CS626: Speech, NLP and Web

Discriminative POS Tagging (CRF, MEMM), Evaluation, Need for POS Tagging

Pushpak Bhattacharyya

Computer Science and Engineering Department

IIT Bombay

Week 4 of 19th August, 2024

1-slide recap of week of 12th Aug

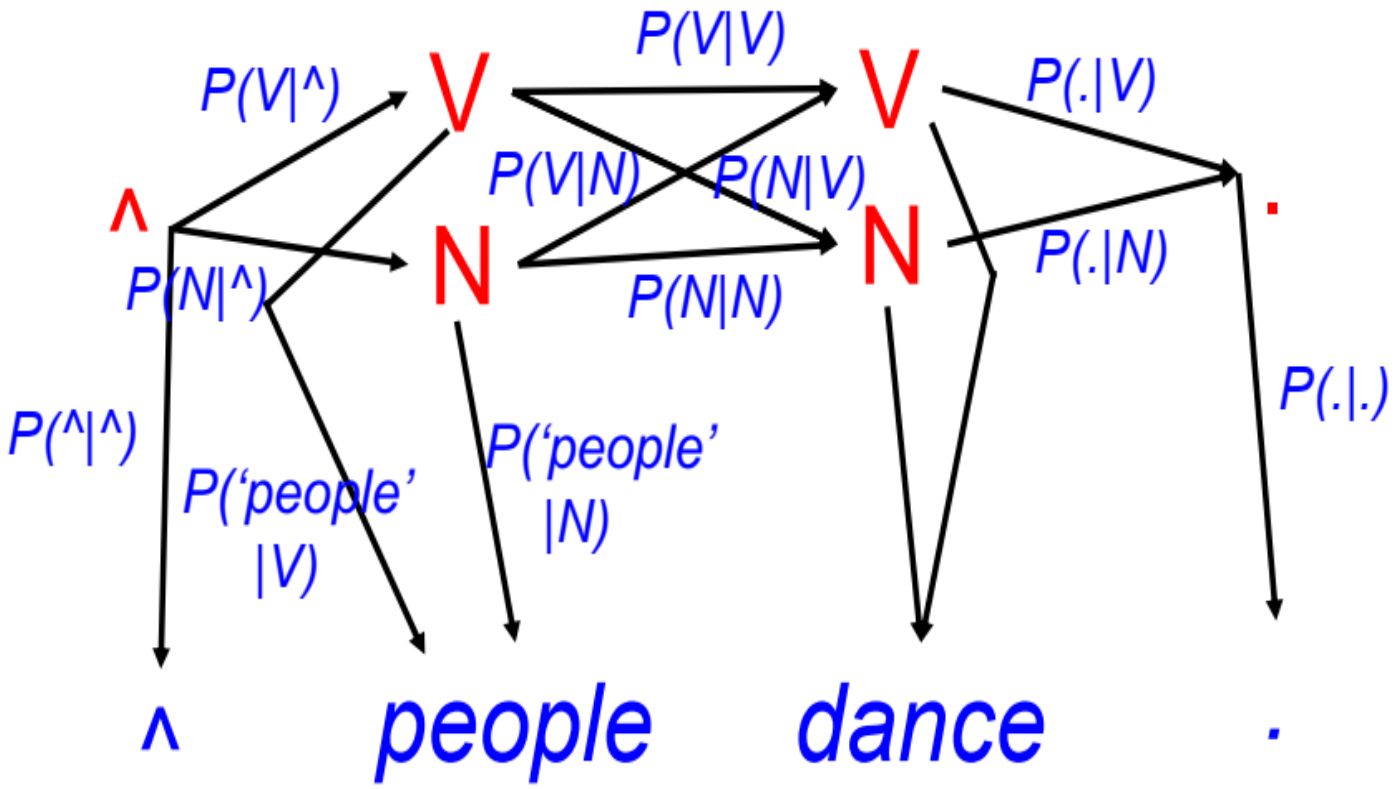
- Should I apply Bayes Rule: ***Cancer detection vs. Visa Application***
- Illustration of Viterbi with ***People laugh → People_NN laugh_VB***
- Why is Viterbi linear time: ***Pruning of paths*** due to Markov Independence assumption
- What is “large” about large language models: ***probability of large (i.e. long sequences)***, plus the model having large number of parameters
- 3 tasks solved by HMM- ***Viterbi, Forward-Backward, Baum Welch***

POS tag assignment statement: HMM part

- Implement a POS tagger in Python using the Hidden Markov Model
- Input and output: Dataset: Brown corpus (tagset = "universal")
- Output: Tagged Sequence
- Accuracy (5-fold cross-validation), confusion matrix, per POS accuracy
- Demo
- HMM needs to be implemented from scratch

Trellis based search

$$T^* = \text{argmax}_T[P(T|W)] = \text{argmax}_T[P(T)P(T|W)]$$



POS tag assignment statement: CRF Part

- IMP: you can use available CRF tools like
  - CRF++
  - CRFSuite (very popular for NLP): install the Python bindings using pip install python-crfsuite
  - Pytorch CRF
  - …etc.
- The probability *P(T|W)* is modelled using discriminative modelling- CRF

What is CRF

- A type of probabilistic graphical model
- Used for structured prediction
- Particularly effective for tasks where the output variables are not independent, but rather have dependencies that must be modeled jointly

CRF: Basics

- Undirected graphical models
- Model the relationships between variables without assuming a specific direction of influence
- The structure of the graph represents the dependencies among the variables
- Model ***P(Y|X)***, where *Y* is the set of labels and *X* is the set of observed features

Formal Definition

- Given:
- $X=(X_1,X_2,\ldots,X_T)$: The sequence of observed variables (e.g., words in a sentence)
- $Y=(Y_1,Y_2,\ldots,Y_T)$: The sequence of output variables (e.g., labels corresponding to each word)

Governing Equation

$$P(Y \mid X) = \frac{1}{Z(X)} \exp\left(\sum_c \varphi_c(Y_c, X_c) \right)$$

- $\psi_c(Y_c, X_c)$ are the potential functions over the cliques c in the graph (subsets of variables that are fully connected)
- $Z(X)$: Z(X) is the partition function, which ensures the probabilities sum to 1

Potential Function $\psi_c(Y_c, X_c)$

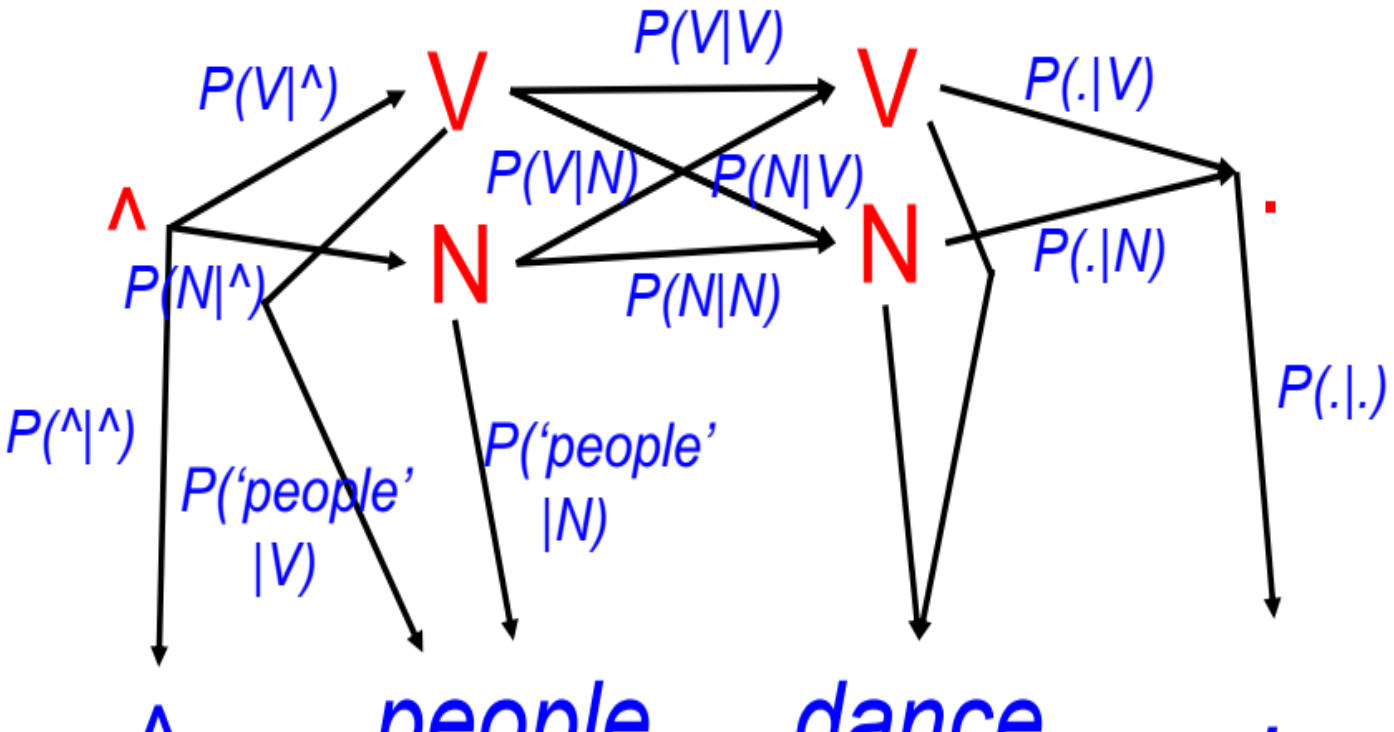
$$\varphi_c(Y_c, X_c) = \exp\left(\sum_k \lambda_k f_k(Y_c, X_c)\right)$$

- $f_k(Y_c, X_c)$: feature functions
- λ_k: weights of the features

CRF based POS tagging

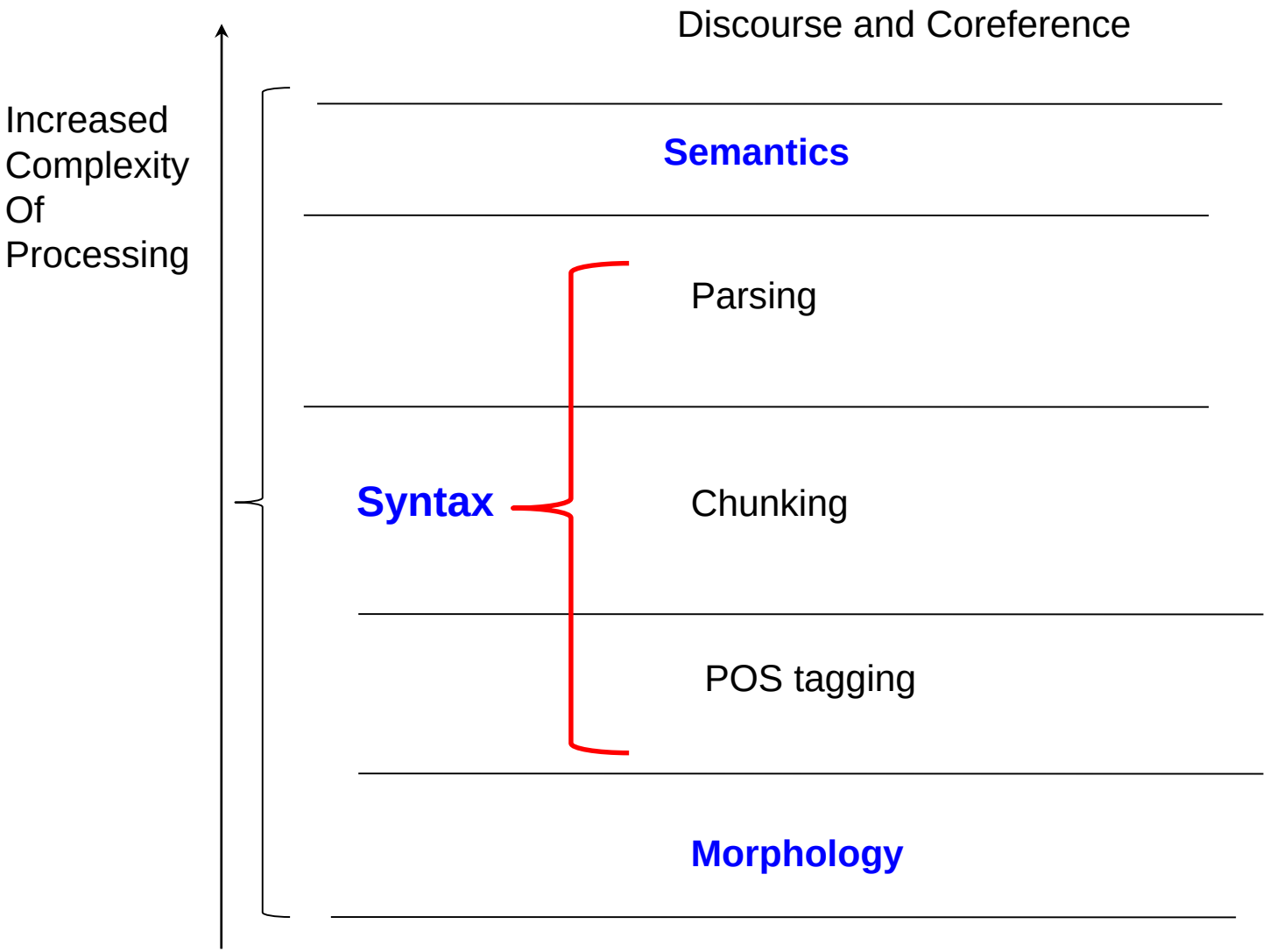
$$P(Y \mid X) = \frac{1}{Z(X)} \exp\left(\sum_c \varphi_c(Y_c, X_c) \right)$$

$$P(Y \mid X) = \frac{\prod_i \varphi_i(y_{i-1}, y_i, x)}{\sum_{y' \in \gamma(x)} \prod_i \varphi_i(y'_{i-1}, y'_i, x)}$$



Discriminative POS Tagging

NLP Layers


Discourse and Coreference
Increased
Complexity
Of
Processing
Semantics
Parsing
Syntax
Chunking
POS tagging
Morphology

POS Task

Input: *Who is the prime minister of India ?_PUNC*

Output: *Who_WP is_VZ the_DT prime_JJ minister_NN of _IN India_NNP ?_PUNC*

Recall Generative POS Tagging

Best tag sequence

$$T^* = \text{argmax}_T \, P(T|W) = \text{argmax}_T \, P(T)P(W|T)$$

Modelling in Discriminative POS Tagging

- *T** is the best possible tag sequence

$$T^* = \arg\max_T P(T \mid W)$$

$$P(T \mid W) = \sum_F P(T, F \mid W) = P(T, F \mid W)$$

$$= P(F \mid W).P(T \mid F, W)$$

$$= 1.P(T \mid F) = P(T \mid F)$$

$$P(T \mid F) = \prod_{i=0}^{n+1} [P(t_i \mid F_i)]$$

Motivation

- HMM based POS tagging cannot handle “free word order” and “agglutination” well
- If *adjective after noun* is equally likely as *adjective before noun*, the transition probability is no better than uniform probability which has high entropy and is uninformative.
- When the words are long strings of many morphemes, POS tagging w/o morph features is highly inaccuarte.

Variability in word order: problem for generative model

Both SOV and SVO!

Agglutination: problem for generative model

- *istahtaisinkohan "I wonder if I should sit down for a while"*
  - *ist: "sit", verb stem*
  - *ahta: verb derivation morpheme, "to do something for a while"*
  - *isi: conditional affix*
  - *n: first-person singular suffix*
  - *ko: question particle*
  - *han: a particle for things like reminder (with declaratives) or "softening" (with questions and imperatives)*

Agglutination in Manipuri

- Words in Manipuri can consists of ten or more morphemes

- *pusinhanjaramgadabanidako* (“I wish (I) myself would have caused to bring in the article”)

- *pu-sin-han-ja-ram-ga-da-ba-ni-da-ko*

Modelling in Discriminative POS Tagging

- *T** is the best possible tag sequence
- Summation dropped, because given *W* and feature engineering, *F* is unique; also *P(F|T)=1*
- The final independence assumption is that the tag at any position *i* depends only on the feature vector at that position

$$T^* = \arg\max_T P(T \mid W)$$

$$P(T \mid W) = \sum_F P(T, F \mid W) = P(T, F \mid W)$$

$$= P(F \mid W).P(T \mid F, W)$$

$$= 1.P(T \mid F) = P(T \mid F)$$

$$P(T \mid F) = \prod_{i=0}^{n+1} [P(t_i \mid F_i)]$$

Feature Engineering

- Running example: ^ ***brown foxes jumped over the fence .***
- *A. Word-based features*

  f_{21} – dictionary index of the current word ('foxes'): integer

  f_{22} – -do- of the previous word ('brown'): integer

  f_{23} – -do- of the next word ('jumped'): integer

B. Part of Speech (POS) tag-based feature

  f_{24} – index of POS of previous word (here JJ): integer

Feature engineering cntd.

- ^ ***brown foxes jumped over the fence .***

- *C. Morphology-based features*
  - f_{25}– does the current word (‘foxes’) have a noun suffix, like ‘s’, ‘es’, ‘ies’, etc.: 1/0- here the value is
  - f_{26}– does the current word (‘foxes’) have a verbal suffix, like ‘d’, ‘ed’, ‘t’, etc.: 1/0- 0
  - f_{27} and f_{28} for ‘brown’ like for ‘foxes
  - f_{29} and $f_{2,10}$ for ‘jumped’ like for ‘foxes; here $f_{2,10}$ is 1 (jumped has ‘ed’ as suffix)

An Aside: word vectors

- These features are opaquely represented in word vectors created from huge corpora
- Word vectors are vectors of numbers representing words
- It is not possible to tell which component in the word vector does what

Modelling Equations

$\boldsymbol{W}$: $\wedge w_0 w_1 w_2 \ldots w_{n-2} w_{n-1} w_n$. $\boldsymbol{T}$: $\wedge t_0 t_1 t_2 \ldots t_{n-2} t_{n-1} t_n$.

$$P(T) = \prod_{i=0}^{n+1} [P(t_i \mid F_i)]$$

$$P(t_i = t \mid F_i) = \frac{e^{\sum_{j=1.k} \lambda_j f_{ij}}}{\sum_{t' \in S} e^{\sum_{j=1.k} \lambda_j f_{ij}(t')}}$$

Maximum Entropy Markov Model (MEMM)

S: set of tags.

The sequence probability of a tag sequence T is the product of $P(t_i|F_i)$, i varying over the positions.

Beam Search Based Decoding

- ^ ***The brown foxes jumped .***
- Let us assume the following tags for the purpose of the discussion:
  - D- determiner like ‘the’
  - A- adjective like ‘brown’
  - N- noun like ‘foxes’, ‘fence’
  - V- verb like ‘jumped’

- Let the decoder start at the state ‘^’ which denotes start of the sentence.

Step-1

- ^ ***The brown foxes jumped .***
- The word '*the*' is encountered. First there are 4 next states possible corresponding to 4 tags, giving rise to 4 possible paths:
- ^ *D* $-P_1$
- ^ *A* $-P_2$
- ^ *N* $-P_3$
- ^ *V* $-P_4$

Commit to Beam Width

- Beam width is an integer which denotes how many of the possibilities should be kept *open*.
- Let the beam width be 2.
  - **This means that out of all the paths obtained so far we retain only the top 2 in terms of their probability scores.**
- We will assume that the actual linguistically viable sub-sequence appears amongst the top two choices.
  - 'The' is a determiner and we get the two highest probability paths for "^ The" as P_1 and P_3.

Step-2

- ^ ***The brown foxes jumped .***
- '*brown*' is the next word. P_1 and P_3 are extended as
- ^ *D D* -P_{11}
- ^ *D A* -P_{12}
- ^ *D N* -P_{13}
- ^ *D V* -P_{14}
- ^ *N D* -P_{31}
- ^ *N A* -P_{32}
- ^ *N N* -P_{33}
- ^ *N V* -P_{34}

Retain two paths

- Keep two possibilities corresponding to correct/almost-correct sub-sequences. '*brown*' is an adjective, but can be noun too (e.g., "*the brown of his eyes*").

^ *D A* $-P_{12}$

^ *D N* $-P_{13}$

Step-3

- ***^ The brown foxes jumped .***
- Can be both noun and verb (verb: "*he was foxed by their guile*").
- From P_{12} and P_{13}, we will get 8 paths, but retain only two, as per the beam width.
- We assume only the paths coming from P_{12} survive with 'A' and 'N' extending the paths:

^ *D A A*	$-P_{122}$ (this is a wrong path!)
^ *D A N*	$-P_{123}$

Step-4

- ^ ***The brown foxes jumped .***

- Can be both a past participial adjective (“*the halted train*”) and a verb.

- Retaining only two top probability paths we get

^ *D A N A* $-P_{1232}$

^ *D A N V* $-P_{1234}$

Step-5

- ^ ***The brown foxes jumped .***

- Can be both a past participial adjective (“*the halted train*”) and a verb.

- Retaining only two top probability paths we get

^ *D A N A* $-P_{1232}$

^ *D A N V* $-P_{1234}$

Step-6: Final Step

- ^ ***The brown foxes jumped .***

- On encountering dot, the beam search stops.
- We assume we get the correct path probabilistically in the beam (width 2)

  ^ *D A N V.*

How to fix the beam width (1/2)

- English POS tagging with Penn POS tag set: approximately 40 tags
- Fine categories like NNS for plural NNP for proper noun, VAUX for auxiliary verb, VBD for past tense verb and so on.
- A word can have on an average at most 3 POSs recorded in the dictionary.

How to fix the beam width (2/2)

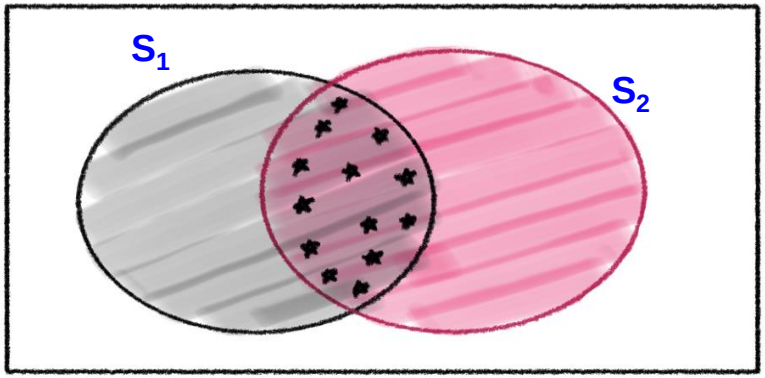
- Allow for 4 finer category POSs under each category and with support from a lexicon that records the broad category POSs

Penn POS TAG Set

1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjun
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative

Evaluation

False Positives, False Negatives, Precision, Recall, F-score



SET S_1 ... OBTAINED
SET S_2 ... ACTUAL
$S_1 \cap S_2$ ... TRUE POSITIVES
$S_1 - (S_1 \cap S_2)$ ... FALSE POSITIVES
$S_2 - (S_1 \cap S_2)$ ... FALSE NEGATIVES
$(S_1 \cup S_2)^c$ ... TRUE NEGATIVES

$$Precision = \frac{|S_1 \bigcap S_2|}{|S_1|}$$

$$Recall = \frac{|S_1 \bigcap S_2|}{|S_2|}$$

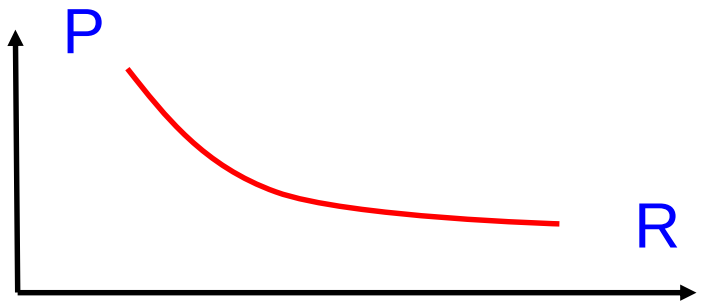
Generalized F-score

$$F_\beta = \frac{(1+\beta^2)PR}{\beta^2 P + R} = \frac{1}{\frac{\beta^2}{(1+\beta^2)R} + \frac{1}{(1+\beta^2)P}}$$

As $\beta \rightarrow 0$, $F_\beta \rightarrow P$ and as $\beta \rightarrow \infty$, $F_\beta \rightarrow R$

Why F-score?

- P and R need balancing act
- P vs. R is a falling curve
- Harmonic mean gives importance to the smallest of the entities
- We cannot afford to be very low on either P or R
- Hence F-score



3 Generations of POS tagging techniques

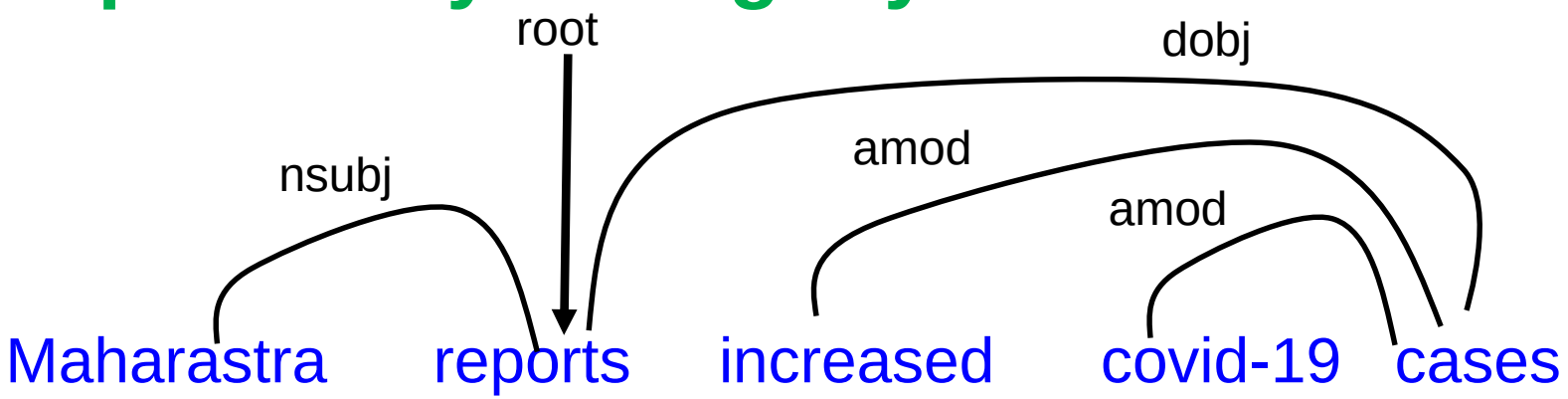
- Rule Based POS Tagging
  - Rule based NLP is also called Model Driven NLP
- Statistical ML based POS Tagging (*Hidden Markov Model, Support Vector Machine*)
- Neural (Deep Learning) based POS Tagging

Necessity of POS Tagging (1/2)

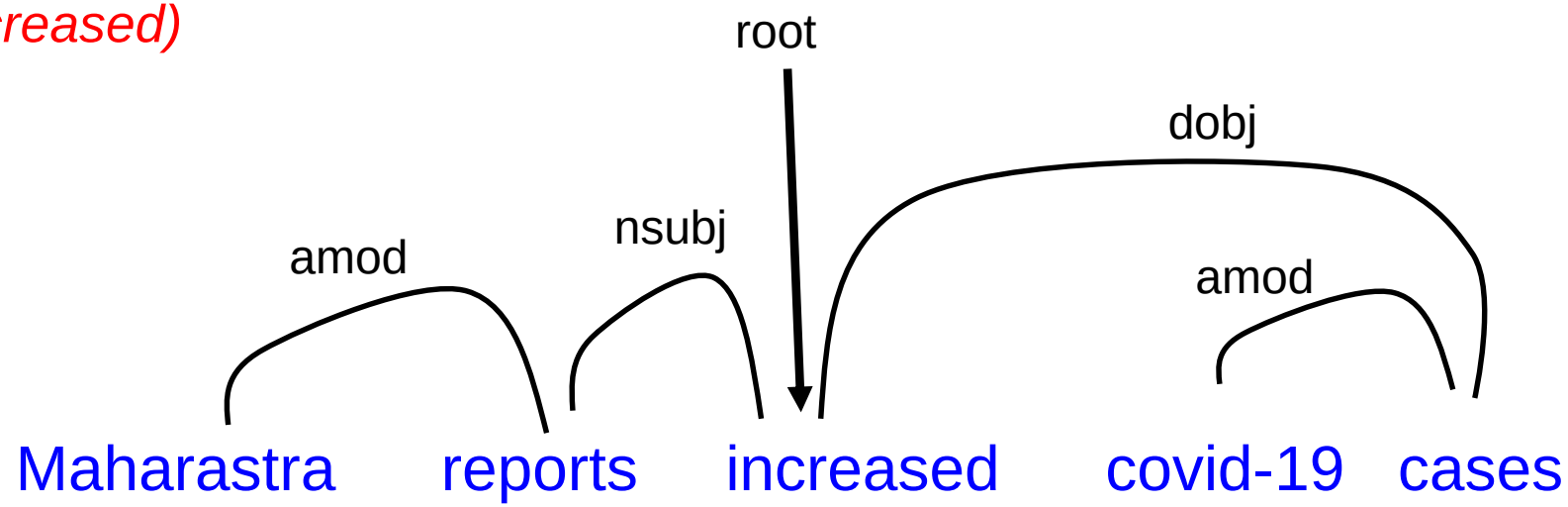
- **Command** Center to Track Best Buses (ToI 30Jan21): POS ambiguity affects

- Elderly with young **face** increased covid 19 risk (ToI Oct 20)

Dependency Ambiguity

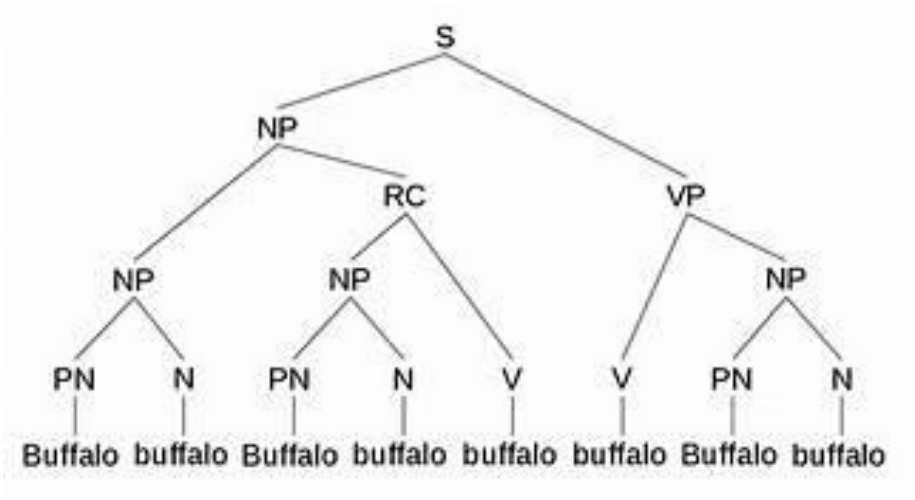


(it is reported by Maharastra Govt. that covid-19 cases have increased)



(it is the Maharastra reports that have increased covid-19 cases!!!)

Buffalo Sentence



The sentence can be parsed as follows: "Buffalo buffalo (Buffalo bison) Buffalo buffalo (Buffalo bison) buffalo (intimidate) buffalo (intimidate) Buffalo buffalo (Buffalo bison)"[1]. In other words, the sentence claims that bison from Buffalo, New York, who are intimidated by other bison in their community, in turn intimidate other bison in their community[1].

Why probability for POS tagging

Data for “present”

He gifted me the/a/this/that **present_NN**.

They **present_VB** innovative ideas.

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 the future.*

      Noun but not preceded by “the”

Rules for disambiguating “present”

- 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** fortells the future.

- Does this rule guarantee 100% precision and 100% recall?

Need for ML in POS tagging

- Rules are challenged by new data
- Need a robust system.

- Machine learning based POS tagging:
  - HMM (Accuracy increased by 10-20% against rule based systems)
  - Jelinek's work inspired from ASR