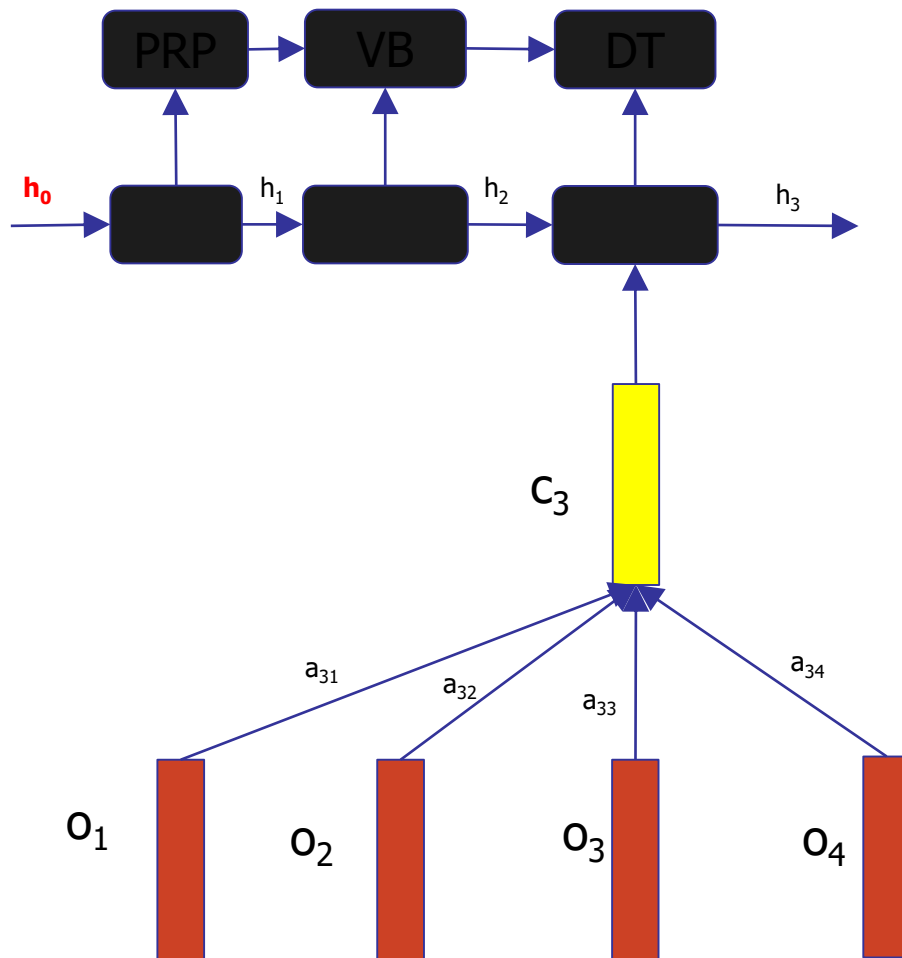
a_{ij} : annotation weight for the j^{th} annotation vector

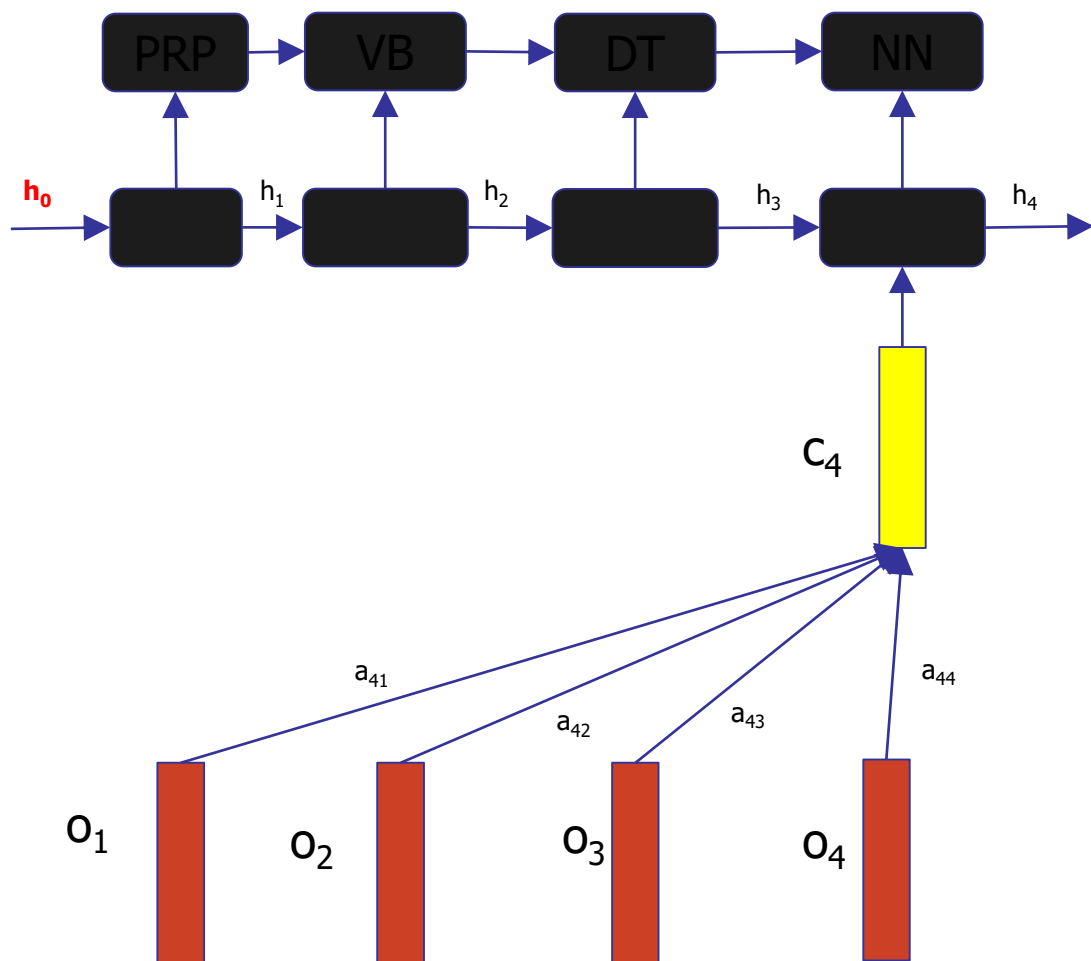
o_j : j^{th} annotation vector

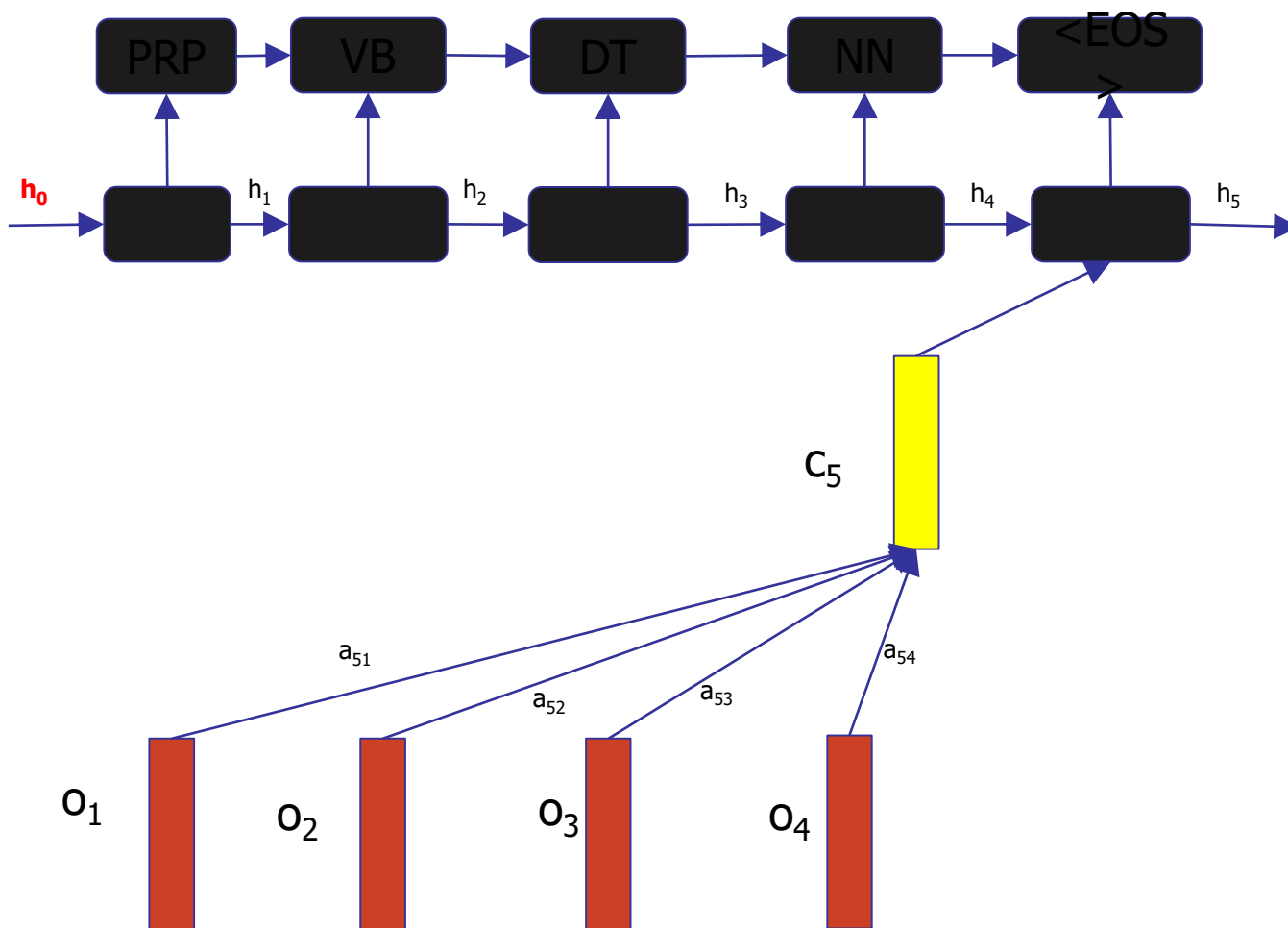


*Let's see an example of how the **attention mechanism** works*









*But we do not know the attention weights?
How do we find them?*

Let the training data help you decide!!

Idea: Pick the attention weights that maximize the POS tagging accuracy

(more precisely, decrease training data loss)

Have an attention function that predicts the attention weights:

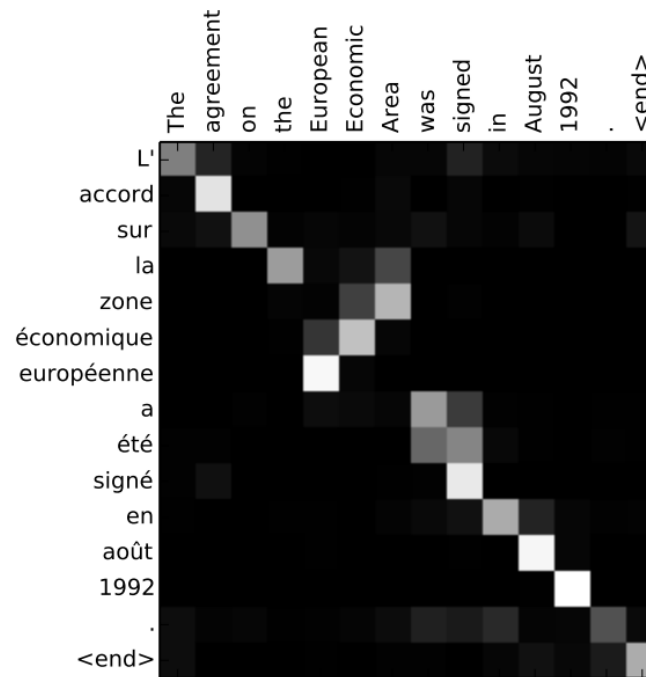
$$a_{ij} = \mathbf{A}(o_j, h_i; \mathbf{o})$$

A could be implemented as a feedforward network which is a component of the overall network

Then training the attention network with the rest of the network ensures that the attention weights are learnt to minimize the translation loss

OK, but do the attention weights actually show focus on certain parts?

Here is an example of how attention weights represent a soft alignment for machine translation



Let's go back to the encoder. What type of encoder cell should we use there?

- Basic RNN: models sequence history by maintaining state information
 - But, cannot model long range dependencies
- LSTM: can model history and is better at handling long range dependencies

The RNN units model only the sequence seen so far, cannot see the sequence ahead

- Can use a bidirectional RNN/LSTM
- This is just 2 LSTM encoders run from opposite ends of the sequence and resulting output vectors are composed

Both types of RNN units process the sequence sequentially, hence parallelism is limited

Alternatively, we can use a **CNN**

- Can operate on a sequence in parallel
- However, cannot model entire sequence history
- Model only a short local context. This may be sufficient for some applications or deep CNN layers can overcome the problem

Perspectivization

MT v/s POS tagging!

- Similarity

- POS

- Every word in a sentence has one corresponding tag.

- MT

- Every word in a sentence has one (or more) corresponding translated word.

- Difference

- Order: Order of translated word may change.
 - Fertility: One word corresponds to many.
Many to one also possible.

Complexity

- POS and HMM
 - Linear time complexity
- MT and Beam search
 - Exponential time complexity
 - Permutation of words produces exponential search space
- However, for related languages, MT is like POS tagging

Properties of related languages

1. Order preserving

2. Fertility ~ 1

3. Morphology preserving

Hindi	<i>Jaaunga</i>
Bengali	<i>Jaabo</i>
English	<i>Will go</i>

Hindi & Bengali ↑

Hindi & English ↓

Properties of related languages

4. Syncretism: Suffix features should be similarly loaded

Hindi	Main <i>jaaunga</i>	Hum <i>jaayenge</i>	Hindi & Bengali
Bengali	Ami <i>jaabo</i>	Aamra <i>jaabo</i>	



5. Idiomaticity: Literal translation should be high

Hindi	<i>Aap Kaise Ho?</i>	Hindi & Bengali
Bengali	<i>Aapni Kemon Achen?</i>	
English	<i>How do you do?</i>	Hindi & English



Reading List

- TnT (<http://www.aclweb.org/anthology-new/A/A00/A00-1031.pdf>)
- Brill Tagger
(http://delivery.acm.org/10.1145/1080000/1075553/p112-brill.pdf?ip=182.19.16.71&acc=OPEN&CFID=129797466&CFTOKEN=72601926&acm__=1342975719_082233e0ca9b5d1d67a9997c03a649d1)
- Hindi POS Tagger built by IIT Bombay
(<http://www.cse.iitb.ac.in/pb/papers/ACL-2006-Hindi-POS-Tagging.pdf>)
- Projection
(<http://www.dipanjandas.com/files/posInduction.pdf>)