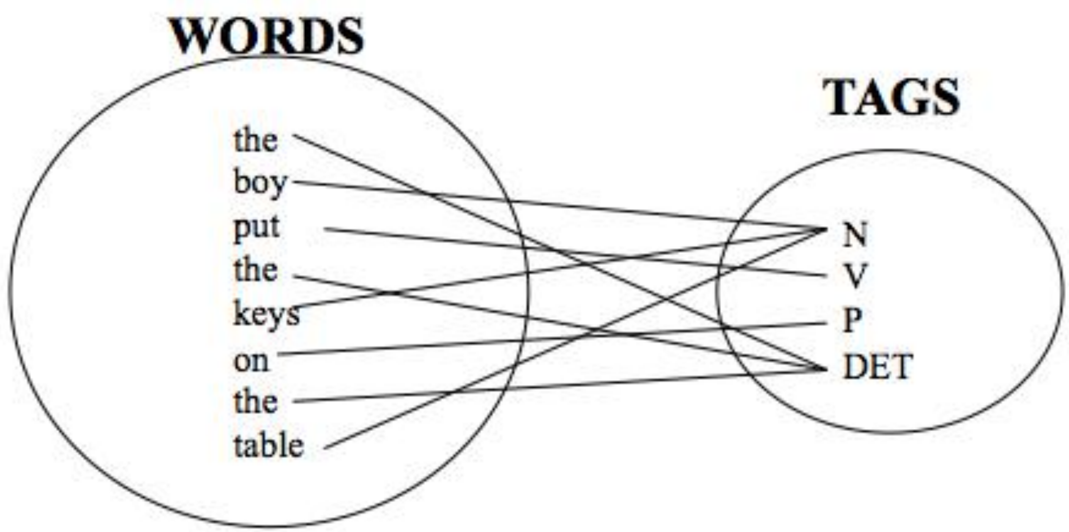
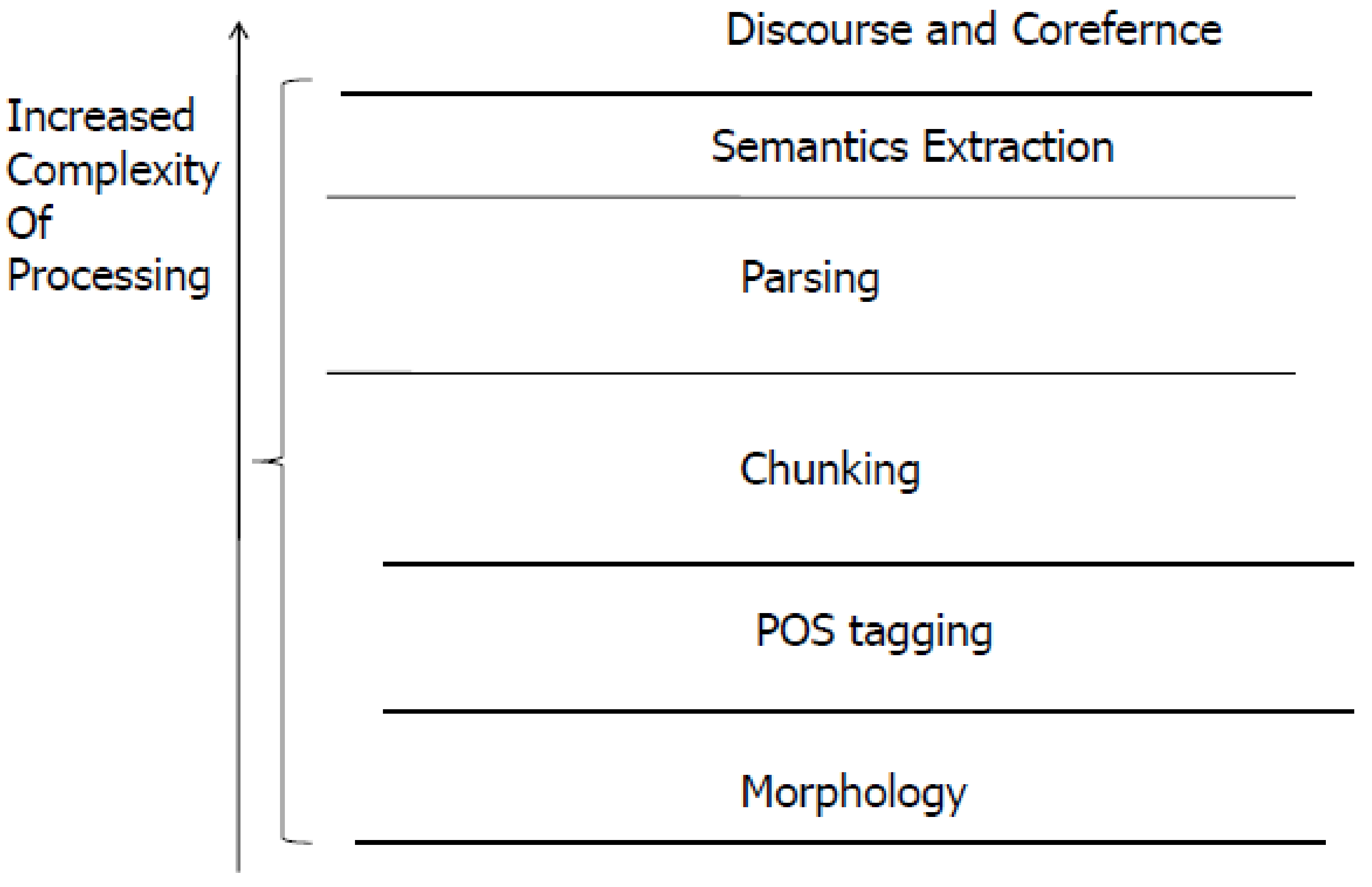
POS tagging

What it is?

- POS Tagging is a process that attaches each word in a sentence with a suitable tag from a given set of tags.



Where does POS tagging fit in



Categories of POS

- Open and closed classes
- Closed classes have a fixed membership of words: determiners, pronouns, prepositions
- Closed class words are usually function word: frequently occurring, grammatically important, often short (e.g. of, it, the, in)
- Open classes: nouns, verbs, adjectives and adverbs

Parts of Speech: How many?

Open class words (content words):

- nouns, verbs, adjectives, adverbs
- mostly content-bearing: they refer to objects, actions, and features in the world
- open class, since new words are added all the time

Parts of Speech: How many?

Closed class words

- pronouns, determiners, prepositions, connectives, ...
- there is a limited number of these
- mostly functional: to tie the concepts of a sentence together

POS examples

N	noun	chair, bandwidth, pacing
V	verb	study, debate, munch
ADJ	adj	purple, tall, ridiculous
ADV	adverb	unfortunately, slowly,
P	preposition	of, by, to
PRO	pronoun	I, me, mine
DET	determiner	the, a, that, those

POS tagging: Choosing a tagset

- To do POS tagging, a standard set needs to be chosen
- Could pick very coarse tagsets

  N, V, Adj, Adv
- More commonly used set is finer grained, "UPenn TreeBank tagset", 45 tags

A Nice Tutorial on POS tags:

https://sites.google.com/site/partofspeechhelp/

UPenn TreeBank POS tag set

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	+,%, &
CD	Cardinal number	*one, two, three*	TO	“to”	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, oops*
EX	Existential ‘there’	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past participle	*eaten*
JJR	Adj., comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj., superlative	*wildest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1, 2, One*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, sing. or mass	*llama*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*llamas*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singular	*IBM*	$	Dollar sign	*$*
NNPS	Proper noun, plural	*Carolinas*	#	Pound sign	#
PDT	Predeterminer	*all, both*	“	Left quote	(‘ or “)
POS	Possessive ending	*’s*	”	Right quote	(’ or ”)
PRP	Personal pronoun	*I, you, he*	(	Left parenthesis	([, (, {, <)
PRP$	Possessive pronoun	*your, one’s*	)	Right parenthesis	( ],), }, >)
RB	Adverb	*quickly, never*	,	Comma	,
RBR	Adverb, comparative	*faster*	.	Sentence-final punc	(. ! ?)
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punc	(: ; ... – -)
RP	Particle	*up, off*			

Definition

Example1:

<s> Come in August, and the COEP campus is abuzz with new and returning students. </s>

After POS tagging :

<s> Come_VB in_IN August_NNP,_, and_CC the_DT COEP_NNP campus_NN is_VBZ abuzz_JJ with_IN new_JJ and_CC returning_VBG students_NNS.

</s>

POS tagging: Definition

- **Example 2:** “_“ The_DT guys_NNS that_WDT make_VBP traditional_JJ hardware_NN are_VBP really_RB being_VBG obsoleted_VBN by_IN microprocessorbased_JJ machines_NNS ,_, ”_” said_VBD Mr._NNP Benton_NNP ._.

Why is POS tagging hard?

- Words often have more than one POS.

Example word: back

- The back door: back/JJ
- On my back: back/NN
- Win the voters back: back/RB
- Promised to back the bill: back/VB

POS tagging problem:

- To determine the POS tag for a particular instance of a word

Brown Corpus: Ambiguous word types

Ambiguity in the Brown corpus:

- 40% of word tokens are ambiguous
- 12% of word types are ambiguous
- Breakdown of ambiguous word types:

Unambiguous (1 tag)	35,340
Ambiguous (2–7 tags)	4,100
2 tags	3,760
3 tags	264
4 tags	61
5 tags	12
6 tags	2
7 tags	1 ("still")

How bad is the ambiguity problem?

- One tag is usually more likely than the others.
  - In the Brown corpus, race is a noun 98% of the time, and a verb 2% of the time
- A tagger for English that simply chooses the most likely tag for each word can achieve good performance
- Any new approach should be compared against the unigram baseline (assigning each token to its most likely tag)

Deciding the correct POS

Can be difficult even for people:

1. Mrs./NNP Shroff/NNP never/RB got/VBD around/_ to/TO joining/VBG.
2. All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/_ the/DT corner/NN.
3. Organic/NNP Onions/NNP costs/VBZ around/_ 250/CD.

Assigning tags:

- Mrs./NNP Shroff/NNP never/RB got/VBD around/RP to/TO joining/VBG.
- All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DT corner/NN.
- Organic/NNP Onions/NNP costs/VBZ around/RB 250/CD.

Relevant knowledge for POS tagging

The word itself:

- Some words may only be nouns, e.g. arrow
- Some words are ambiguous, e.g. flies, like, bank
- Probabilities may help, if one tag is more likely than another

Relevant knowledge for POS tagging

Local context:

- Two determiners rarely follow each other
- Two base form verbs rarely follow each other
- Determiner is almost always followed by adjective or noun

POS tagging: Two approaches

Rule-based Approach:

- Assign each word in the input a list of potential POS tags
- Then reduce down this list to a single tag using hand-written rules

Statistical tagging:

- Get a training corpus of tagged text, learn the transformation rules from the most frequent tags
- Probabilistic: Find the most likely sequence of tags T for a sequence of words W

Probabilistic Tagging: Two different families of models

Problem at hand:

- We have some data {(d, c)} of paired observations d and hidden classes c.

Different instances of d and c:

  - **Part-of-Speech Tagging**: words are observed and tags are hidden.
  - **Text Classification**: sentences/documents are observed and the category is hidden.
  - Categories can be positive/negative for sentiments ..
  - sports/politics/business for documents ...

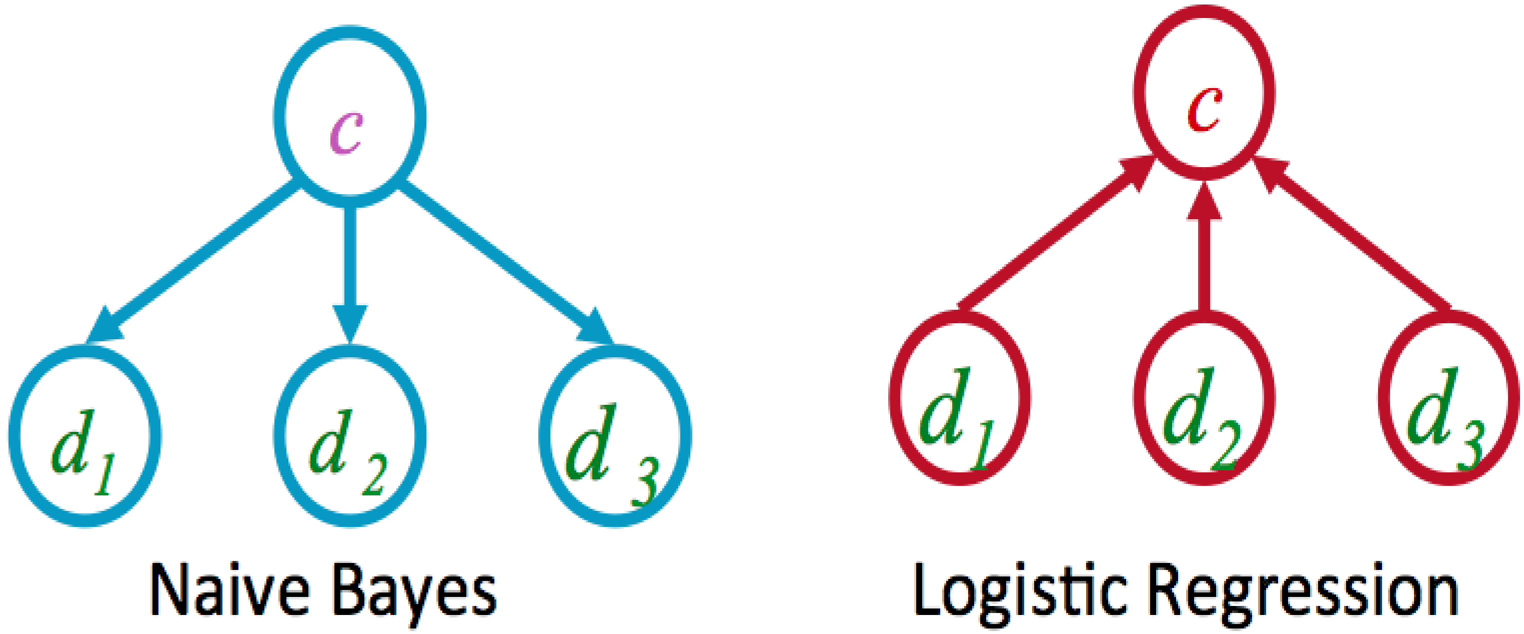
Gives rise to two families?

- Whether they generate the observed data from hidden stuff or the hidden structure given the data?

Generative vs. Conditional Models

- **Generative (Joint) Models:**
  - Generate the observed data from hidden stuff, i.e. put a probability over the observations given the class: P(d, c) in terms of P(d/c)
  - Egs: Naïve Bayes’ classifiers, Hidden Markov Models etc.
- **Discriminative (Conditional) Models:**
  - Take the data as given, and put a probability over hidden structure given the data: P(c/d)
  - e.g. Logistic regression, maximum entropy models, conditional random fields

Generative vs. Discriminative Models

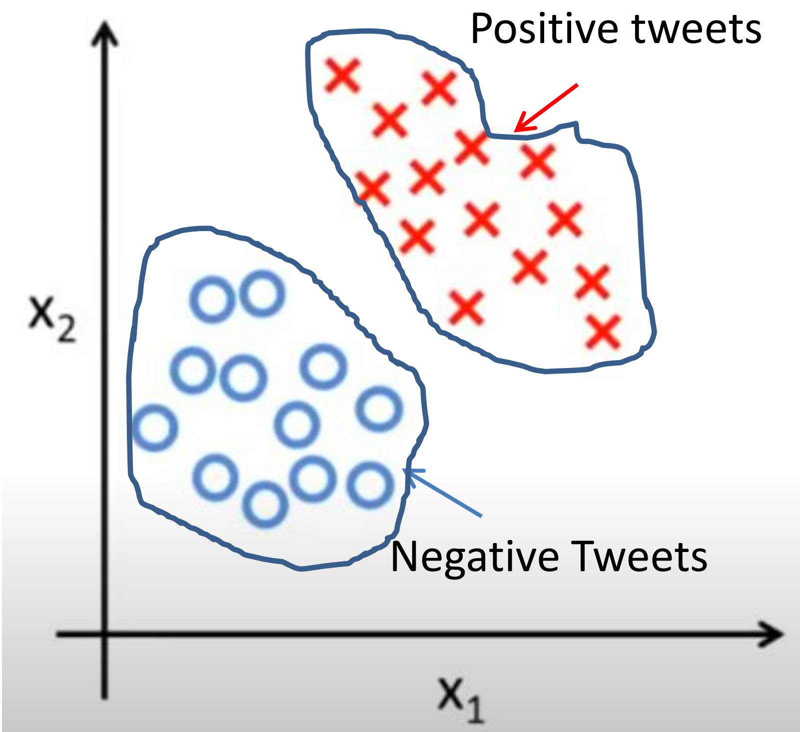


Joint vs. conditional likelihood:

- A joint model gives probabilities P(d/c) and tries to maximize this joint likelihood.
- A conditional model gives probabilities P(c/d), taking the data as given and modeling only the conditional probability of the class.

Generative (Joint) Models

Example: Naive Bayes



- In Generative Model, we have to learn p(x|y) and p(y) (class priors) i.e. p(y=negative tweets), p(y=positive)
- Generative algorithms try to learn p(x, y) which can be transformed into p(y|x) later to classify the data.

Generative (Joint) Models

- Suppose we have model: p(x/y) and p(y)
- Given new x .
- To predict class for x, we need to compute:

$$p(y=1/x) = \frac{p(x/y=1) * p(y=1)}{p(x)} \quad \text{by Bayes rule}$$

Now, we have p(x/y) and p(y) from the model and

$$p(x) = \sum_y p(x,y) = p(x/y=1)\ p(y=1) + p(x/y=0)\ p(y=0)$$

Example: Generative Model

- Suppose we have trained a generative model, and get a new test example x. Our model tells us that:

  $p(x/y=0)=0.01$

  $p(x/y=1)=0.03$

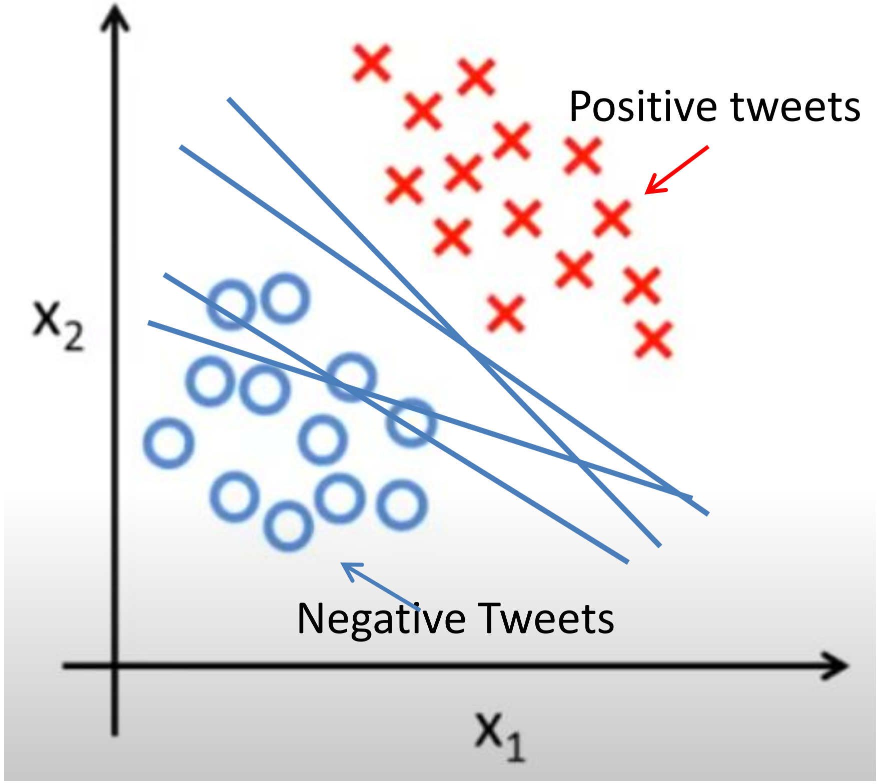
  $p(y=1)=p(y=0)= 0.5$

What is $p(y=1/x)$?

Solution:

Discriminative (Conditional) Models

Example: Logistic regression



- In Discriminative Model, it directly tries to find a straight line separating the two classes.
- Learns p(y/x) directly

Discriminative (Conditional) Models

- A discriminative algorithm does not care about how the data was generated, it simply categorizes the given data.
- So, discriminative algorithms try to learn $p(y|x)$ directly from the data and then try to classify data.
- Discriminative models do not need to model the distribution of the observed variables.

Mathematics of POS tagging

Argmax Computation

Suppose:

$$x^* = \underset{x}{\operatorname{argmax}}\,(f(x))$$

Find out value of x which maximizes f(x)

Bigram Assumption

Best tag sequence

= T*

= argmax P(T|W)

T

= argmax P(T)P(W|T)   (by Bayes Theorem)

P(T) = P(t0=^ t1t2 … tn+1= .)

= P(t0)P(t1|t0)P(t2|t1t0)P(t3|t2t1t0) …

P(tn|tn-1tn-2…t0)P(tn+1|tntn-1…t0)

= P(t0)P(t1|t0)P(t2|t1) … P(tn|tn-1)P(tn+1|tn)

Bigram Assumption

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

Lexical Probability Assumption

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

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

$$= P(w_o|t_o)P(w_1|t_1) \ldots P(w_{n+1}|t_{n+1})$$

$$= \prod_{i=0}^{n+1} P(w_i|t_i)$$

$$= \prod_{i=0}^{n+1} P(w_i|t_i) \qquad \text{(Lexical Probability Assumption)}$$

Best tag sequence

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

$$P(w_i/t_i)$$

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

Process

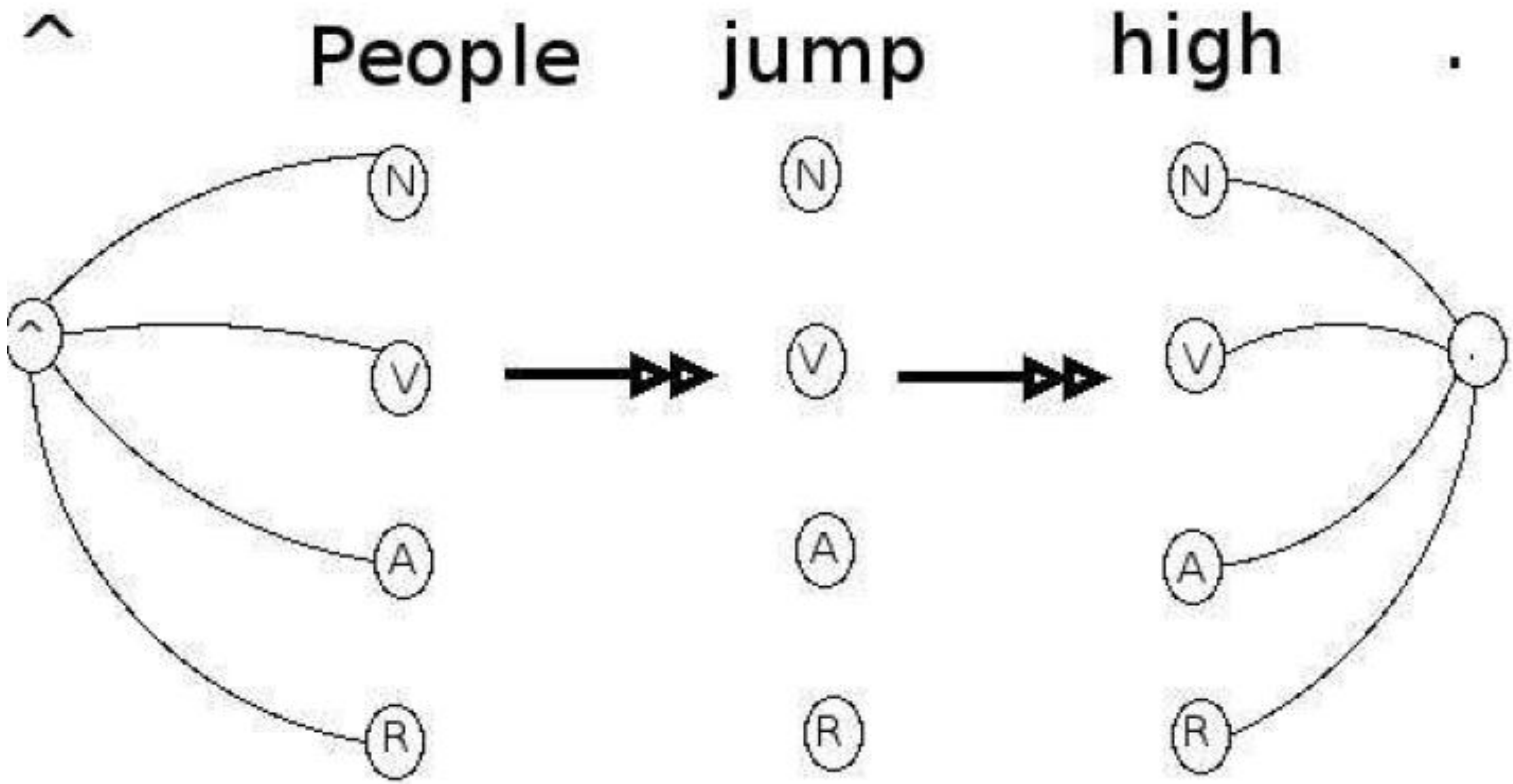
1. List all possible tag for each word in sentence.
2. Choose best suitable tag sequence.

Example

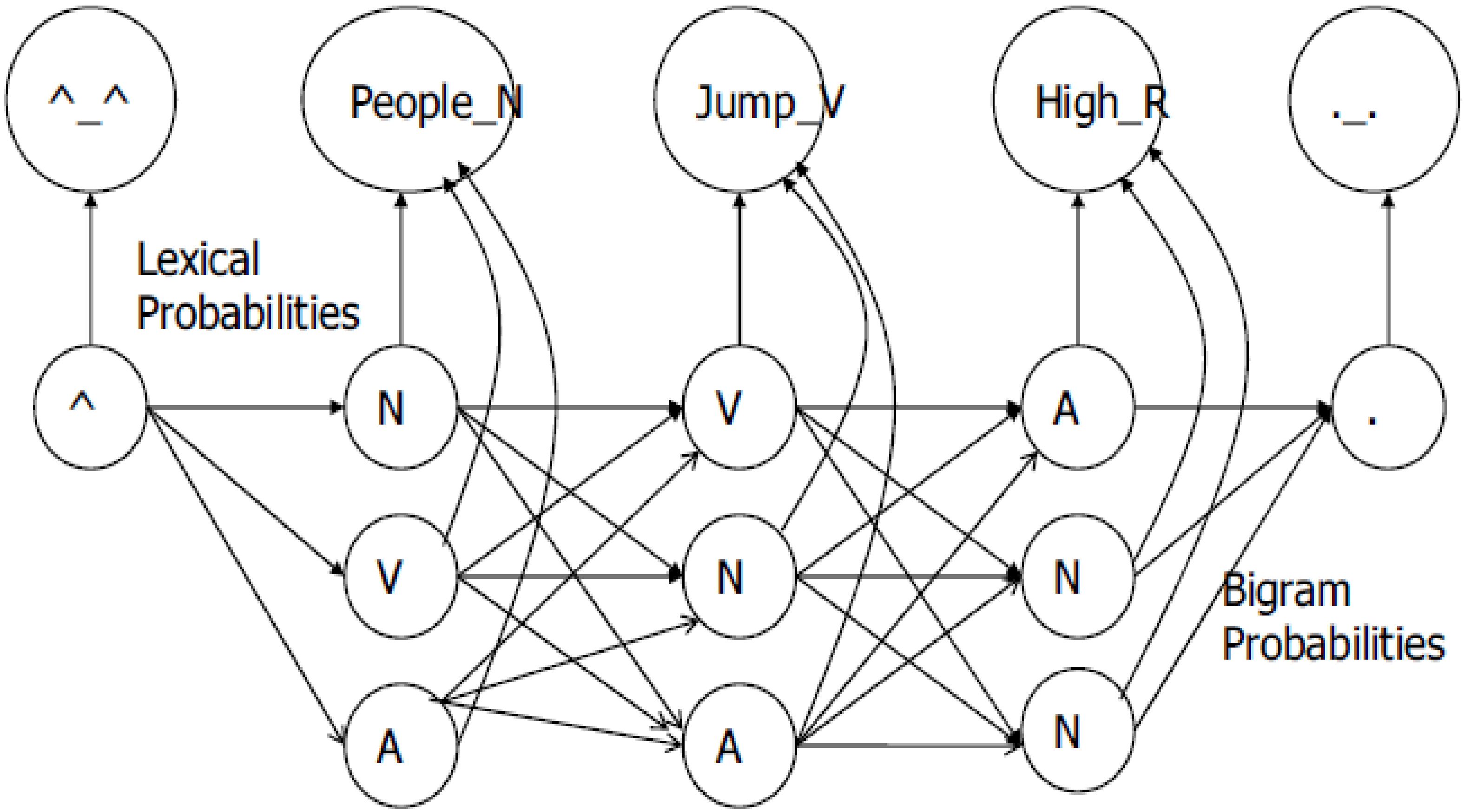
”**People jump high**”.

- People : Noun/Verb/Adjective
- jump : Noun/Verb/Adjective
- high : Noun/Verb/Adjective

Process:



Model



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

Bigram probabilities from the Corpus

Corpus contains:300 sentences & 4 categories(N,V,Art,P)

Words: 1998, Nouns: 833, Verbs: 300,

Articles: 558, Prepositions: 307

P(ART/^) = Count (^, ART) / Count(^)

Category	Count	Pair	Count	Bigram	Prob. Estimate
^	300	^, ART	213	P(ART/ ^)	0.71
^	300	^, N	87	P(N /^)	0.29
ART	558	ART, N	558	P(N /ART)	1
N	833	N, V	358	P(V /N)	0.43
N	833	N, N	108	P(N/ N)	0.13
N	833	N, P	366	P(P/ N)	0.44
V	300	V, N	75	P(N /V)	0.35
V	300	V, ART	194	P(ART/ V)	0.65
P	307	P, ART	226	P(ART /P)	0.74
P	307	P, N	81	P(N /P)	0.26

Summary of word count in corpus

	N	V	ART	P	TOTAL
flies	21	23	0	0	**44**
fruit	49	5	1	0	**55**
like	10	30	0	21	**61**
a	1	0	201	0	**202**
the	1	0	300	2	**303**
flower	53	15	0	0	**68**
flowers	42	16	0	0	**58**
birds	64	1	0	0	**65**
others	592	210	56	284	**1142**
Total	**833**	**300**	**558**	**307**	**1998**

Lexical generation Probabilities: $\mathbf{P(the/ART) = \dfrac{Count\ (the\ as\ ART)}{Count(ART)}}$

P(the/ART)	0.54	P(like/P)	0.068	P(flower/N)	0.063
P(flies/N)	0.025	P(like/N)	0.012	P(flowers/V)	0.05
P(flies/V)	0.076	P(a/ART)	0.360	P(birds/N)	0.076
P(like/V)	1	P(a/N)	0.001	P(fruit/N)	0.06

Calculation from actual data

- Corpus
  - ^ People Jump High .

Bigram probabilities

	N	**V**	**A**
N	0.2	0.7	0.1
V	0.6	0.2	0.2
A	0.5	0.2	0.3

Lexical Probability

	Noun	**Verb**	**Adjective**
People	10-5	0.4 X 10-3	10-7
Jump	10-7	10-2	10-7
high	0	0	10-1

- values in cell are P(row-heading/ col-heading)

Observations leading to why probability is needed

1. Many tasks are sequence labeling tasks
2. Tasks carried out in layers
3. Within a layer, there are limited windows of information
4. This naturally calls for strategies for dealing with uncertainty
5. Probability and Markov process give a way for dealing with uncertainty.