UNIT 7

Word Classes and Part-of-Speech Tagging

History of Parts of Speech

- Dionysius Thrax of Alexandria (c. 100 B.C.), wrote a grammatical sketch of Greek (a “techn¯e”) which summarized the linguistic knowledge of his day.

- That work is the direct source of an astonishing proportion of our modern linguistic vocabulary, including among many other words, syntax, diphthong, clitic, and analogy.

- Also included are a description of eight parts-of-speech: noun, verb, pronoun, preposition, adverb, conjunction, participle, and article.

- Although earlier scholars had their own lists of parts-of speech, it was Thrax’s set of eight which became the basis for practically all subsequent part-of-speech descriptions of Greek, Latin, and most European languages for the next 2000 years.

History of Parts of Speech

- More recent lists of parts-of-speech (or tagsets) have many more word classes;
- 45 for the Penn Treebank (Marcus et al., 1993),
- 87 for the Brown corpus (Francis, 1979;Francis and Kuˇcera, 1982), and
- 146 for the C7 tagset (Garside et al., 1997)

Parts of Speech Tagging Task

- Input: a sequence of word tokens **w**
- Output: a sequence of part-of-speech tags **t**, one per word

Parts of Speech-Advantages in NLP

- The significance of parts-of-speech (also known as **POS, word classes, tagsets, morphological classes, or lexical tags**) are:
  1. Large amount of information they give about a word and its neighbors for language processing.
     - For example these tagsets distinguish between **possessive pronouns** (my, your, his, her, its) and **personal pronouns** (I, you, he, me).
     - Knowing whether a word is a possessive pronoun or a personal pronoun can tell us what words are likely to occur in its vicinity (possessive pronouns are likely to be followed by a noun, personal pronouns by a verb).
     - This can be useful in a language model for speech recognition.

- A word's part-of-speech can tell us something about how the word is pronounced.
  - Example - the word content, can be a noun or an adjective.

- They are pronounced differently (the noun is pronounced **CONtent** and the adjective **conTENT**).

- Thus knowing the part-of-speech can produce more natural pronunciations in a speech synthesis system and more accuracy in a speech recognition system.

- Parts-of-speech can also be used in stemming for informational retrieval (IR), since knowing a word's part-of-speech can help tell us which morphological affixes it can take

Parts-of-speech Tagging

- **Part-of-speech tagging** is a computational method for assigning parts-of-speech to words.
- Many algorithms have been applied to this problem including
  - Hand-written rules (**rule-based tagging**),
  - Probabilistic methods (**HMM tagging** and **maximum entropy tagging**), **Transformation based tagging**
  - **Memory-based tagging**

ENGLISH WORD CLASSES

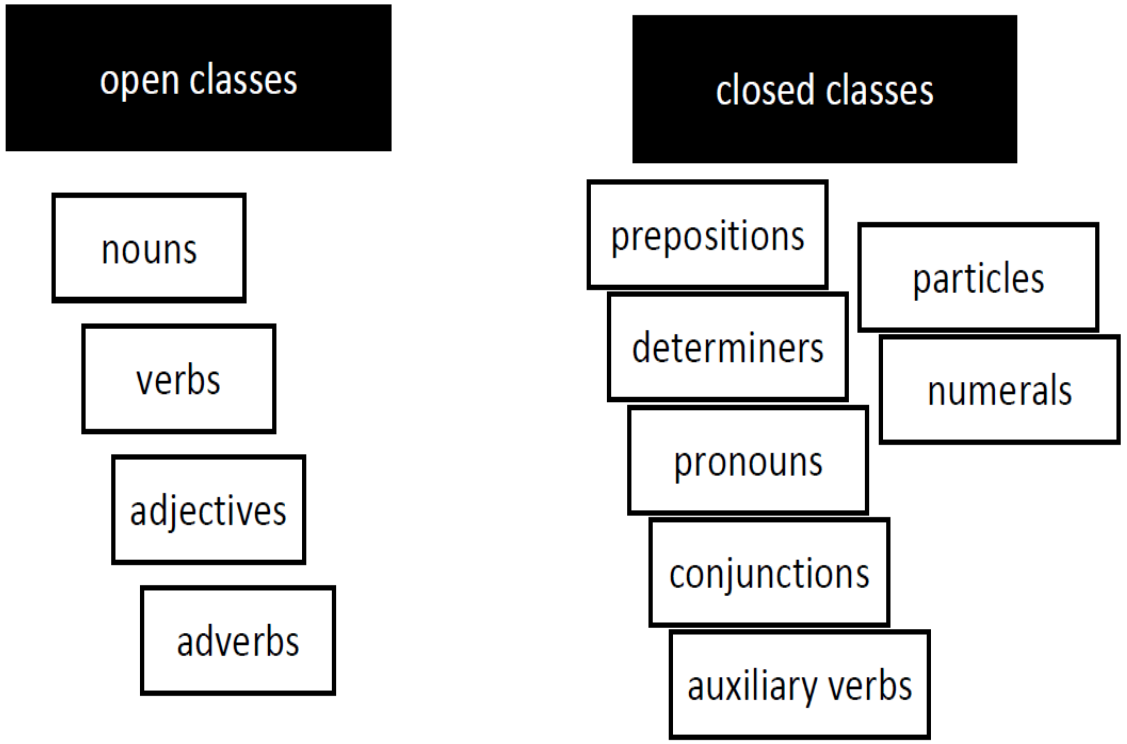
- Words that somehow ‘behave’ alike:
  - Appear in similar contexts
  - Perform similar functions in sentences
  - Undergo similar transformations

- More complete definition of POS and other classes:
  - Traditionally based on **syntactic** and **morphological** function, grouping words that have similar neighboring words (their distributional properties) or take similar affixes (their morphological properties).

  - While word classes do have **semantic tendencies**—adjectives, for example, often describe properties and nouns

English Word Class

- Parts-of-speech can be divided into two broad supercategories:
  - **CLOSED CLASS**
    - Closed classes are those that have relatively fixed membership.
    - Example: Prepositions are a closed class because there is a fixed set of them in English;
  - **OPEN CLASS**
    - Open classes continually coined or borrowed from other languages
    - **Example:** nouns and verbs are open classes because new nouns and verbs can be taken from different language.



- Closed class words are also generally **function words** like of, it, and, or you, which tend to be very short, occur frequently, and often have structuring uses in grammar.

- There are **four major open classes** that occur in the languages of the world; ***nouns, verbs, adjectives, and adverbs***

Open Word Classes

Noun

- Name given to the syntactic class in which the words for most people, places, or things occur
- Since syntactic classes like **noun** are defined syntactically and morphologically rather than semantically
  - some words for people, places, and things may not become nouns
  - Some nouns may not be words for people, places or things
- Thus, nouns include
  - Concrete terms like *ship* and *chair*,
  - Abstractions like *bandwidth* and *relationship*
  - Verb like terms like *pacing*
- Noun in English
  - Things to occur like determiners *(a goat, it's bandwidth, Plato's Republic)*
  - To take possessives *(IBM's annual revenue)*
  - To occur in plural form *(goat)*

- Nouns are grouped into **proper nouns** and **common nouns**.
  - Proper nouns
    - like *Regina*, *Colorado*, and *IBM*, are names of specific persons or entities.
    - In English, they generally aren't preceded by articles (e.g., *the book is upstairs*, but *Regina is upstairs*).
    - In written English, proper nouns are usually capitalized.
- common nouns are divided into **count nouns** and **mass nouns**.
  - **Count nouns**
    - are those that allow grammatical enumeration;
    - they can occur in both the singular and plural (*goat/goats, relationship/relationships*)
    - they can be counted (*one goat, two goats*).
  - **Mass nouns**
    - are used when something
    - is conceptualized as a homogeneous group.
    - So words like *snow, salt*, and *communism* are not counted (i.e., **two snows* or **two communisms*).
    - Mass nouns can also appear without articles where singular count nouns cannot (*Snow is white* but not **Goat is white)*

Open Verb class

- Includes most of the words referring to actions VERB and processes, including main verbs like *draw*, *provide*, *differ*, and *go*.
- A number of morphological forms (non-3rd-person-sg (*eat*), 3rd-person-sg (*eats*), progressive (*eating*), past participle (*eaten*))
- A subclass of English verbs called **auxiliaries** will be discussed when we turn to closed-class forms.

- **Adjectives**
  - Terms describing properties or qualities
  - Most languages have adjective for the concept of color (white, black), age (young, old) and value(good,bad)
  - There are languages without adjectives Example: chineese

Adverbs

- Words viewed as modifying something
  - **Directional adverbs** or **locative adverbs :** specify the direction or location of some action (*home*, *here*, *downhill*)
  - **Degree adverbs :** specify the extent of some action, process, or property (*extremely*, *very*, *somewhat*)
  - **Manner adverbs :** describe the manner of some action or process (*slowly*, *slinkily*, *delicately*)
  - **Temporal adverb :** describe the time that some action or event took place (*yesterday*, *Monday*).

Closed Word Classes

- Some important closed classes in English
  - **Prepositions:** on, under, over, near, by, at, from, to, with
  - **Determiners:** a, an, the
  - **Pronouns:** she, who, I, others
  - **Conjunctions:** and, but, or, as, if, when
  - **Auxiliary verbs:** can, may, should, are
  - **Particles:** up, down, on, off, in, out, at, by
  - **Numerals:** one, two, three, first, second, third

- **Prepositions** occur before nouns, semantically they are relational
  - Indicating spatial or temporal relations, whether literal (*on it, before then, by the house*) or metaphorical (*on time, with gusto, beside herself*)
  - Other relations as well

of	540,085	through	14,964	worth	1,563	pace	12
in	331,235	after	13,670	toward	1,390	nigh	9
for	142,421	between	13,275	plus	750	re	4
to	125,691	under	9,525	till	686	mid	3
with	124,965	per	6,515	amongst	525	o'er	2
on	109,129	among	5,090	via	351	but	0
at	100,169	within	5,030	amid	222	ere	0
by	77,794	towards	4,700	underneath	164	less	0
from	74,843	above	3,056	versus	113	midst	0
about	38,428	near	2,026	amidst	67	o'	0
than	20,210	off	1,695	sans	20	thru	0
over	18,071	past	1,575	circa	14	vice	0

Figure 5.1 Prepositions (and particles) of English from the CELEX on-line dictionary. Frequency counts are from the COBUILD 16 million word corpus.

- A **particle** is a word that resembles a preposition or an adverb, and that often combines with a verb to form a larger unit call a **phrasal verb**

  So I *went on* for some days cutting and hewing timber …

  Moral reform is the effort to *throw off* sleep …

aboard	aside	besides	forward(s)	opposite	through
about	astray	between	home	out	throughout
above	away	beyond	in	outside	together
across	back	by	inside	over	under
ahead	before	close	instead	overhead	underneath
alongside	behind	down	near	past	up
apart	below	east, etc.	off	round	within
around	beneath	eastward(s),etc.	on	since	without

Figure 5.2 English single-word particles from Quirk et al. (1985).

A closed class that occurs with nouns, often marking the beginning of a noun phrase, is the **determiners**.

One small subtype of determiners is the **articles**:

- English has three: *a, an,* and *the*
  - Articles begin a noun phrase.
  - Articles are frequent in English.
- **Conjunctions** are used to join two phrases, clauses, or sentences.
  - *and, or,* or, *but*
  - Subordinating conjunctions are used when one of the elements is of some sort of embedded status. *I thought* ***that*** *you might like some milk..* **complementizer**

- **Pronouns** act as a kind of shorthand for referring to some noun phrase or entity or event.
  - **Personal pronouns:** persons or entities (*you, she, I, it, me, etc*)
  - **Possessive pronouns:** forms of personal pronouns indicating actual possession or just an abstract relation between the person and some objects.
  - **Wh-pronouns:** used in certain question forms, or may act as complementizer.

it	199,920	how	13,137	yourself	2,437	no one	106
I	198,139	another	12,551	why	2,220	wherein	58
he	158,366	where	11,857	little	2,089	double	39
you	128,688	same	11,841	none	1,992	thine	30
his	99,820	something	11,754	nobody	1,684	summat	22
they	88,416	each	11,320	further	1,666	suchlike	18
this	84,927	both	10,930	everybody	1,474	fewest	15
that	82,603	last	10,816	ourselves	1,428	thyself	14
she	73,966	every	9,788	mine	1,426	whomever	11
her	69,004	himself	9,113	somebody	1,322	whosoever	10
we	64,846	nothing	9,026	former	1,177	whomsoever	8
all	61,767	when	8,336	past	984	wherefore	6
which	61,399	one	7,423	plenty	940	whereat	5
their	51,922	much	7,237	either	848	whatsoever	4
what	50,116	anything	6,937	yours	826	whereon	2
my	46,791	next	6,047	neither	618	whoso	2
him	45,024	themselves	5,990	fewer	536	aught	1
me	43,071	most	5,115	hers	482	howsoever	1
who	42,881	itself	5,032	ours	458	thrice	1
them	42,099	myself	4,819	whoever	391	wheresoever	1
no	33,458	everything	4,662	least	386	you-all	1
some	32,863	several	4,306	twice	382	additional	0
other	29,391	less	4,278	theirs	303	anybody	0
your	28,923	herself	4,016	wherever	289	each other	0
its	27,783	whose	4,005	oneself	239	once	0
our	23,029	someone	3,755	thou	229	one another	0
these	22,697	certain	3,345	'un	227	overmuch	0
any	22,666	anyone	3,318	ye	192	such and such	0
more	21,873	whom	3,229	thy	191	whate'er	0
many	17,343	enough	3,197	whereby	176	whenever	0
such	16,880	half	3,065	thee	166	whereof	0
those	15,819	few	2,933	yourselves	148	whereto	0
own	15,741	everyone	2,812	latter	142	whereunto	0
us	15,724	whatever	2,571	whichever	121	whichsoever	0

Pronouns of English from the CELEX on-line dictionary.

- **Auxiliary verbs:** mark certain semantic feature of a main verb, including
  - whether an action takes place in the present, past or future (tense),
  - whether it is completed (aspect),
  - whether it is negated (polarity), and
  - whether an action is necessary, possible, suggested, desired, etc (mood).
  - Including **copula** verb *be*, the two verbs *do* and *have* along with their inflection forms, as well as a class of **modal verbs.**

can	70,930	might	5,580	shouldn't	858
will	69,206	couldn't	4,265	mustn't	332
may	25,802	shall	4,118	'll	175
would	18,448	wouldn't	3,548	needn't	148
should	17,760	won't	3,100	mightn't	68
must	16,520	'd	2,299	oughtn't	44
need	9,955	ought	1,845	mayn't	3
can't	6,375	will	862	dare	??
have	???				

English modal verbs from the CELEX on-line dictionary.

Tagsets for English

- There are various standard tagsets to choose from; some have a lot more tags than others
- The choice of tagset is based on the application
- Accurate tagging can be done with even large tagsets

  - **Example**-Small 45-tag Penn Treebank Tagset, The Penn Treebank tagset was culled from the original 87-tag tagset for the Brown corpus.

  - This reduced set leaves out information that can be recovered from the identity of the lexical item.

  - This tagset has been used to label a wide variety of corpora, including the Brown corpus, the Wall Street Journal corpus, and the Switchboard corpus.

Tagsets for English

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	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, sing.	*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*			

Figure:Penn Treebank Part-of-Speech Tags (including punctuation)

Tagsets for English

- Using Penn Treebank tags, tag the following sentence from the Brown Corpus:

- The grand jury commented on a number of other topics.

  The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

- There are 70 children there

  There/EX are/VBP 70/CD children/NNS there/RB

- Preliminary findings were reported in today's New England Journal of Medicine.

  Preliminary/JJ findings/NNS were/VBD reported/VBN in/IN today/NN 's/POS New/NNP England/NNP Journal/NNP of/IN Medicine/NNP ./.

Tagsets for English

- How do tagsets differ?
  - Degree of granularity
  - Idiosyncratic decisions,
    - Penn Treebank doesn't distinguish **to/Prep** from **to/Inf**
    - eg. **I/PP want/VBP to/TO go/VB to/TO Zanzibar/NNP ./.**
  - Don't tag it if you can recover from a word

Part-of-speech tagging

- Part-of-speech tagging is the process of assigning a part-of-speech to each word in tagging a text.

- The input is a sequence of (tokenized) words and a tagset, and $\mathbf{x_1;x_2; :; :; :; x_n}$

- The output is a sequence $\mathbf{y_1;y_2; : ; : ; : ;y_n}$ of tags, each output $\mathbf{y_i}$ corresponding exactly to one input $\mathbf{x_i}$.

- Tagging: Process of assigning a part-of-speech or other syntactic class marker to each word in a corpus

Parts of Speech Tagging

- The input to a tagging algorithm is a string of words and a specified **tagset** of the kind described previously.

```
VB    DT   NN .
Book that flight .

VBZ DT  NN   VB    NN  ?
Does that flight serve dinner ?
```

- Automatically assigning a tag to a word is not trivial
  - For example, *book* is **ambiguous:** it can be a verb or a noun
  - Similarly, *that* can be a determiner, or a complementizer
- The problem of POS-tagging is to **resolve** the ambiguities, choosing the proper tag for the context.

Ambiguity in the Brown Corpus

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")

Why is Part of Speech Tagging harder?

Words often have more than one POS: 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

- Some of the most ambiguous frequent words are **that, back, down**, **put** and **set**

- Examples of the **6** different parts-of-speech for the word **back**:
  - earnings growth took a **back/JJ** seat
  - a small building in the **back/NN**
  - a clear majority of senators **back/VBP** the bill
  - Dave began to **back/VB** toward the door
  - enable the country to buy **back/RP** about debt
  - I was twenty-one **back/RB** then

- Most words in English are unambiguous;
  - i.e., they have only a single tag
- But many of the most common words of English are ambiguous
  - Example ***can*** can be an auxiliary **('to be able')**, a noun ('a metal container'), or a verb ('to put something in such a metal container'))

- The problem of POS-tagging is to **resolve** these ambiguities, choosing the proper tag for the context.
- Part-of-speech tagging is thus one of the many disambiguation tasks.

Approaches to POS-Tagging

- Most tagging algorithms fall into one of two classes
  - **Rule-based taggers**
    - Involve a large database of hand-written disambiguation rules to tag input sequences
    - Example, rules like : an ambiguous word is a noun rather than a verb if it follows a determiner.
    - **EngCG Tagger** is based on the Constraint Grammar architecture of Karlsson et. al. (1995)
  - **Stochastic taggers**
    - Resolve tagging ambiguities by using a training corpus to compute the probability of a given word having a given tag in a given context.
    - Example: **Hidden Markov model or HMM Tagger**
  - **Hybrid systems** (e.g. **Brill's transformation-based learning**)
    - Share features of both tagging architectures.
    - Like the rule-based tagger, it is based on rules which determine when an ambiguous word should have a given tag. Like the stochastic taggers,it has a machine-learning component: the rules are automatically induced from a previously tagged training corpus.

Rule-based Part-of-Speech Tagging

- The earliest algorithms for automatically assigning part-of-speech were based on a two stage architecture (Harris, 1962; Klein and Simmons, 1963; Greene and Rubin, 1971).

- The first stage used a dictionary to assign each word a list of potential parts-of-speech.

- The second stage used large lists of hand-written disambiguation rules to window down this list to a single part-of-speech for each word.

EngCG (English Constraint Grammer) Tagger (Voutilainen, ENGCG 1995, 1999)

- The EngCG lexicon is based on the **two-level morphology**
- It has about 56,000 entries for English word stems ,counting a word with multiple parts-of-speech (e.g., nominal and verbal senses of hit) as separate entries, and not counting inflected and many derived forms
- Each entry is annotated with a set of morphological and syntactic features
- Fig. in the next slide shows some selected words, together with a slightly simplified listing of their features; these features are used in rule writing.

EngCG ENGTWOL

Word	POS	Additional POS features
smaller	ADJ	COMPARATIVE
entire	ADJ	ABSOLUTE ATTRIBUTIVE
fast	ADV	SUPERLATIVE
that	DET	CENTRAL DEMONSTRATIVE SG
all	DET	PREDETERMINER SG/PL QUANTIFIER
dog's	N	GENITIVE SG
furniture	N	NOMINATIVE SG NOINDEFDETERMINER
one-third	NUM	SG
she	PRON	PERSONAL FEMININE NOMINATIVE SG3
show	V	PRESENT -SG3 VFIN
show	N	NOMINATIVE SG
shown	PCP2	SVOO SVO SV
occurred	PCP2	SV
occurred	V	PAST VFIN SV

Figure 5.11 Sample lexical entries from the ENGTWOL lexicon described in Voutilainen (1995) and Heikkilä (1995).

- **SG** for singular
- **SG3** for other than third-person-singular
- **ABSOLUTE** means non-comparative and non-superlative for an adjective
- **NOMINATIVE** just means non-genitive
- **PCP2** means past participle
- **PRE, CENTRAL**, and **POST** are ordering slots for determiners
  - Eg: (predeterminers (all) come before determiners (the): all the president's men)
- **NOINDEFDETERMINER** means that words like furniture do not appear with the indefinite determiner **a**
- **SV, SVO**, and **SVOO** specify the subcategorization or complementation pattern for the verb

- In the first stage of tagger,
  - each word is run through the two-level lexicon transducer and
  - the entries for all possible POS are returned.
- A set of about 1,100 constraints are then applied to the input sentences to rule out incorrect POS.

```
Pavlov      PALOV N NOM SG PROPER
had         HAVE V PAST VFIN SVO
            HAVE PCP2 SVO
shown       SHOW PCP2 SVOO SVO SV
that        ADV
            PRON DEM SG
            DET CENTRAL DEM SG
            CS
salivation  N NOM SG
…
```

- The boldfaced entries in the table above show the desired result, in which the simple past tense tag (rather than the past participle tag) is applied to **had,** and the complementizer (CS) tag is applied to **that**

- The constraints are used in a negative way, to eliminate tags that are inconsistent with the context

- A simplified version of the constraint:

ADVERBIAL-THAT RULE
Given input: "that"
if
(+1 A/ADV/QUANT); /* *if next word is adj, adverb, or quantifier* */
(+2 SENT-LIM); /* *and following which is a sentence boundary,* */
(NOT -1 SVOC/A); /* *and the previous word is not a verb like* */
/* *'consider' which allows adj as object complements* */
then eliminate non-ADV tags
else eliminate ADV tags

- The first two clauses of this rule check to see that the ***that*** directly precedes a sentence-final adjective, adverb, or quantifier

- In all other cases the adverb reading is eliminated

- The last clause eliminates cases preceded by verbs like ***consider*** or ***believe*** which can take a noun and an adjective; this is to avoid tagging the following instance of ***that*** as an adverb:

  Ex: I consider that odd.

- Another rule is used to express the constraint that the complementizer sense of ***that*** is most likely to be used if the previous word is a verb that expects a complement (like ***believe***, ***think***, or ***show***), and if ***that*** is followed by the beginning of a noun phrase, and a finite verb.

Stochastic Tagger

HMM (Hidden Markov Model) is a Stochastic technique for POS tagging

To find **the most probable tag sequence** given the observation sequence of **n** words $\mathbf{w_1^n}$, that is, find $\mathbf{P(t_1^n|w_1^n)}$ is highest.

But $P(t_1^n|w_1^n)$ is difficult to compute and Bayesian classification rule is used:

$$\mathbf{P(x|y) = P(x)\ P(y|x)\ /\ P(y)}$$

When applied to the sequence of words, **the most probable tag sequence** would be

$$\mathbf{P(t_1^n|w_1^n) = P(t_1^n)\ P(w_1^n|t_1^n)/P(w_1^n)}$$

where $P(w_1^n)$ does not change and thus do not need to be calculated

Thus, the **most probable tag sequence** is the product of two probabilites for each possible sequence:

- **Prior probability of the tag sequence. Context $\mathbf{P(t_1^n)}$**
- **Likelihood of the sequence of words considering a sequence of (hidden) tags. $\mathbf{P(w_1^n|t_1^n)}$**

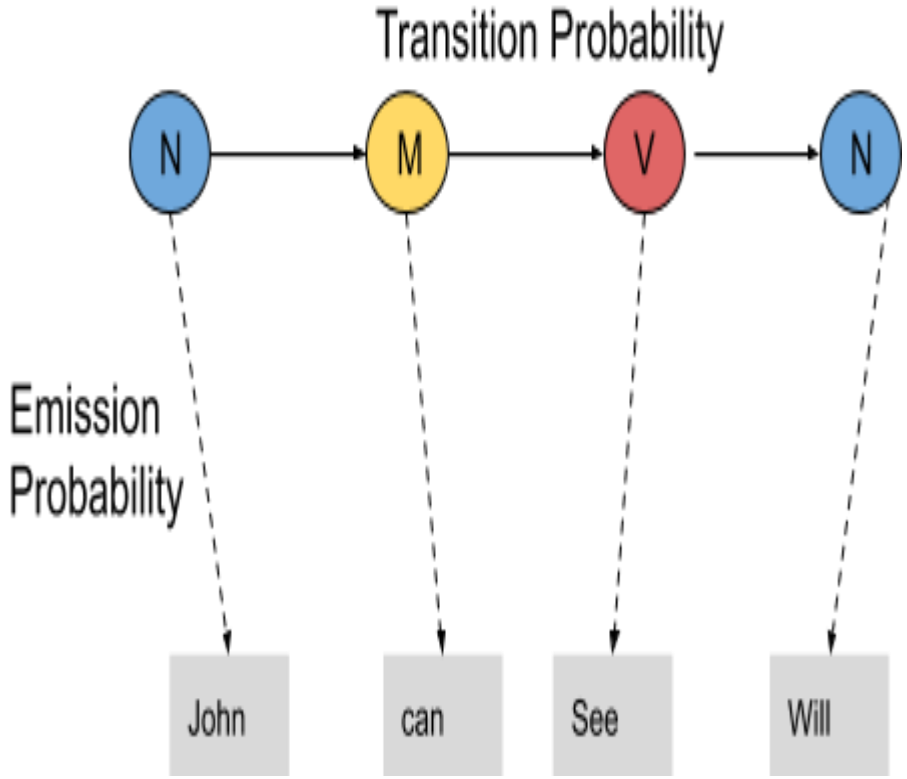
Hidden Markov Model has the following values:

- **States**: Hidden States – POS Tags; i.e. VP, NP, JJ, DT
- **Emission Probability**: $P(w_i \mid t_i)$ – Probability that given a tag $\boldsymbol{t_i}$, the word is $\boldsymbol{w_i}$

  e.g. P(book| NP) is the probability that the word ***book*** is a **Noun**.
- **Transition Probability Matrix**: $P(t_{i+1} \mid t_i)$ – Transition Probabilities from one tag t_i to another t_{i+1}

  e.g. P(VP | NP) is the probability that current tag is **Verb** given previous tag is a **Noun**.
- **Observation**: Words
- **Initial Probability**: Probability of the initial word

Stochastic Tagger-An Example proposed by Dr. Luis Serrano



- In this example, we consider only 3 POS tags that are noun, model and verb
- Let the sentence " **Ted will spot Will** " be tagged as noun, model, verb and a noun
- To calculate the probability associated with this particular sequence of tags we require their **Transition probability** and **Emission probability**

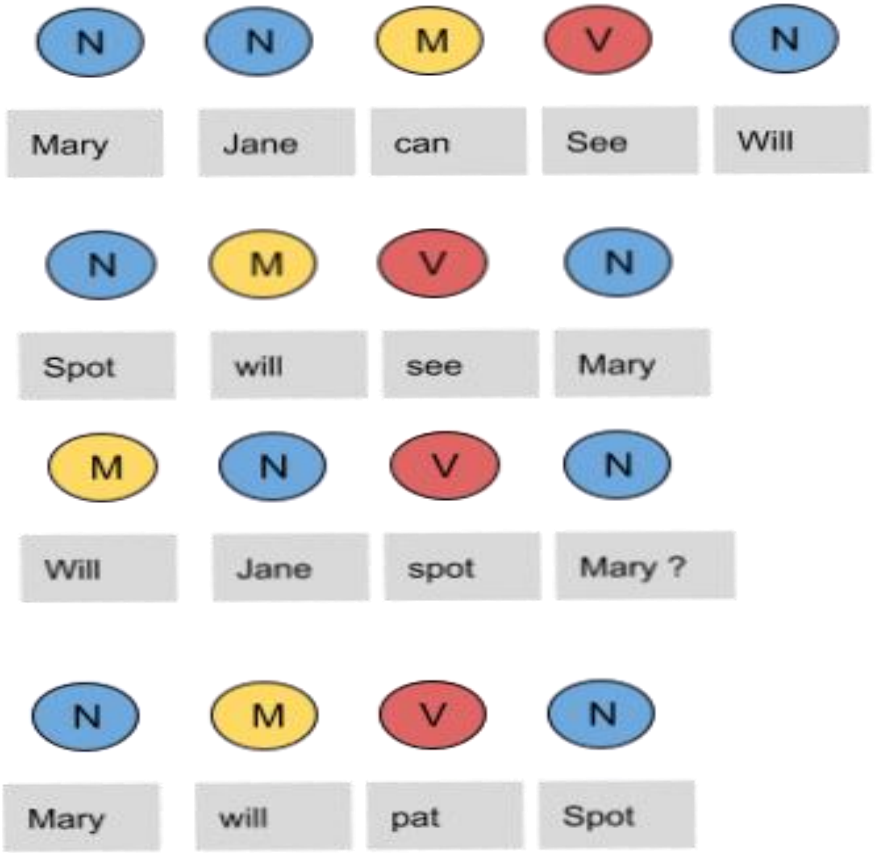
Example

- The **transition probability** is the likelihood of a particular sequence for example, **how likely is that a noun is followed by a model and a model by a verb and a verb by a noun**
- It should be high for a particular sequence to be correct.
- Now, what is the probability that the word **Ted** is a noun, **will** is a model, **spot** is a verb and **Will** is a noun
- These sets of probabilities are **Emission probabilities** and should be high for our tagging to be likely

Example

- Calculate the Emission and Transitional probabilities for the set of sentences below

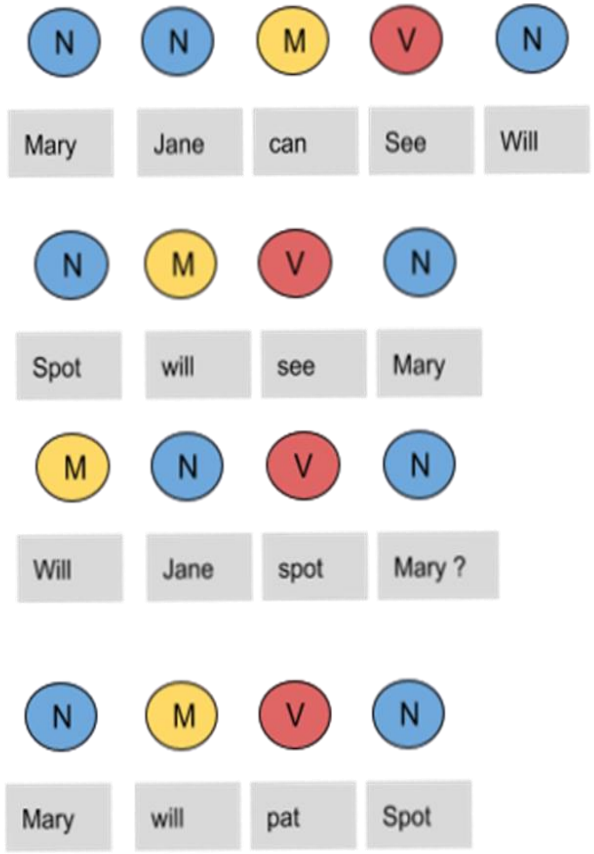
- **Mary Jane can see Will**
- **Spot will see Mary**
- **Will Jane spot Mary?**
- **Mary will pat Spot**



Example

- To calculate the **emission probabilities**, create a **counting table** in a similar manner.

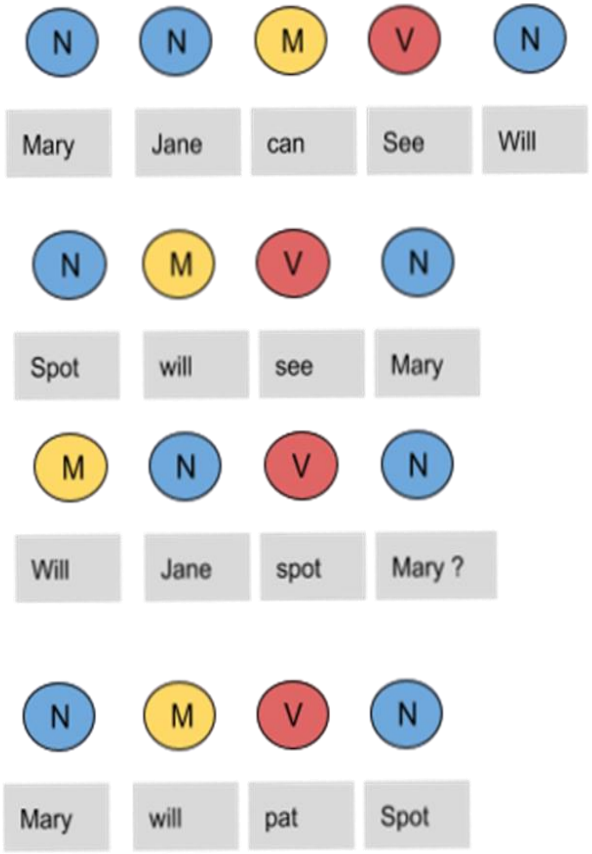
Words	Noun	Model	Verb
Mary	4	0	0
Jane	2	0	0
Will	1	3	0
Spot	2	0	1
Can	0	1	0
See	0	0	2
pat	0	0	1



Example

Divide each column by the total number of their appearances
For example, '**noun**' appears **nine times** in the above sentences so **divide** each term by **9 in the noun column**. We get the following table after this operation.

Words	Noun	Model	Verb
Mary	4/9	0	0
Jane	2/9	0	0
Will	1/9	3/4	0
Spot	2/9	0	1/4
Can	0	1/4	0
See	0	0	2/4
pat	0	0	1



Example

From the above table, it is inferred as:

- The probability that Mary is Noun = 4/9
- The probability that Mary is Model = 0
- The probability that Will is Noun = 1/9
- The probability that Will is Model = 3/4

Example

- Next, we have to calculate the transition probabilities, so define two more tags <S> and <E>.

Example

- Create a table and fill it with the co-occurrence counts of the tags.

	N	M	V	<E>
<S>	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0

Example

- In the above figure, we can see that the **<S>** tag is followed by the **N** tag **three** times, thus the first entry is **3**

- The **model** tag follows the **<S>** just once, thus the second entry is **1**

- In a similar manner, the rest of the table is filled.

- Next, we divide each term in a row of the table by the **total number of co-occurrences of the tag in consideration**
  - For example, The **Model tag** is followed by any other tag **four times** as shown below, thus we **divide each element in the third row by four**.

Example


<S>
N
N
M
V
N
<E>
Mary
Jane
can
See
Will
<S>
N
M
V
N
<E>
Spot
will
see
Mary
<S>
M
N
V
N
<E>
Will
Jane
spot
Mary ?
<S>
N
M
V
N
<E>
Mary
will
pat
Spot

Example

	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	3/9	1/9	4/9
M	1/4	0	3/4	0
V	4/4	0	0	0

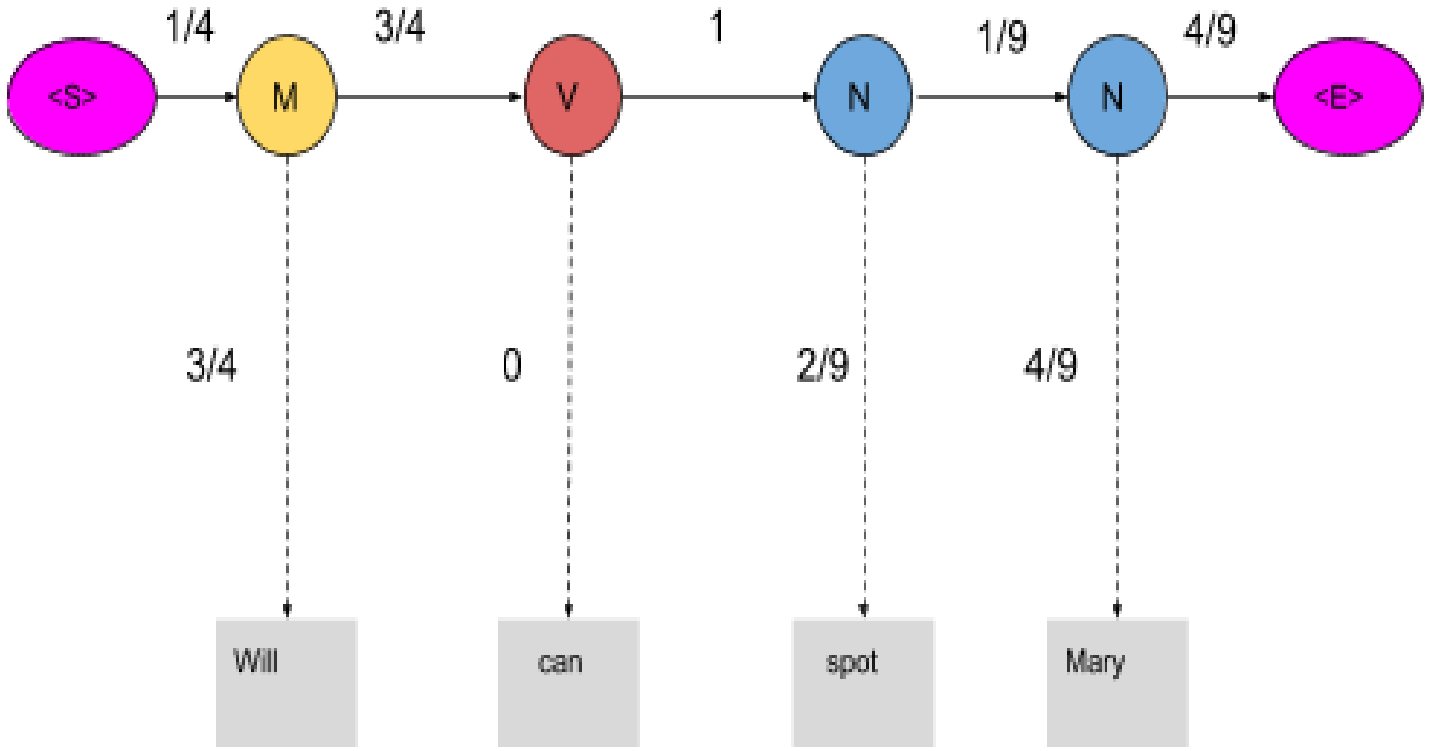
- These are the respective transition probabilities for the above four sentences

Example

- Now how does the HMM determine the **appropriate sequence of tags for a particular sentence from the above tables**? Let us find it out.

- Take a new sentence and tag them with wrong tags.
  Let the sentence, ‘ **Will can spot Mary**’ be tagged as-

  - Will as a modal
  - Can as a verb
  - Spot as a noun
  - Mary as a noun



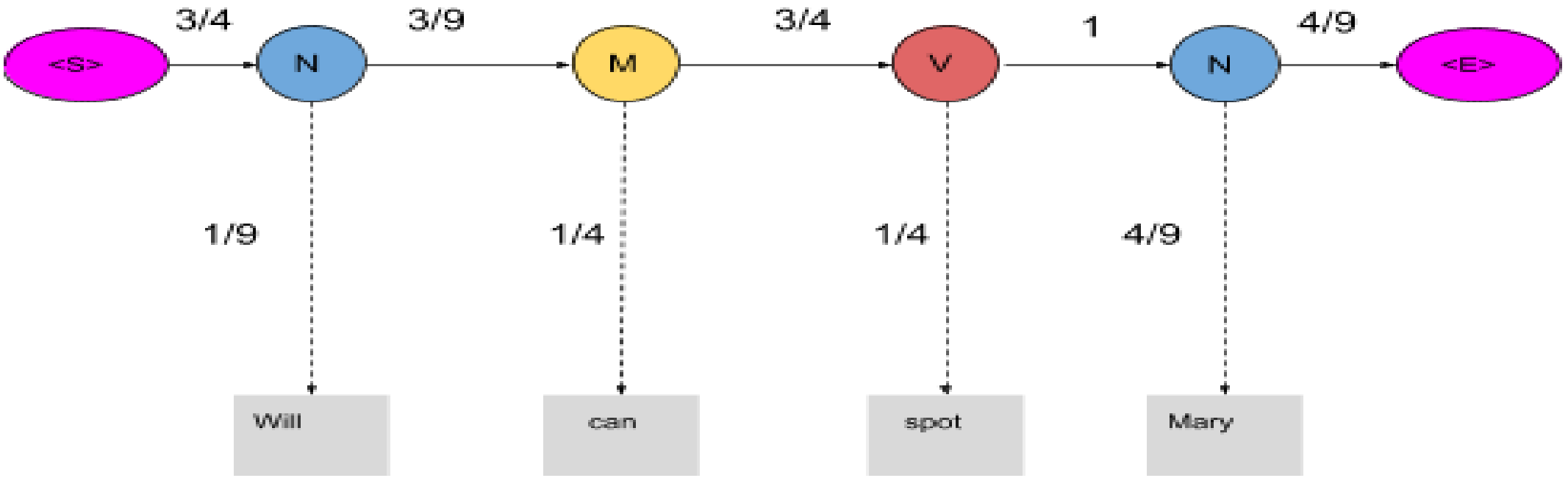
Words	Noun	Model	Verb
Mary	4/9	0	0
Jane	2/9	0	0
Will	1/9	3/4	0
Spot	2/9	0	1/4
Can	0	1/4	0
See	0	0	2/4
pat	0	0	1

- Now the product of these probabilities is the likelihood that this sequence is right
- Since the tags are not correct, the product is zero.
- **1/4*3/4*3/4*0*1*2/9*1/9*4/9*4/9=0**

	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	3/9	1/9	4/9
M	1/4	0	3/4	0
V	4/4	0	0	0

Example

- When these words are correctly tagged, we get a probability greater than zero as shown below:



•Calculating the product of these terms we get,

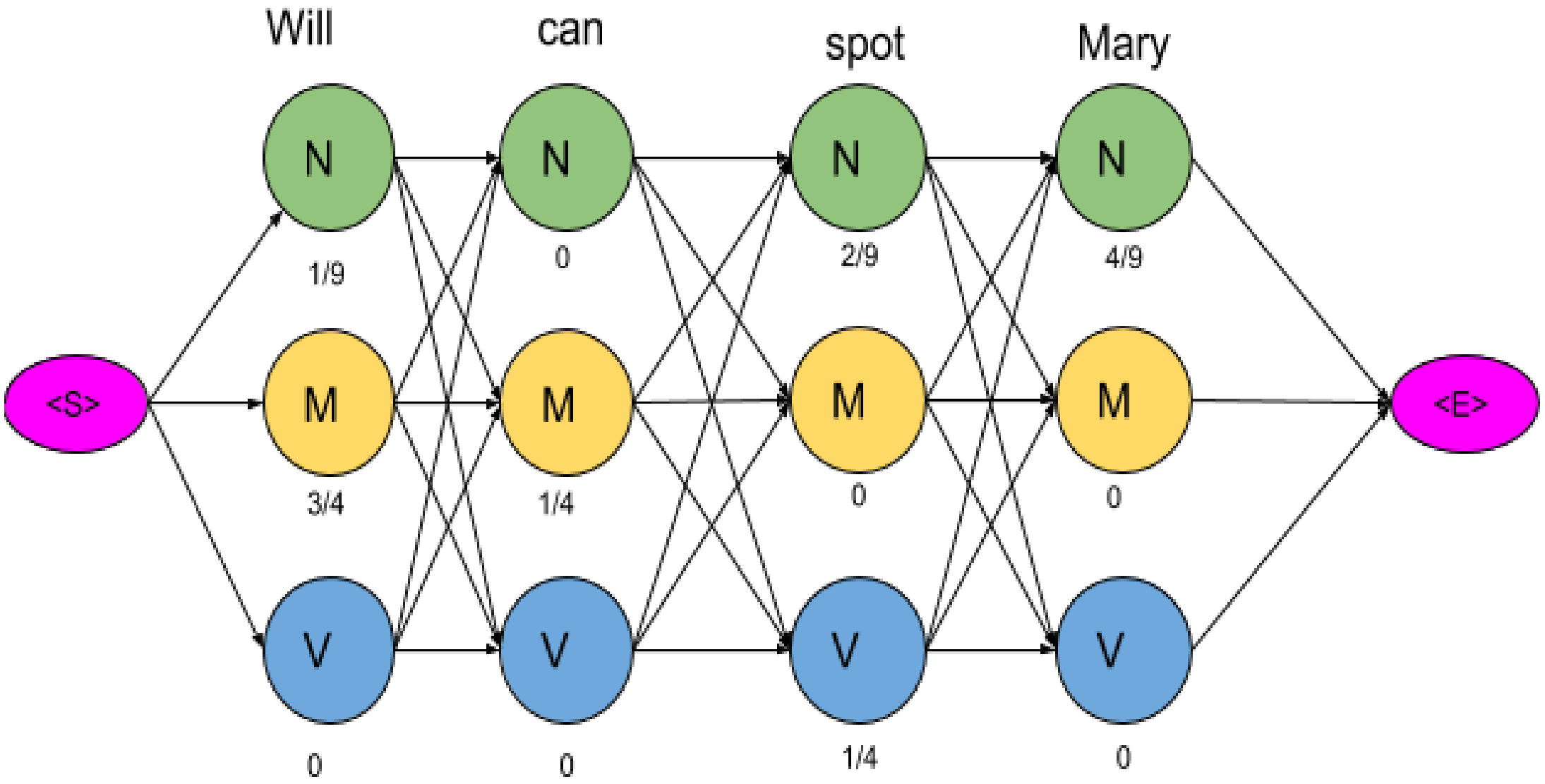
3/4*1/9*3/9*1/4*3/4*1/4*1*4/9*4/9=0.00025720164

- For our example, keeping into consideration just three POS tags we have mentioned, 81 different combinations of tags can be formed

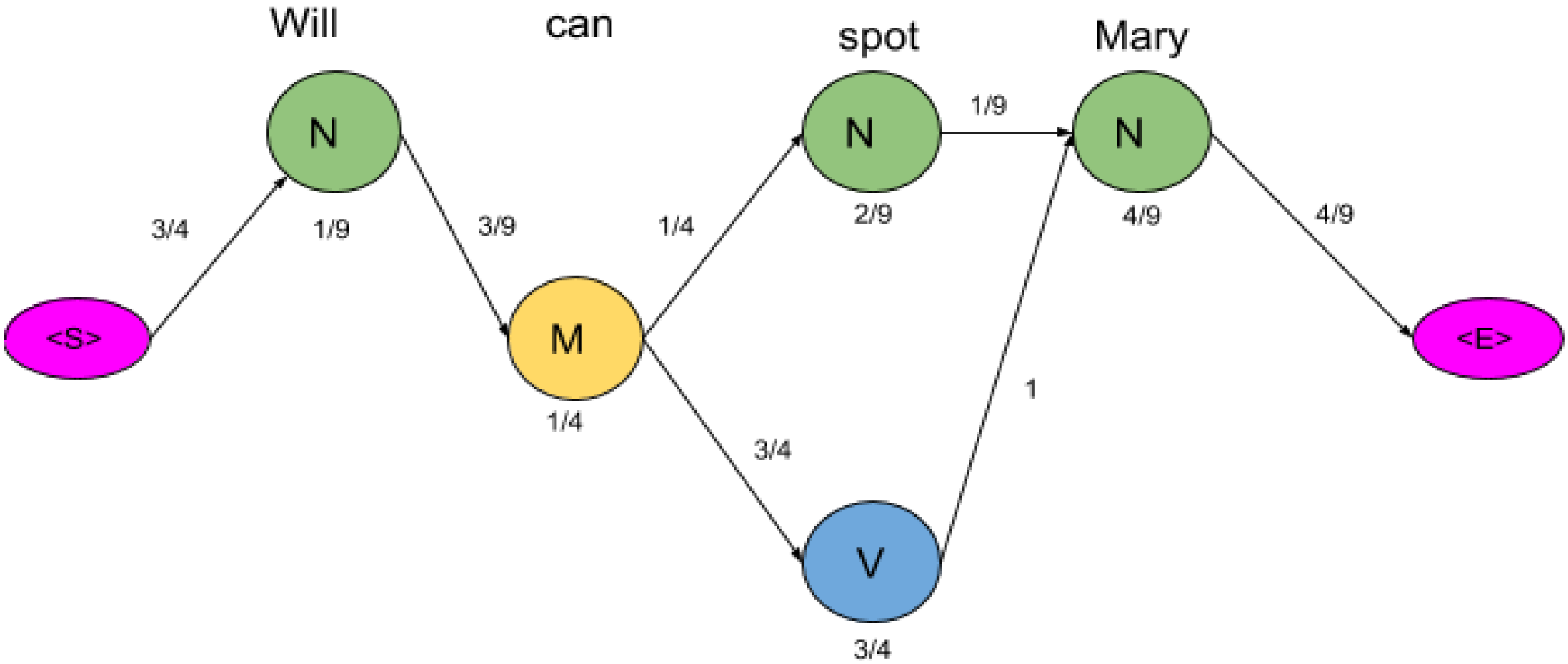
- In this case, calculating the probabilities of all 81 combinations seems achievable

- But when the task is to tag a larger sentence and all the POS tags in the Penn Treebank project are taken into consideration, the number of possible combinations grows exponentially and this task seems impossible to achieve.

- let us visualize these 81 combinations as paths and using the transition and emission probability, mark each vertex and edge as shown in the figure



The next step is to delete all the vertices and edges with probability zero, also the vertices which do not lead to the endpoint are removed


Will
can
spot
Mary
N
N
N
<S>
M
V
<E>
3/4
1/9
3/9
1/4
1/9
2/9
4/9
4/9
1/4
3/4
1
3/4

Example

- Now there are only two paths that lead to the end, let us calculate the probability associated with each path.

- **<S>→N→M→N→N→<E> =3/4*1/9*3/9*1/4*1/4*2/9*1/9*4/9*4/9=0.00000846754**

- **<S>→N→M→N→V→<E>=3/4*1/9*3/9*1/4*3/4*1/4*1*4/9*4/9=0.00025720164**

- **Clearly, the probability of the second sequence is much higher and hence the HMM is going to tag each word in the sentence according to this sequence.**

- Given an HMM and an input string, the **Viterbi algorithm** is used to decode the optimal tag sequence

- For any model, such as an HMM, that contains hidden variables, the task of determining which sequence of variables is the underlying source of some sequence of observations is called the **decoding** task.

- The **Viterbi** algorithm is the most common decoding algorithm used for HMMs, whether for part-of-speech tagging or for speech recognition.

- The term **Viterbi** is common in speech and language processing, but this is a standard application of the classic **dynamic programming** algorithm, and looks a lot like the **minimum edit distance** algorithm

Formal definition of HMM

- The goal of HMM decoding is to choose the tag sequence that is most probable given the observation sequence of n word w_1^n:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n) \tag{10.4}$$

by using Bayes' rule to instead compute:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)} \tag{10.5}$$

Furthermore, we simplify Eq. 10.5 by dropping the denominator $P(w_1^n)$:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(w_1^n | t_1^n) P(t_1^n) \tag{10.6}$$

HMM taggers make two further simplifying assumptions. The first is that the probability of a word appearing depends only on its own tag and is independent of neighboring words and tags:

$$P(w_1^n | t_1^n) \approx \prod_{i=1}^{n} P(w_i | t_i) \tag{10.7}$$

Formal definition of HMM

The second assumption, the **bigram** assumption, is that the probability of a tag is dependent only on the previous tag, rather than the entire tag sequence;

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1}) \tag{10.8}$$

Plugging the simplifying assumptions from Eq. 10.7 and Eq. 10.8 into Eq. 10.6 results in the following equation for the most probable tag sequence from a bigram tagger, which as we will soon see, correspond to the **emission probability** and **transition probability** from the HMM of Chapter 9.

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n|w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} \overbrace{P(w_i|t_i)}^{\text{emission}} \overbrace{P(t_i|t_{i-1})}^{\text{transition}} \tag{10.9}$$

Transformation-Based Tagging

- Also called **Brill tagging**, is an instance of the **Transformation-Based Learning** (TBL) approach to machine learning (Brill,1995)
- It draws inspiration from both the rule-based and stochastic taggers.
- Like the rule-based taggers, TBL is based on rules that specify what tags should be assigned to what words.
- But like the stochastic taggers, TBL is a machine learning technique, in which rules are automatically induced from the data.
- Like some but not all of the HMM taggers, TBL is a supervised learning technique; it assumes a pre-tagged training corpus

TBL Concept

- The TBL algorithm has a set of tagging rules.
- A corpus is first tagged using the broadest rule, that is, the one that applies to the most cases.
- Then a slightly more specific rule is chosen, which changes some of the original tags.
- Next an even narrower rule, which changes a smaller number of tags (some of which might be previously changed tags).

How TBL Rules Are Applied?

- Before the rules apply, the tagger labels every word with its most-likely tag. We get these most-likely tags from a tagged corpus.
- For example, in the Brown corpus, ***race*** is most likely to be a noun:

  $P(\text{NN}|\text{race}) = .98$

  $P(\text{VB}|\text{race}) = .02$

- This means that the two examples of ***race*** that we saw above will both be coded as **NN**.
- In the first case, this is a mistake, as NN is the incorrect tag:

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

- In the second case this *race* is correctly tagged as an NN:

  the/DT race/NN for/IN outer/JJ space/NN

- After selecting the most-likely tag, Brill's tagger applies its transformation rules.

- As it happens, Brill's tagger learned a rule that applies exactly to this mistagging of ***race***

  Change **NN** *to* **VB** *when the previous tag is* **TO**

- This rule would change **race/NN** to **race/VB** in exactly the following situation since it is preceded by **to/TO**

  expected/VBN to/TO race/NN→ expected/VBN to/TO race/VB

How TBL rules are learned?

Brill's TBL has three major stages:

1. It first labels every word with its most likely tag.
2. It then examines every possible transformation and selects the one that results in the most improved tagging.
3. Finally, it then re-tags the data according to this rule.

- The last two stages are repeated until some stopping criterion is reached, such as insufficient improvement over the previous pass.
- Note that stage two requires that TBL knows the correct tag of each word; that is, TBL is a supervised learning algorithm.
- The output of the TBL process is an ordered list of transformations; these then constitute a "**tagging procedure**" that can be applied to a new corpus.

Challenges in POS Tagging:

- The design of the training set or **training corpus** needs to be carefully considered.
  - If the training corpus is too specific to the task or domain, the probabilities may be too narrow and not generalize well to tagging sentences in very different domains.
  - But if the training corpus is too general, the probabilities may not do a sufficient job of reflecting the task or domain.
- The problem with having a fixed training set, development set, and test set is that in order to save lots of data for training, the test set might not be large enough to be representative.
  - Thus a better approach would be to somehow use **all** our data both for training and test
  - The idea is to use **cross-validation**.
  - In cross-validation, we randomly choose a training and test set division of our data, train our tagger, and then compute the error rate on the test set.
  - Generally, 10-cross validation is performed

- Tag Indeterminacy and Tokenization
  - Tag indeterminacy arises when a word is ambiguous between multiple tags and it is impossible or very difficult to disambiguate
  - Common tag indeterminacies include adjective versus preterite versus past participle (JJ/VBD/VBN), and adjective versus noun as a prenominal modifier (JJ/NN).
  - Given a corpus with these indeterminate tags, there are 3 ways to deal with tag indeterminacy when training and scoring part of speech taggers

- Unknown Words
  - All the tagging algorithms we have discussed require a dictionary that lists the possible parts-of-speech of every word.
  - But the largest dictionary will still not contain every possible word
  - Therefore in order to build a complete tagger we cannot always use a dictionary to give us $p(w_i|t_i)$. We need some method for guessing the tag of an unknown word.

END