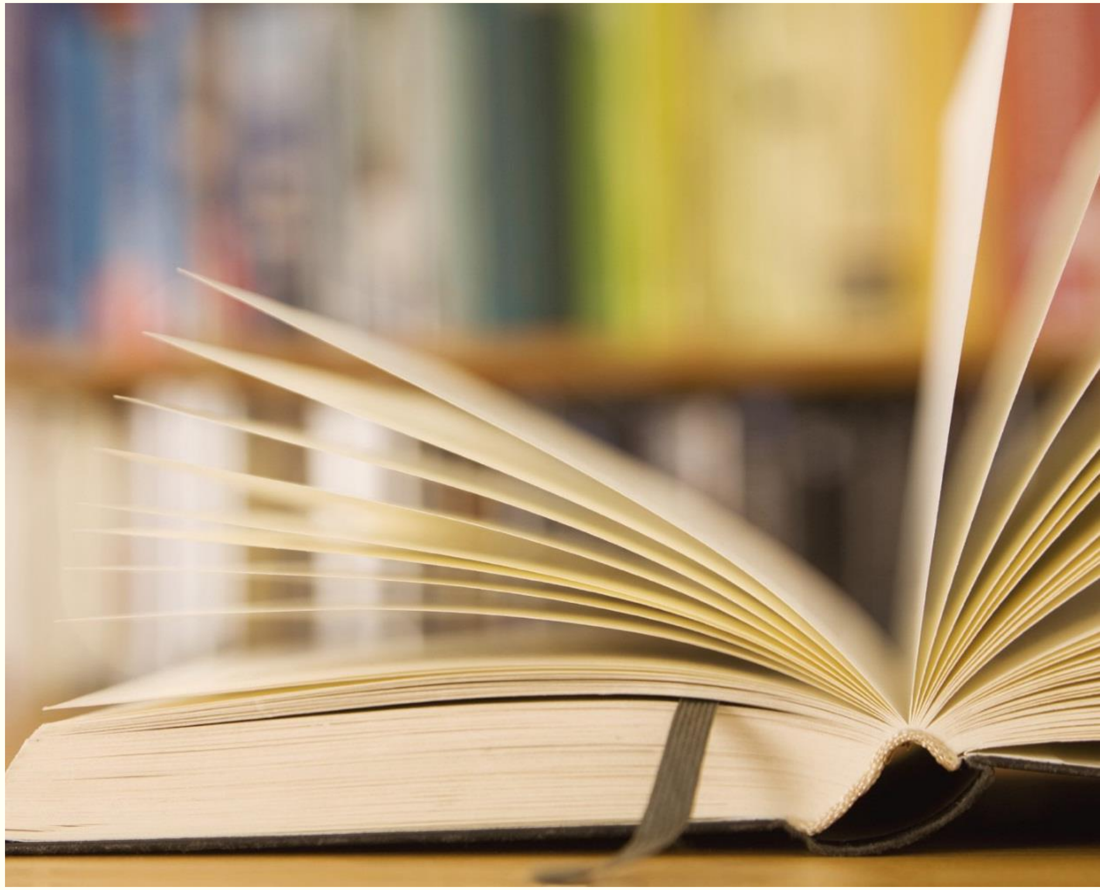
NATURAL LANGUAGE PROCESSING

Dr. G. Bharadwaja Kumar



PARTS OF SPEECH

- The parts of speech explain how a word is used in a sentence.

- Based on their usage and functionality words are categorized into several types or parts of speech.

- Words having major parts of speech contribute for meaning to a greater extent, and hence are sometimes called content words
  - Nouns, verbs, adjectives, and adverbs are content parts of speech.
- Function words are words that exist to explain or create grammatical or structural relationships into which the content words may fit.
  - Pronouns, prepositions, conjunctions, determiners, qualifiers/intensifiers, and interrogatives are some function parts of speech.

Use of POS Tagging

- Useful in
  - Information Retrieval
  - Text to Speech: ***ob***ject(N) vs. ob***ject***(V);
    discount(N) vs. dis***count***(V)
  - Word Sense Disambiguation
- Useful as a preprocessing step of parsing
  - Unique tag to each word reduces the number of parses

Nouns

- This part of a speech refers to words that are used to name persons, things, animals, places, ideas, or events.

- *Tom Hanks* is very versatile.
  - The italicized noun refers to a name of a person.
- *Dog*s can be extremely cute.
  - In this example, the italicized word is considered a noun because it names an animal.
- It is my *birthday*.
  - The word “birthday” is a noun which refers to an event.

Noun – Subcategories

- **Proper**– proper nouns always start with a capital letter and refers to specific names of persons, places, or things.
- Examples: Volkswagen Beetle, Shakey's Pizza, Game of Thrones
- **Common**– common nouns are the opposite of proper nouns. These are just generic names of persons, things, or places.
- Examples: car, pizza parlor, TV series
- **Concrete**– this kind refers to nouns which you can perceive through your five senses.
- Examples: folder, sand, board
- **Abstract**- unlike concrete nouns, abstract nouns are those which you can't perceive through your five senses.
- Examples: happiness, grudge, bravery

- **Count**– it refers to anything that is countable, and has a singular and plural form.
- Examples: kitten, video, ball
- **Mass**– this is the opposite of count nouns. Mass nouns are also called non-countable nouns, and they need to have “counters” to quantify them.
- Examples of Counters: kilo, cup, meter
- Examples of Mass Nouns: rice, flour, garter
- **Collective**– refers to a group of persons, animals, or things.
- Example: faculty (group of teachers), class (group of students), pride (group of lions)

Pronoun

- A pronoun is a part of a speech which functions as a replacement for a noun.

- Some examples of pronouns are: *I*, *it, he, she, mine, his, hers, we, they, theirs,* and *ours.*

Sample Sentences:

•Janice is a very stubborn child. *She* just stared at me and when I told her to stop.

•The largest slice is *mine*.

•*We* are number one.

Adjectives

- This part of a speech is used to describe a noun or a pronoun. Adjectives can specify the quality, the size, and the number of nouns or pronouns.

- The carvings are *intricate*.
  - The italicized word describes the appearance of the noun “carvings.”
- I have *two* hamsters.
  - The italicized word “two,” is an adjective which describes the number of the noun “hamsters.”
- Wow! That doughnut is *huge*!
  - The italicized word is an adjective which describes the size of the noun “doughnut.”

Conjuctions

- The conjunction is a part of a speech which joins words, phrases, or clauses together.
- Examples of Conjunctions: *and, yet, but, for, nor, or,* and *so*
- Sample Sentences:
  - This cup of tea is delicious *and* very soothing.
  - Kiyoko has to start all over again *because* she didn't follow the professor's instructions.
  - Homer always wanted to join the play, *but* he didn't have the guts to audition.
- The italicized words in the sentences above are some examples of conjunctions.

Verbs

- This is the most important part of a speech, for without a verb, a sentence would not exist. Simply put, this is a word that shows an action (physical or mental) or state of being of the subject in a sentence.
- Examples of "State of Being Verbs" : *am*, *is*, *was*, *are*, and *were*
- Sample Sentences:
- As usual, the Stormtroopers *missed* their shot.
  - The italicized word expresses the action of the subject "Stormtroopers."
- They are always prepared in emergencies.
  - The verb "are" refers to the state of being of the pronoun "they," which is the subject in the sentence.

Adverb

- Just like adjectives, adverbs are also used to describe words, but the difference is that adverbs describe adjectives, verbs, or another adverb.
- The different types of adverbs are:
- **Adverb of Manner**– this refers to how something happens or how an action is done.
- Example: Annie *danced* gracefully.
- The word "gracefully" tells how Annie *danced*.
- **Adverb of Tim**e- this states "when" something happens or "when" it is done.
- Example: She came *yesterday*.
- The italicized word tells when she "came."

- **Adverb of Place**– this tells something about “where” something happens or ”where” something is done.
- Example: Of course, I looked everywhere!
- The adverb “everywhere” tells where I “looked.”
- **Adverb of Degree**– this states the intensity or the degree to which a specific thing happens or is done.
- Example: The child is *very* talented.
- The italicized adverb answers the question, “To what degree is the child talented?”

Prepositions

- This part of a speech basically refers to words that specify location or a location in time.
- Examples of Prepositions: *above, below, throughout, outside, before, near,* and *since*
- Sample Sentences:
  - Micah is hiding *under* the bed.
    - The italicized preposition introduces the prepositional phrase "under the bed," and tells **where** Micah is hiding.
  - *During* the game, the audience never stopped cheering for their team.
    - The italicized preposition introduces the prepositional phrase "during the game," and tells **when** the audience cheered.

Interjections

- This part of a speech refers to words which express emotions. Since interjections are commonly used to convey strong emotions, they are usually followed by an exclamation point.
- Sample Sentences:
- Ouch! That must have hurt.
- Hurray, we won!
- Hey! I said enough!

Parts of Speech

Woodward ENGLISH

NOUN

Name of a person, place, thing or idea.

Examples: Daniel, London, table, hope

- ***Mary*** uses a blue ***pen*** for her ***notes***.

PRONOUN

A pronoun is used in place of a noun or noun phrase to avoid repetition.

Examples: I, you, it, we, us, them, those

- ***I*** want ***her*** to dance with ***me***.

ADJECTIVE

Describes, modifies or gives more information about a noun or pronoun.

Examples: cold, happy, young, two, fun

- The ***little*** girl has a ***pink*** hat.

VERB

Shows an action or a state of being.

Examples: go, speak, eat, live, are, is

- I ***listen*** to the word and then ***repeat*** it.

ADVERB

Modifies a verb, an adjective or another adverb. It tells how (often), where, when.

Examples: slowly, very, always, well, too

- ***Yesterday***, I ate my lunch ***quickly***.

PREPOSITION

Shows the relationship of a noun or pronoun to another word.

Examples: at, on, in, from, with, about

- I left my keys ***on*** the table ***for*** you.

CONJUNCTION

Joins two words, ideas, phrases together and shows how they are connected.

Examples: and, or, but, because, yet, so

- I was hot ***and*** tired ***but*** still finished it.

INTERJECTION

A word or phrase that expresses a strong emotion. It is a short exclamation.

Examples: Ouch! Hey! Oh! Watch out!

- ***Wow!*** I passed my English exam.

www.grammar.cl www.woodwardenglish.com www.vocabulary.cl

Brown/Penn Treebank tags

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 ”)
PP	Personal pronoun	*I, you, he*	(	Left parenthesis	([, (, {, <)
PP$	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*			

Example English Part-of-Speech Tagsets

- Brown corpus - 87 tags
  - Allows compound tags
    - "I'm" tagged as PPSS+BEM
      - PPSS for "non-3rd person nominative personal pronoun" and BEM for "am, 'm"
- Others have derived their work from Brown Corpus
  - LOB Corpus: 135 tags
  - Lancaster UCREL Group: 165 tags
  - London-Lund Corpus: 197 tags.
  - BNC – 61 tags (C5)
  - PTB – 45 tags
- Other languages have developed other tagsets

- Rule-Based POS tagging
  - e.g., ENGTWOL [Voutilainen, 1995]
    - large collection (> 1000) of constraints on what sequences of tags are allowable
- Transformation-based tagging
  - e.g.,Brill's tagger [Brill, 1995]
- Stochastic (Probabilistic) tagging
  - e.g., TNT [Brants, 2000]

Sample rules

N-IP rule:

A tag N (noun) cannot be followed by a tag IP (interrogative pronoun)

… man who …

- man: {N}
- who: {RP, IP} --> {RP} relative pronoun

ART-V rule:

A tag ART (article) cannot be followed by a tag V (verb)

…the book…

- the: {ART}
- book: {N, V} --> {N}

A Simple Strategy for POS Tagging

- Choose the most likely tag for each ambiguous word, independent of previous words
  - i.e., assign each token the POS category it occurred as most often in the training set
  - e.g., race – which POS is more likely in a corpus?
- This strategy gives you 90% accuracy in controlled tests
  - So, this "unigram baseline" must always be compared against

Example of the Simple Strategy

- Which POS is more likely in a corpus (1,273,000 tokens)?

	NN	VB	Total
race	400	600	1000

- P(NN|race) = P(race&NN) / P(race) by the definition of conditional probability
  - P(race) ≅ 1000/1,273,000 = .0008
  - P(race&NN) ≅ 400/1,273,000 =.0003
  - P(race&VB) ≅ 600/1,273,000 = .0005
- And so we obtain:
  - P(NN|race) = P(race&NN)/P(race) = .0003/.0008 =.375
  - P(VB|race) = P(race&VB)/P(race) = .0004/.0008 = .625

Hand-coded Rules: ENGCG System

- Uses 56,000-word lexicon which lists parts-of-speech for each word (using two-level morphology)
- Uses up to 3,744 rules, or constraints, for POS disambiguation

ADV-that rule

Given input "that" (ADV/PRON/DET/COMP)

If (+1 A/ADV/QUANT) #next word is adj, adverb, or quantifier

(+2 SENT_LIM) #and following word is a sentence boundary

(NOT -1 SVOC/A) #and the previous word is not a verb like

#*consider* which allows adjs as object complements

Then eliminate non-ADV tags

Else eliminate ADV tag

1. TBL: A Symbolic Learning Method

- A method called error-driven Transformation-Based Learning (TBL) (Brill algorithm) can be used for symbolic learning
  - The rules (actually, a sequence of rules) are learned from an annotated corpus
  - Performs about as accurately as other statistical approaches

How TBL Rules are Learned

- We will assume that we have a tagged corpus.
- Brill's TBL algorithm has three major steps.
  - Tag the corpus with the most likely tag for each (unigram model)
  - Choose a transformation that deterministically replaces an existing tag with a new tag such that the resulting tagged training corpus has the lowest error rate out of all transformations.
  - Apply the transformation to the training corpus.
- These steps are repeated until a stopping criterion is reached.
- The result (which will be our tagger) will be:
  - First tags using most-likely tags
  - Then apply the learned transformations

Brill Algorithm (More Detailed)

- 1. Label every word token with its most likely tag (based on lexical generation probabilities).
- 2. List the positions of tagging errors and their counts, by comparing with "truth" (T)
- 3. For each error position, consider each instantiation I of X, Y, and Z in Rule template.
  - If Y=T, increment improvements[I], else increment errors[I].
- 4. Pick the I which results in the greatest error reduction, and add to output
  - VB NN PREV1OR2TAG DT improves on 98 errors, but produces 18 new errors, so net decrease of 80 errors
- 5. Apply that I to corpus
- 6. Go to 2, unless stopping criterion is reached

Most likely tag:

$P(NN|race) = .98$

$P(VB|race) = .02$

Is/VBZ expected/VBN to/TO race/NN tomorrow/NN

Rule template: *Change a word from tag X to tag Y when previous tag is Z*

Rule Instantiation for above example: NN VB PREV1OR2TAG TO

Applying this rule yields:

Is/VBZ expected/VBN to/TO race/VB tomorrow/NN

Transformation Rules

Rewrite rules: what to replace

POS: $t_i \rightarrow t_j$; $* \rightarrow t_j$ *(replace tag t_i / any tag by tag t_j)*

Triggering environment: when to replace

POS:
Non-lexicalized templates:

1. The preceding (following) word is tagged t_a.
2. The word two before (after) is tagged t_a.
3. One of the two preceding (following) words is tagged t_a.
4. One of the three preceding (following) words is tagged t_a.
5. The preceding word is tagged t_a and the following word is tagged t_b.
6. The preceding (following) word is tagged t_a and the word two before (after) is tagged t_b.

Lexicalized templates:

1. The preceding (following) word is w_a.
2. The word two before (after) is w_a.
3. One of the two preceding (following) words is w_a.
4. The current word is w_a and the preceding (following) word is w_b.
5. The current word is w_a and the preceding (following) word is tagged t_a.
6. The current word is w_a.
7. The preceding (following) word is w_a and the preceding (following) tag is t_a.
8. The current word is w_a, the preceding (following) word is w_b and the preceding (following) tag is t_a.

Rules Learnt

The first rules learnt by Brill's POS tagger (with examples):

#	From	To	If
1	NN	VB	previous tag is TO
	to/TO conflict/NN→ NB		
2	VBP	VB	one of the previous 3 tags is MD
	might/MD vanish/VBP→ VB		
3	NN	VB	one of the previous two tags is MD
	might/MD not reply/NN→ VB		
4	VB	NN	one of the previous two tags is DT
	the/DT amazing play/VB→ NN		

Tagging Unknown Words

Additional rule templates use character-based cues:
Change the tag of an unknown word from X to Y if:

1. Deleting the prefix (suffix) x, $|x| \leq 4$, results in a word.
2. The first (last) 1–4 characters of the word are x.
3. Adding the character string x, $|x| \leq 4$, as a prefix (suffix) results in a word.
4. Word w appears immediately to the left (right) of the word.
5. Character z appears in the word.

Unknown Words: Rules Learnt

#	From	To	If
1	NN	NNS	has suffix **-s**
	rules/NN→ NNS		
4	NN	VBN	has suffix **-ed**
	tagged/NN→ VBN		
5	NN	VBG	has suffix **-ing**
	applying/NN→ VBG		
18	NNS	NN	has suffix **-ss**
	actress/NNS→ NN		

Strengths of transformation-based tagging

- exploits a wider range of lexical and syntactic regularities

- can look at a wider context
  - condition the tags on preceding/next words not just preceding tags.
  - can use more context than bigram or trigram.

- transformation rules are easier to understand

Stochastic POS tagging

- Assume that a word’s tag only depends on the previous tags (not following ones)
- Use a training set (manually tagged corpus) to:
  - learn the regularities of tag sequences
  - learn the possible tags for a word
  - model this info through a Markov process

Hidden Markov Model

- For Markov chains, the output symbols are the same as the states.
  - See **sunny** weather: we're in state **sunny**
- But in part-of-speech tagging (and other things)
  - The output symbols are **words**
  - But the hidden states are **part-of-speech tags**
- So we need an extension!
- A Hidden Markov Model is an extension of a Markov chain in which the output symbols are not the same as the states.
- This means we don't know which state we are in.

Hidden Markov Model (HMM) Taggers

- Goal: maximize P(word|tag) x P(tag|previous n tags)

Lexical information

Syntagmatic information

- P(word|tag)
  - word/lexical likelihood
  - probability that given this tag, we have this word
  - NOT probability that this word has this tag
  - modeled through language model (word-tag matrix)

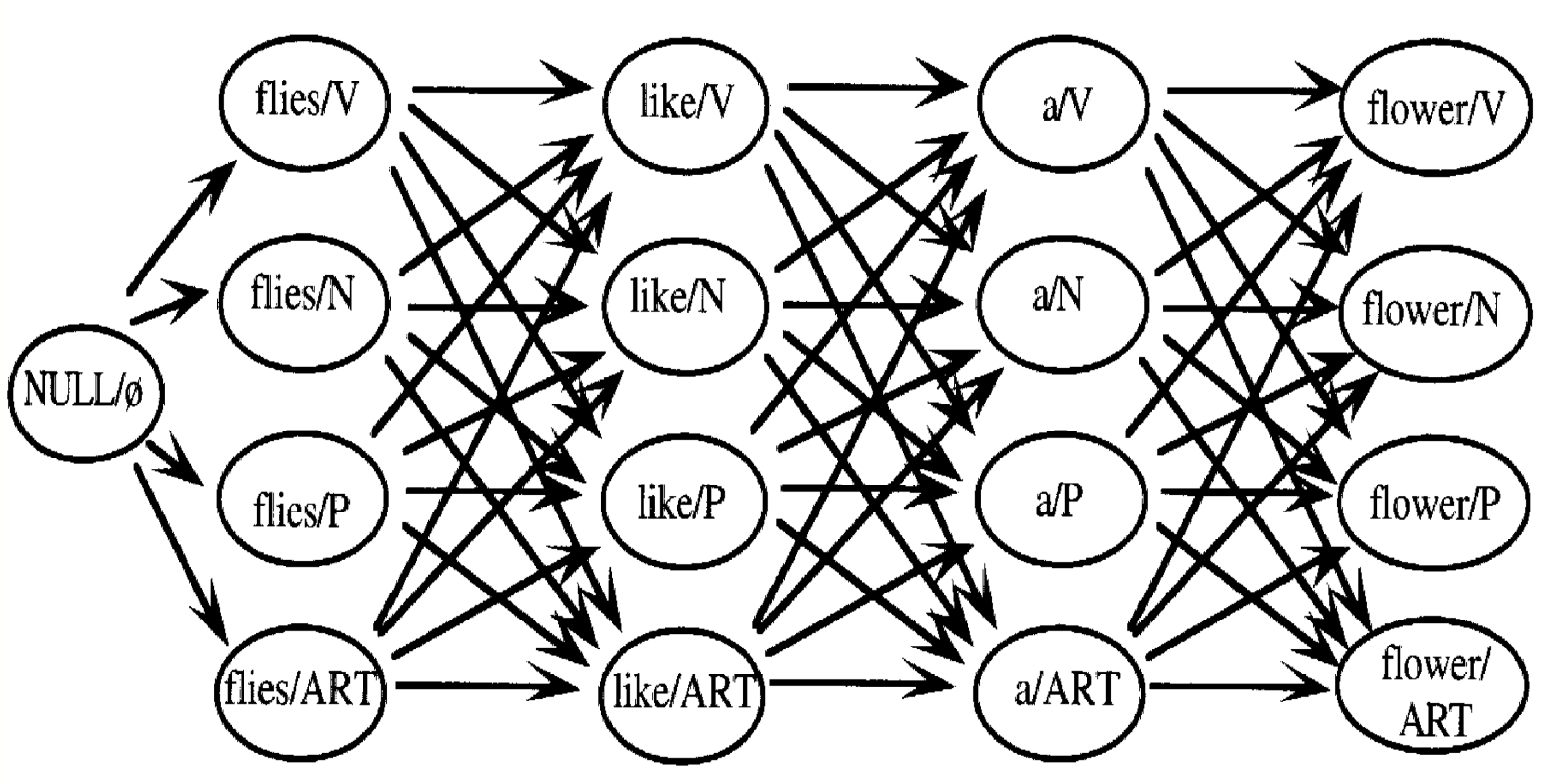
- P(tag|previous n tags)
  - tag sequence likelihood
  - probability that this tag follows these previous tags
  - modeled through language model (tag-tag matrix)

Efficient Tagging

- How to find the most likely sequence of tags for a sequence of words

- Given the contextual and lexical estimates, we can use the Viterbi algorithm to avoid using the brute force method, which for N tags and T words examines N^T sequences.

For "Flies like a flower", there are four words and four possible tags, giving 256 sequences depicted below.
In a brute force method, all of them would be examined.



Viterbi Notation

- To track the probability of the best sequence leading to each possible tag at each position, the algorithm uses δ, an N$\times$n array, where N is the number of tags and n is the number of words in the sentence. $\delta_t(t^i)$ records the probability of the best sequence up to position *t* that ends with the tag, t^i.
- To record the actual best sequence, it suffices to record only the one preceding tag for each tag and position. Hence, another array γ, an N$\times$n array, is used. $\gamma_t(t^i)$ indicates for the tag t^i in position *t* which tag at position *t-1* is in the best sequence.

Viterbi Algorithm

- Given the word sequence $w_{1,n}$, the lexical tags $t^{1,N}$, the lexical probabilities $P(w_t|t_t)$, and the bigram probabilities $P(t^i|t^j)$, find the most likely sequence of lexical tags for the word sequence.

Initialization Step:

For i= 1 to N do // For all tag states $t^{1,N}$

$\delta_1(t^i) = P(w_1|t^i) \times P(t^i|\text{ø})$

$\gamma_1(t^i) = 0$ // Starting point

Viterbi Algorithm

Iteration Step:

For f=2 to n // next word index

For i= 1 to N // tag states $t^{1,N}$

$\delta_f(t^i) = \max_{j=1,N} (\delta_{f-1}(t^j) \times P(t^i \mid t^j)) \times P(w_f \mid t^i))$

$\gamma_f(t^i) = \text{argmax}_{j=1,N} (\delta_{f-1}(t^j) \times P(t^i \mid t^j)) \times P(w_f \mid t^i))$ //index that gave max

Sequence Identification Step:

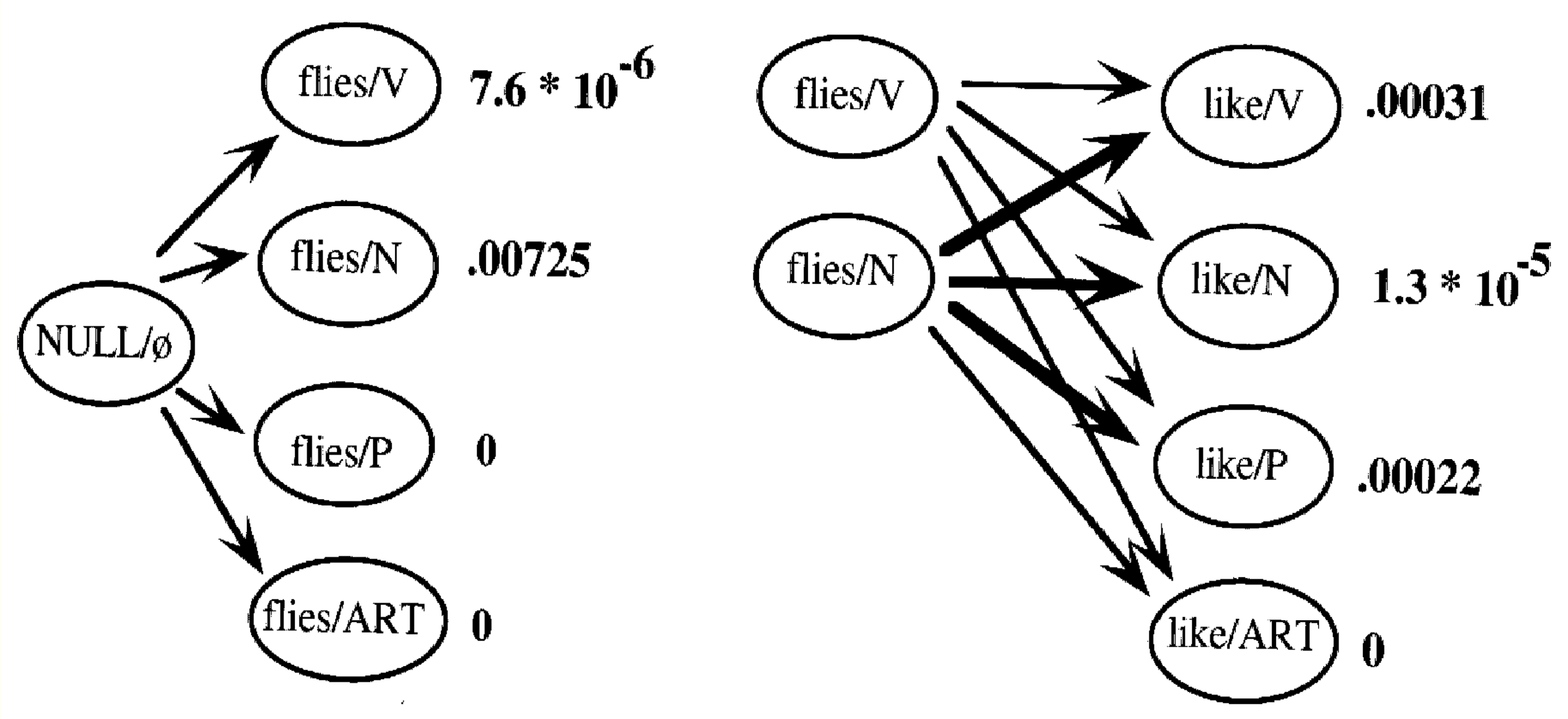
$X_n = \text{argmax}_{j=1,N}\, \delta_n(t^j)$ // Get the best ending tag state for w_n

For i = n-1 to 1 do // Get the rest

$X_i = \gamma_{i+1}(X_{i+1})$ // Use the back pointer from subsequent state

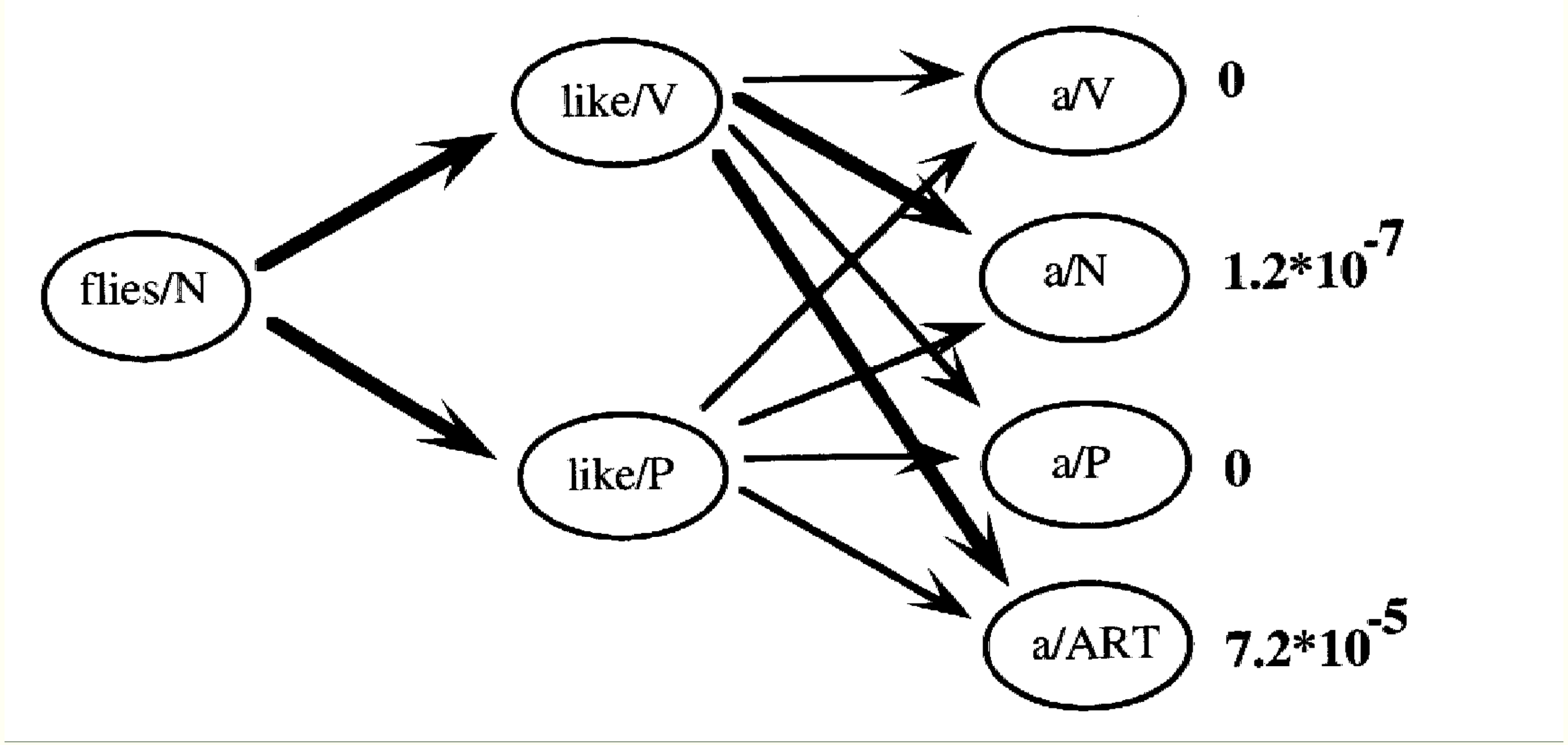
$P(X_1,\ldots, X_n) = \max_{j=1,N}\, \delta_n(t^j)$

Example

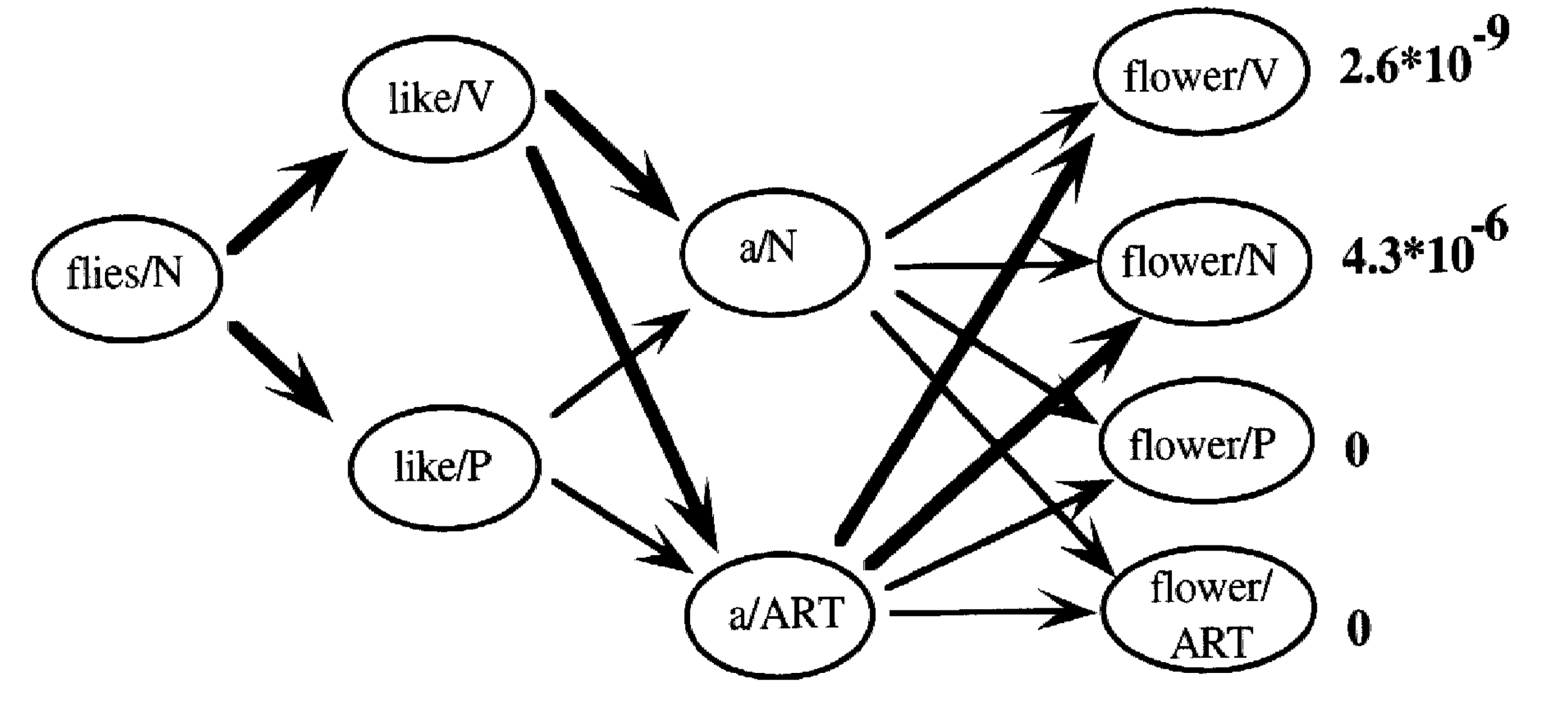


NULL/ø
flies/V
7.6 * 10-6
flies/N
.00725
flies/P
0
flies/ART
0
flies/V
flies/N
like/V
.00031
like/N
1.3 * 10-5
like/P
.00022
like/ART
0

Second Iteration Step



Final Iteration



Now we have to backtrack to get the best sequence
"Flies N like V a ART flower N"

HMM Training

- **Supervised Learning:**
  - All training sequences are completely labeled (tagged).
  - That is, nothing is really "hidden" strictly speaking.
  - Learning is very simple ➔ by **MLE estimate**
- **Unsupervised Learning**:
  - All training sequences are unlabeled (tags are unknown)
  - We do assume the number of tags, i.e. states
  - True HMM case. ➔ **Forward-Backward Algorithm**, (also known as "**Baum-Welch algorithm**") which is a special case of ***Expectation Maximization (EM)*** training

HMM Learning: Supervised

- Estimate state transition probabilities based on tag bigram and unigram statistics in the labeled data.

$$a_{ij} = \frac{C(q_t = s_i, q_{t+1} = s_j)}{C(q_t = s_i)}$$

- Estimate the observation probabilities based on tag/word co-occurrence statistics in the labeled data.

$$b_j(k) = \frac{C(q_i = s_j, o_i = v_k)}{C(q_i = s_j)}$$

- Use appropriate smoothing if training data is sparse.