CS689: Computational Linguistics for Indian Languages
Parts-of-Speech (POS) Tagging

Arnab Bhattacharya
arnabb@cse.iitk.ac.in

Computer Science and Engineering,
Indian Institute of Technology, Kanpur
http://web.cse.iitk.ac.in/~cs689/

2nd semester, 2023-24

Tue 10:30–11:45, Thu 12:00–13:15 at RM101/KD102

Parts of Speech (POS)

- Parts-of-Speech (POS) are *roles/functions* that a word takes in a sentence
- POS Tagging is the process of assigning *every* word in a sentence a POS
- Example of sequence labeling task
- Input is a word sequence $X = \{x_i\}$ of length n
- Output is a POS tag sequence $Y = \{y_i\}$ of length n with a one-to-one correspondence

English POS Tags

- Penn Treebank tagset: 45

Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coord. conj.	*and, but, or*	NNP	proper noun, sing.	*IBM*	TO	"to"	*to*
CD	cardinal number	*one, two*	NNPS	proper noun, plu.	*Carolinas*	UH	interjection	*ah, oops*
DT	determiner	*a, the*	NNS	noun, plural	*llamas*	VB	verb base	*eat*
EX	existential 'there'	*there*	PDT	predeterminer	*all, both*	VBD	verb past tense	*ate*
FW	foreign word	*mea culpa*	POS	possessive ending	*'s*	VBG	verb gerund	*eating*
IN	preposition/ subordin-conj	*of, in, by*	PRP	personal pronoun	*I, you, he*	VBN	verb past participle	*eaten*
JJ	adjective	*yellow*	PRP$	possess. pronoun	*your, one's*	VBP	verb non-3sg-pr	*eat*
JJR	comparative adj	*bigger*	RB	adverb	*quickly*	VBZ	verb 3sg pres	*eats*
JJS	superlative adj	*wildest*	RBR	comparative adv	*faster*	WDT	wh-determ.	*which, that*
LS	list item marker	*1, 2, One*	RBS	superlatv. adv	*fastest*	WP	wh-pronoun	*what, who*
MD	modal	*can, should*	RP	particle	*up, off*	WP$	wh-possess.	*whose*
NN	sing or mass noun	*llama*	SYM	symbol	*+,%, &*	WRB	wh-adverb	*how, where*

- Penn Treebank corpora has annotations based on this

Universal POS Tags

- Universal Dependencies (UD) tagset: 17

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red, young, awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very, slowly, home, yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm, cat, mango, beauty*
	VERB	words for actions and processes	*draw, provide, go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina, IBM, Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh, um, yes, hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by, under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that, which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

- Closed class has a more or less fixed set of members
- Open class gets new words added much more frequently

More Granularity

- Nouns can be proper or common, which can again be count or mass
- Verbs can be more fine-grained such as base form, with tense, continuous, present participle, gerund, past participle, etc.
- Verb forms have tense, aspect and mood
  - Together called TAM
  - Sometimes also called tense-modality-aspect (TMA)

POS Tags in Indian Languages

- Indian languages mostly follow the UD tags called UPOS
- Sanskrit does not have symbols (modern Sanskrit does)
- Language-specific XPOS tags
- Word groups for Indian languages
  - Typographical convention is to introduce whitespaces
  - One semantic unit
- Types of word groups
  - `Vibhakti` groups: Rama_ko
  - `Kriyā` groups: khaate_khaate
  - `Samāsa` groups: Rama_Lakshmana
  - Named entity: Subhas_Chandra_Bose

Verbs in Indian Languages

- TAM tags capture three dimensions of a verb
- Tense to denote *time of action*
  - past, future, present
- Aspect to denote *extension of action*
  - simple, perfective, habitual, progressive
- Mood to denote *reality of action*
  - indicative, conditional, potential, hortative, desiderative, optative, injunctive, presumptive, contrafactual, benedictive, interrogative, necessitive
- A single Indian language may not have all possibilities
- Verbs also have voice to denote the *way* of speaking
  - active (kartṛvācya), passive (karmavācya, bhāvavācya)

POS Tags for Indian Languages

- Viśeṣya (noun)
  - Sāmānya (common noun): pustaka
  - Viśeṣa (proper noun): Mahabharata
  - Deśakālasāpekṣa (spatio-temporal noun): upara, aage
- Sarvanāma (pronoun)
  - Sāmānya (common): aapa
  - Praśnvācaka (interrogative): kauna, kaba, kahaa
  - Saṃketavācaka (demonstrative): kisa, kauna
- Kriyā (verb)
  - Samāpikā (complete verb): khaate hain
  - Asamāpikā (incomplete verb)
    - Nimittārthaka (causal): khaane ke liye
    - Samānakāla (continuous): khaate khaate
    - Adjective-like: khaate huye
    - Time relations: khaake
- Viśeṣaṇa (adjective): sundara
  - Guṇavācaka Viśeṣaṇa: bahut
  - Parimāṇavācaka Viśeṣaṇa: bahut

POS Tags for Indian Languages (contd.)

- Kriyā-Viśeṣaṇa: jaldi, sundara
- Avyaya (indeclineable)
  - Samanvayaka (conjunction): aur, kintu
  - Adhīnastha (subordinate): agaara, to
  - Uktivācaka (quotative): iti
  - Vismayādibodhaka (interjection): are, he
  - Tīvratābodhaka (intensifier): bahuta
  - Nakārātmaka (negative): naa, nehii
- Parimāṇavācī Parimaanavachii (quantifier)
  - Sāmānya (common): thoraa
  - Gaṇanāsūcaka (count): eka
  - Kramasūcaka (order): pahalaa
- Avaśeṣa (others)
  - Foreign word: book
  - Symbol: &
  - Punctuation: .
  - Unknown: $<$ foreign script $>$

POS Tagging as a Sequence Task

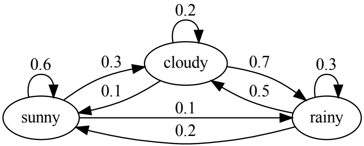
- Given a sequence of word groups (as a sentence), mark the POS tag for every word
- Sequence-to-sequence task
- A sentence W of length n consisting of word groups in order: w_i
- POS tag of word group w_i is t_i
- Find all POS tags $t_1, \ldots, t_n$
- Several tools

Markov Chains

- Sequential process that moves from state to state
- Markov property: current state depends *only* on previous state
  - *1st order*
- At every state, there is an observation
- Observation is *equal* to the state
- Markov chain

Example of Markov Chain

- Weather
- Three states: sunny, cloudy, rainy



- A number of states
- A transition matrix of among states
- A start vector for states

$$MC = \langle n; T; \pi \rangle$$

$$n = 3; \quad T = \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.7 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}; \quad \pi = \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix}$$

Training an MC

- Learning an MC is simply finding the count ratios
- Number of states is obvious from the data
- Start state probabiities

$$\pi(t_i) = \frac{\#\text{start state is } i}{\#\text{all start states}} = \frac{C(i)}{\sum_{\forall i} C(i)}$$

- Transition probabilities

$$P(t_j|t_i) = \frac{\#\text{observation is } i \text{ immediately followed by } j}{\#\text{observation is } i} = \frac{C(i \to j)}{\sum_{\forall j} C(i \to j)}$$

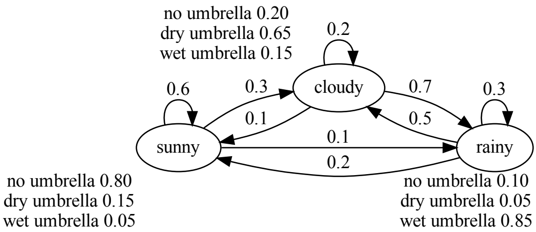
Hidden Markov Model

- States are *hidden*
- States are only indicated by the observations
- A person in a closed room watching others enter with umbrellas trying to guess weather
- POS tags are *hidden states* while words are *observations*
- Number of states η
- Number/range of observations
  - Number of observations need not be equal to number of states
- States have transition probabilities A among them
- There is an observation probability distribution B from each state
  - Can be discrete or continuous
- Start state probability distribution Π

$$HMM = \langle \eta; A; B; \pi \rangle$$

HMM Example

- Weather



$$HMM = \langle \eta; A; B; \pi \rangle$$

$$\eta = 3; A = \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.7 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}; B = \begin{bmatrix} 0.80 \\ 0.15 \\ 0.05 \end{bmatrix} \begin{bmatrix} 0.20 \\ 0.65 \\ 0.15 \end{bmatrix} \begin{bmatrix} 0.10 \\ 0.05 \\ 0.85 \end{bmatrix}; \pi = \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix}$$

POS Tagging as HMM Decoding

- Given a sequence of words w_i of length n, mark the POS tag t_i for every word
- Choose the tag sequence that is the *most probable* given the word sequence

$$\hat{t}_{1:n} = \arg\max P(t_1 \ldots t_n | w_1 \ldots w_n)$$

- Using Bayes' rule for conditional probabilities

$$\begin{aligned}
\hat{t}_{1:n} &= \arg\max P(t_1 \ldots t_n | w_1 \ldots w_n) \\
&= \arg\max \frac{P(w_1 \ldots w_n | t_1 \ldots t_n) \cdot P(t_1 \ldots t_n)}{P(w_1 \ldots w_n)} \\
&= \arg\max P(w_1 \ldots w_n | t_1 \ldots t_n) \cdot P(t_1 \ldots t_n)
\end{aligned}$$

Simplifications

- Markov assumption

$$\begin{aligned} P(t_1 \dots t_n) &= \pi(t_1) \cdot P(t_2|t_1) \cdot P(t_3|t_2, t_1) \dots P(t_n|t_1 \dots t_{n-1}) \\ &\approx P(t_2|t_1) \cdot P(t_3|t_2) \dots P(t_n|t_{n-1}) = \Pi_{\forall i} P(t_i|t_{i-1}) \end{aligned}$$

- Independence assumption

$$P(w_1 \dots w_n|t_1 \dots t_n) = \Pi_{\forall i} P(w_i|t_i)$$

- Combining

$$\begin{aligned} \hat{t}_{1:n} &= \arg\max P(t_1 \dots t_n|w_1 \dots w_n) \\ &= \arg\max \Pi_{\forall i} \left(P(w_i|t_i) \cdot P(t_i|t_{i-1})\right) \end{aligned}$$

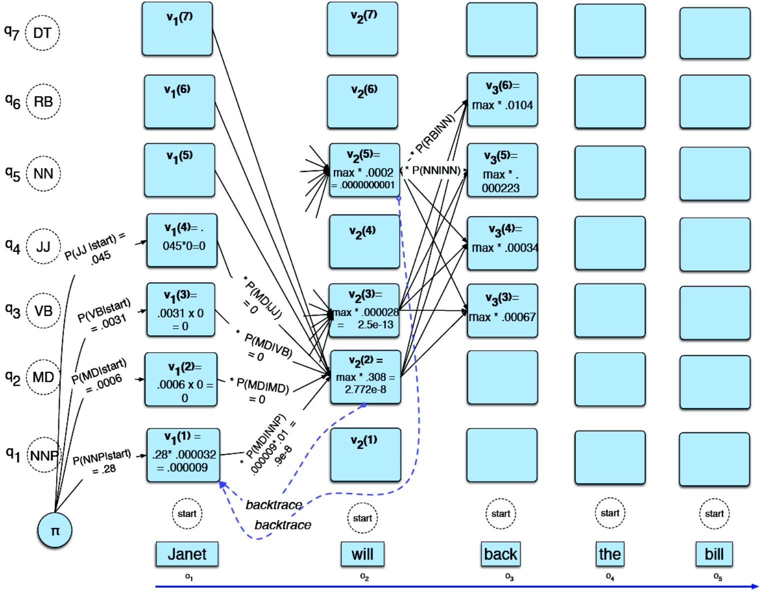
Viterbi Algorithm

- Viterbi algorithm is a *dynamic programming* solution
- a_{ij} denotes transition from state (tag) i to state (tag) j
- $b_j(k)$ denotes observation of word k from state (tag) j
- π_i denotes start from state (tag) i
- Cell $v_t(j)$ denotes *maximum probability* that state (tag) sequence is at state (tag) j at time t (word w_t)

$$v_t(j) = \max_{i=1}^{N} \left(v_{t-1}(i) \cdot a_{ij} \cdot b_j(t) \right)$$

  - Maximum over all possible states (tags) at time $t-1$
  - *Back pointers* to keep track: $\arg\max$
- Finally, maximum of $v_n(\cdot)$

Example



q7 DT
q6 RB
q5 NN
q4 JJ
q3 VB
q2 MD
q1 NNP
π
P(JJ |start) = .045
P(VB|start) = .0031
P(MD|start) = .0006
P(NNP|start) = .28
v1(7)
v1(6)
v1(5)
v1(4)= .045*0=0
v1(3)= .0031 x 0 = 0
v1(2)= .0006 x 0 = 0
v1(1) = .28* .000032 = .000009
* P(MD|JJ) = 0
* P(MD|VB) = 0
* P(MD|MD) = 0
* P(MD|NNP) = .000009*.01 = 9e-8
v2(7)
v2(6)
v2(5)= max * .0002 = .0000000001
v2(4)
v2(3)= max * .000028 = 2.5e-13
v2(2) = max * .308 = 2.772e-8
v2(1)
* P(RB|NN)
* P(NN|NN)
v3(6)= max * .0104
v3(5)= max * .000223
v3(4)= max * .00034
v3(3)= max * .00067
backtrace
backtrace
start
Janet
will
back
the
bill
o1
o2
o3
o4
o5

Learning and Using an HMM

- Likelihood problem: Given an observation sequence, find its probability
  - Forward algorithm
- Decoding problem: Given an observation sequence, find its most likely state sequence
  - Viterbi algorithm
- Training problem: Given a set of observation sequences, learn the parameters of HMM
  - Baum-Welch algorithm or forward-backward algorithm

Training an HMM

- Number of hidden states from guess or domain knowledge
- Unlike an MC, simple count ratios cannot be used
- Forward probability $\alpha_t(j)$ is probability of being in state j at time t, i.e., after observing $w_1 \ldots w_t$

$$\alpha_t(j) = \sum_{i=1}^{N} \left(\alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(t)\right)$$

- Backward probability $\beta_t(i)$ is probability of observing $w_{t+1} \ldots w_n$ starting from state i

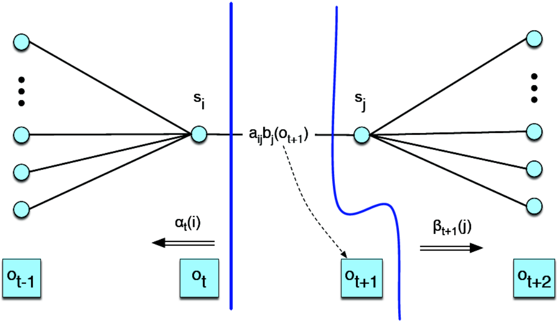
$$\beta_t(i) = \sum_{j=1}^{N} \left(a_{ij} \cdot b_j(t+1) \cdot \beta_{t+1}(j)\right)$$

- *Dynamic programming*

Estimating Probabilities

- Transition from state i to state j for observation sequence W is $\xi(i,j)$

$$\xi(i,j) = \alpha_t(i) \cdot a_{ij} \cdot b_j(t) \cdot \beta_{t+1}(j)$$



- Requires proper *normalization* to produce probabilities
- Probability of observation W

$$P(W) = \sum_{i=1}^{N} (\alpha_t(i) \cdot \beta_t(i))$$

Utility of Features

- Use of features is indirect in HMM
- If capitalized, then high chance of proper noun
- Tag sequence T from word sequence W
- HMM uses Bayes' rule

$$
\begin{aligned}
\widehat{T} &= \arg\max_{T} P(T|W) \\
&= \arg\max_{T} P(W|T) \cdot P(T) = \arg\max_{T} \left(\prod_{\forall i} P(w_i|t_i) \cdot \prod_{\forall i} P(t_i|t_{i-1}) \right)
\end{aligned}
$$

- Alternatively, $P(T|W)$ can be directly computed

Linear Regression

- From a feature vector x, predict class y
- Linear regression

$$y = \vec{w} \cdot \vec{x}$$

  - *Dot product* of vectors: $\vec{w} \cdot \vec{x} = \sum_{\forall i}(w_i \cdot x_i)$
- If 2-class, *sign* of y
- Instead of *hard classification*, sometimes useful to predict *probability of being in a class*
- $w \cdot x$ does not work
  - May be negative
  - Is not constrained between 0 and 1
  - Does not add up to 1

Logistic Regression

- Logistic regression tries to solve these
- Take exponent to convert to positive: $exp(w \cdot x)$
- Take odds: *ratio* of probability of being in class to probability of not being in class

$$\frac{P(y)}{1 - P(y)} = exp(w \cdot x)$$

$$\ln \frac{P(y)}{1 - P(y)} = w \cdot x$$

- This is also called log-odds or the logit function

$$P(y) = \frac{exp(w \cdot x)}{1 + exp(w \cdot x)} = \frac{1}{1 + exp(-w \cdot x)}$$

$$P(\neg y) = 1 - P(y) = \frac{1}{1 + exp(w \cdot x)} = P(-y)$$

- Sigmoid or logistic function

Conditional Random Field

- Conditional Random Field (CRF) directly maximizes $P(T|W)$

$$\begin{aligned}\widehat{T} &= \arg\max_{T} P(T|W) \\ &= \arg\max_{T} P(T,W)/P(W) = \arg\max_{T} P(T,W) \\ &= \arg\max_{T} \sum_{\forall f_k} \left(w_k \cdot F_k(T,W) \right)\end{aligned}$$

- $F_k(T,W)$ are global features that depend on entire word and tag sequences
- Assumption: feature at tag position i is dependent only on tag at position $i-1$
  - Similarity with Markov assumption
- Linear chain CRF
- Global feature is sum of local features

$$F_k(T,W) = \sum_{i=1}^{n} f_k(t_{i-1}, t_i, W)$$

Linear Chain CRF

- Finding most likely *tag sequence* for a word sequence

$$\widehat{T} = \arg\max_{T} \sum_{\forall f_k} \left(w_k \cdot F_k(T, W) \right)$$
$$= \arg\max_{T} \sum_{i=1}^{n} \sum_{\forall f_k} \left(w_k \cdot f_k(t_{i-1}, t_i, W) \right)$$

- Because of *linear chain* assumption, can use Viterbi-like algorithm

$$v_{\tau}(j) = \max_{i=1}^{N} \left(v_{\tau-1}(i) \cdot \sum_{\forall f_k} \left(w_k \cdot f_k(t_{i-1}, t_i, W) \right) \right)$$