Natural Language Processing

UNIT-2

Word Level Analysis

- Refer videos and notes on
  - N-gram Model
  - Advantages and Disadvantages with N-gram Model
  - N-gram Smoothing
  - Perplexity

English Parts of Speech

- **Noun (person, place or thing)**
  - Singular (NN): dog, fork
  - Plural (NNS): dogs, forks
  - Proper (NNP, NNPS): John, Springfields
  - Personal pronoun (PRP): I, you, he, she, it
  - Wh-pronoun (WP): who, what
- **Verb (actions and processes)**
  - Base, infinitive (VB): eat
  - Past tense (VBD): ate
  - Gerund (VBG): eating
  - Past participle (VBN): eaten
  - Non 3rd person singular present tense (VBP): eat
  - 3rd person singular present tense: (VBZ): eats
  - Modal (MD): should, can
  - To (TO): to (to eat)

English Parts of Speech (cont.)

- **Adjective (modify nouns)**
  - Basic (JJ): red, tall
  - Comparative (JJR): redder, taller
  - Superlative (JJS): reddest, tallest
- **Adverb (modify verbs)**
  - Basic (RB): quickly
  - Comparative (RBR): quicker
  - Superlative (RBS): quickest
- **Preposition (IN): on, in, by, to, with**
- **Determiner:**
  - Basic (DT) a, an, the
  - WH-determiner (WDT): which, that
- **Coordinating Conjunction (CC): and, but, or,**
- **Particle (RP): off (took off), up (put up)**

Closed vs. Open Class

- ***Closed class*** categories are composed of a small, fixed set of grammatical function words for a given language.
  - Pronouns, Prepositions, Modals, Determiners, Particles, Conjunctions
- **Open class** categories have large number of words and new ones are easily invented.
  - Nouns (Googler, textiles'), Verbs (Google), Adjectives (geeky), Adverb (automatically)

Ambiguity in POS Tagging

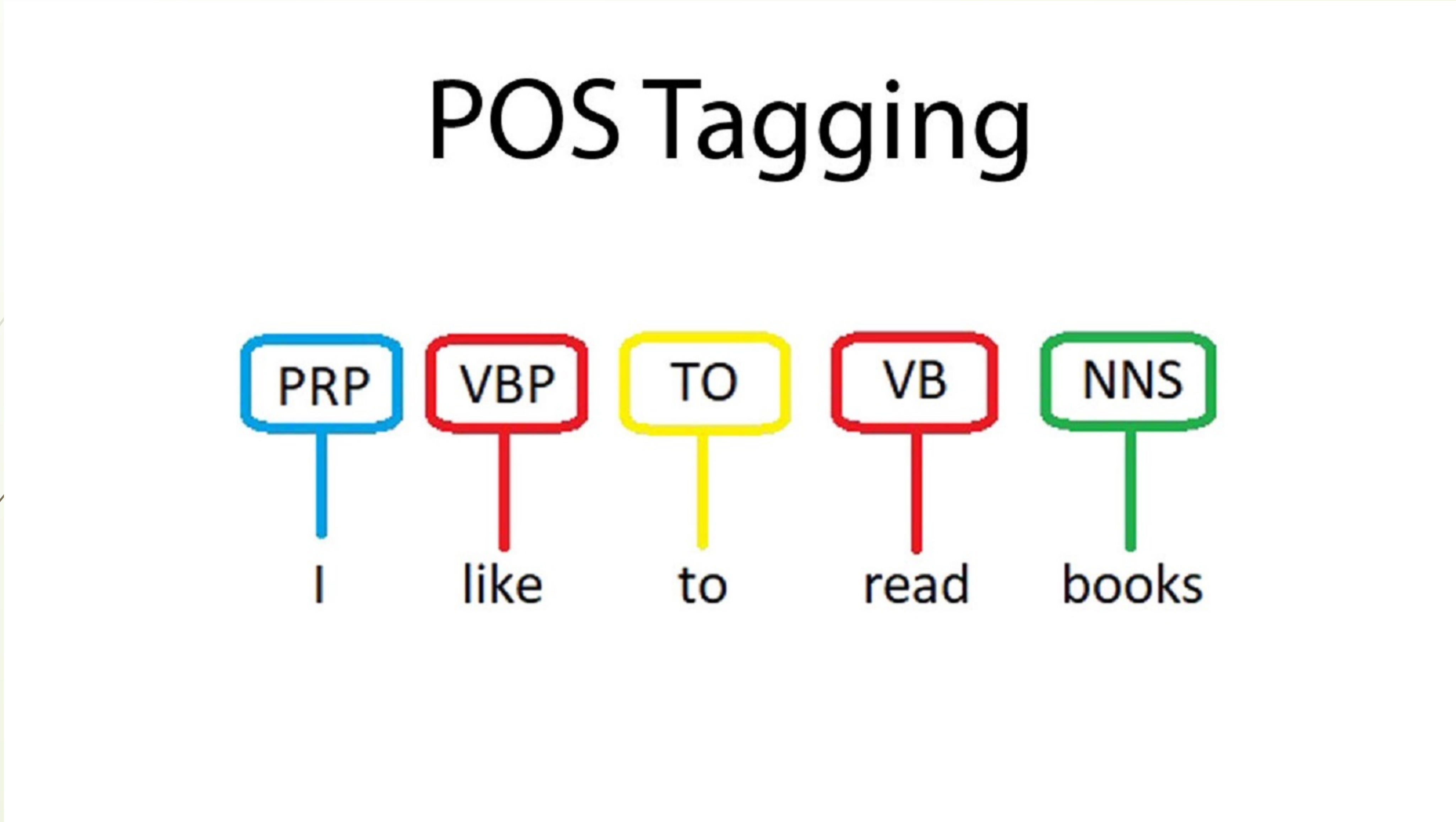
- "Like" can be a verb or a preposition
  - I like/VBP candy.
  - Time flies like/IN an arrow.
- "Around" can be a preposition, particle, or adverb
  - I bought it at the shop around/IN the corner.
  - I never got around/RP to getting a car.
  - A new Prius costs around/RB $25K.

Part of Speech Tagging

- Part of speech tagging is simply assigning the correct part of speech for each in an input sentence
- We assume that we have the following:
  - A set of tags (our tag set)
  - A dictionary that tells us the possible tags for each word (including all morphological variants).
  - A text to be tagged.
- There are different algorithms for tagging.
  - Rule Based Tagging – uses hand-written rules
  - Stochastic Tagging – uses probabilities computed from training corpus
  - Transformation Based Tagging – uses rules learned automatically

How hard is tagging?

- Most words in English are unambiguous. They have only a single tag.
- But many of most common words are ambiguous:
  - can/verb can/auxiliary can/noun
- The number of word types in Brown Corpus
  - unambiguous (one tag) 35,340
  - ambiguous (2-7 tags) 4,100
    - 2 tags 3760
    - 3 tags 264
    - 4 tags 61
    - 5 tags 12
    - 6 tags 2
    - 7 tags 1
- While only 11.5% of word types are ambiguous, over 40% of Brown corpus tokens are ambiguous.


POS Tagging
PRP
VBP
TO
VB
NNS
I
like
to
read
books

Problem Setup

- There are M types of POS tags
  - Tag set: $\{t_1,..,t_M\}$.
- The word vocabulary size is V
  - Vocabulary set: $\{w_1,..,w_V\}$.
- We have a word sequence of length n:

  $W = w_1, w_2 \ldots w_n$
- Want to find the best sequence of POS tags:

  $T = t_1, t_2 \ldots t_n$

$$T_{best} = \arg\max_{T} \Pr(T \mid W)$$

Rule-Based Part-of-Speech Tagging

- **First Stage**: Uses a dictionary to assign each word a list of potential parts-of-speech.
- **Second Stage**: Uses a large list of handcrafted rules to window down this list to a single part-of-speech for each word.

- The ENGTWOL is a rule-based tagger
  - In the first stage, uses a two-level lexicon transducer
  - In the second stage, uses hand-crafted rules (about 1100 rules)

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}

After The First Stage

- Example: He had a book.
- After the first stage:
  - he **he/pronoun**
  - had **have/verbpast** have/auxliarypast
  - a **a/article**
  - book **book/noun** book/verb

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 language model (n-gram)

Stochastic Tagging (cont.)

- Making some simplifying Markov assumptions, the basic HMM equation for a single tag is:

$$t_i = \text{argmax}_j \; P(t_j \mid t_{i-1}) * P(w_i \mid t_j)$$

  - The function $\text{argmax}_x F(x)$ means "the x such that F(x) is maximized"
  - The first P is the tag sequence probability, the second is the word likelihood given the tag.
- Most of the better statistical models report around 95% accuracy on standard datasets
- But, note you get 91% accuracy just by picking the most likely tag!

A Simple Example

- From the Brown Corpus
- Secretariat/NNP is/VBZ expected/VBN to/TO *race*/VB tomorrow/NN
- People/NNS continue/VBP to/TO inquire/VB the/DT reason/NN for/IN the/DT *race*/NN for/IN outer/JJ space/NN

Assume previous words have been tagged, and we want to tag the word *race*.

Bigram tagger

- to/TO *race*/?
- the/DT *race*/?

Example (cont,.)

- Goal: choose between NN and VB for the sequence *to race*
- Plug these into our bigram HMM tagging equation:

P(word | tag) * P(tag | previous-n-tags)

- P(race | VB) * P(VB | TO)
- P(race | NN) * P(NN | DT)

How do we compute the tag sequence probabilities and the word likelihoods?

Word Likelihood

- We must compute the likelihood of the word race given each tag. I.e., P(race | VB) and P(race | NN)

- Note: we are NOT asking which is the most likely tag for the word.
- Instead, we are asking, if we were expecting a verb, how likely is it that this verb would be *race*?
- From the Brown and Switchboard Corpora:

  P(*race* | VB) = .00003

  P(*race* | NN) = .00041

Tag Sequence Probabilities

- Computed from the corpus by counting and normalizing.
- We expect VB more likely to follow TO because infinitives (*to race, to eat*) are common in English, but it is possible for NN to follow TO (*walk to school, related to fishing*).
- From the Brown and Switchboard corpora:

  P(VB | TO) = .340

  P(NN | TO) = .021

And the Winner is…

Multiplying tag sequence probabilities by word likelihoods gives

- $P(race \mid VB) * P(VB \mid TO) = .000010$
- $P(race \mid NN) * P(NN \mid TO) = .000007$

So, even a simple bigram version correctly tags race as a VB, despite the fact that it is the less likely sense.

Performance

- This method has achieved 95-96% correct with reasonably complex English tagsets and reasonable amounts of hand-tagged training data.

Transformation-Based (Brill) Tagging

A hybrid approach

- Like rule-based taggers, this tagging is based on rules
- Like (most) stochastic taggers, rules are also automatically induced from hand-tagged data

Basic Idea: do a quick and dirty job first, and then use learned rules to patch things up

Overcomes the pure rule-based approach problems of being too expensive, too slow, too tedious etc…

An instance of Transformation-Based Learning.

Examples

- Race
  - “race” as NN: .98
  - “race” as VB: .02
- So you’ll be wrong 2% of the time, which really isn’t bad
- Patch the cases where you know it has to be a verb
  - Change NN to VB when previous tag is TO

Brill’s Tagger 3 Stages

1. Label every word with its most likely tag.
2. Examine every possible transformation, and select the one that results in the most improved tagging.
3. Re-tag the data according to the selected rule.

Go to 2 until stopping criterion is reached.

Stopping:

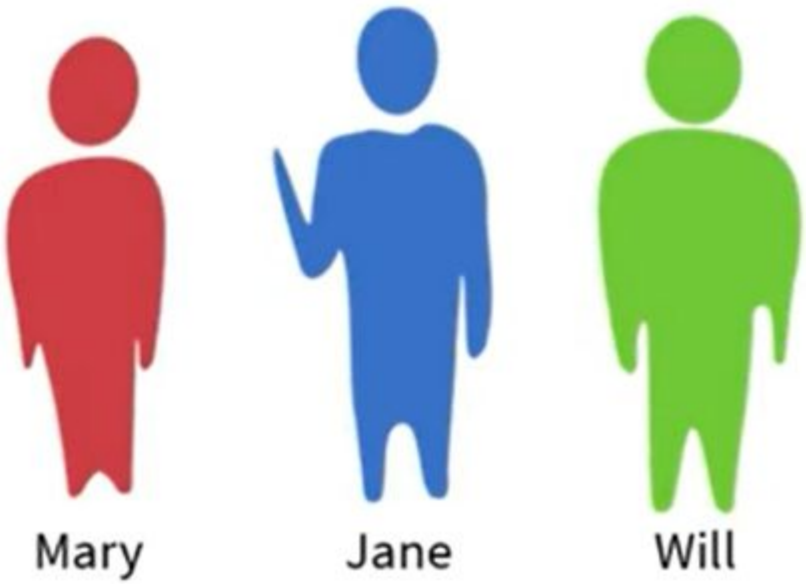
Insufficient improvement over previous pass.

Output: Ordered list of transformations. These constitute a tagging procedure.

HMM Model

Parts of speech

- **Noun** Eg: John, car, India, apple, dog, house
- **Modal Verb** Eg: must, will, would, can, may
- **Verb** Eg: run, swim, talk, eat, speak, etc



Mary saw Will.

Jane	saw	Will	Marry	saw	Jane
noun	verb	noun	noun	verb	noun

Lookup Table

Collect data
Labelled data

Create lookup table
Tag each word with the most common part of speech

Tag our target sentence
While going word by word tag the most common part of speech associated with it

Done
🙂

Lookup Table

Mary saw Will.

Jane	saw	Will
noun	*verb*	*noun*
Marry	saw	Jane
noun	*verb*	*noun*

	N	V
Mary	1	0
saw	0	2
Jane	2	0
Will	1	0

Lookup Table

			noun
Marry	**will**	**see**	**will**
noun	*modal*	*verb*	*???*

Our data!

Mary	**will**	**see**	**Jane**
noun	*modal*	*verb*	*noun*
Will	**will**	**see**	**Mary**
noun	*modal*	*verb*	*noun*
Jane	**will**	**see**	**Will**
noun	*modal*	*verb*	*noun*

	N	V	M
Mary	2	0	0
see	0	3	0
Jane	2	0	0
Will	2	0	3

So how do we consider **context**?

Lookup Table

Extra

Marry	will	see	will
?	?	?	?

Our data!

Mary	will	see	Jane
noun	*modal*	*verb*	*noun*
Will	will	see	Mary
noun	*modal*	*verb*	*noun*
Jane	will	see	Will
noun	*modal*	*verb*	*noun*

BIGRAMS

	N-M	M-V	V-N
mary-will	1	0	0
will-see	0	3	0
see-jane	0	0	1
will-will	1	0	0
see-mary	0	0	1
jane-will	1	0	0
see-will	0	0	1

Lookup Table

Marry	will	see	will
noun	*modal*	*verb*	*noun*

Our data!

Mary	**will**	**see**	**Jane**
noun	*modal*	*verb*	*noun*
Will	**will**	**see**	**Mary**
noun	*modal*	*verb*	*noun*
Jane	**will**	**see**	**Will**
noun	*modal*	*verb*	*noun*

BIGRAMS

	N-M	M-V	V-N
mary-will	1	0	0
will-see	0	3	0
see-jane	0	0	1
will-will	1	0	0
see-mary	0	0	1
jane-will	1	0	0
see-will	0	0	1

Our data

Mary Jane can see Will

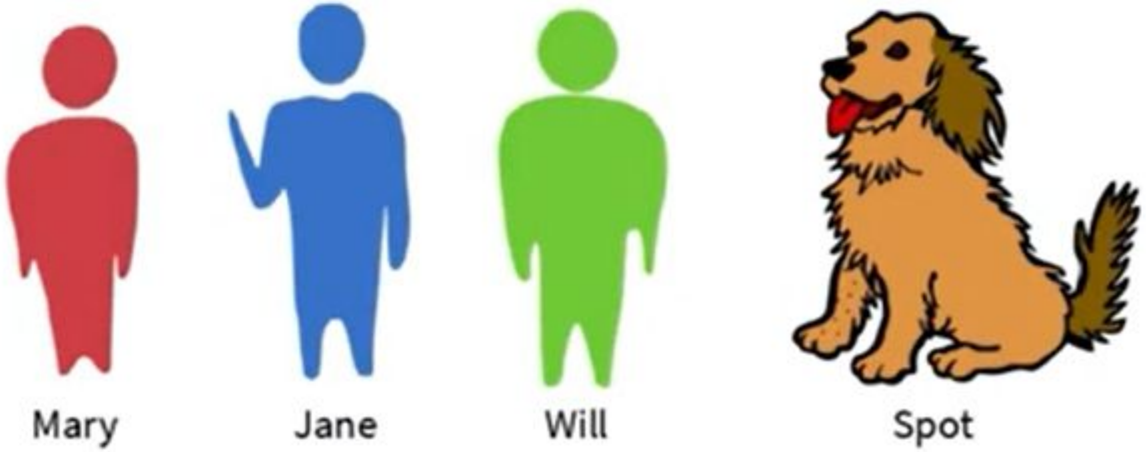
Spot will see Mary

Will Jane spot Marry?

Marry will pat Spot

Jane **will** **spot** **will**

? ? ? ?



Jane	will	spot	will
?	?	?	?

Our data

Mary Jane can see Will

Spot will see Mary

Will Jane spot Marry?

Marry will pat Spot

	N-M	M-V	V-N	etc...
mary-jane				
jane-can				
can-see				
see-will				
spot-will				
will-see				
see-mary				
will-jane				
jane-spot				
spot-mary				
mary-will				
will-pat				
pat-spot				

Jane	will	spot	will
noun	*modal*	*verb*	*noun*

We will need two things

Two probabilities

Transition Probabilities

How likely is it that…

Noun is followed by a modal which is followed by a verb and than a noun

Emission probabilities

How likely is that Jane will be a noun, will be a modal,...

…

🙂

Emission Probabilities

	N	M	V
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

N N M V N
Mary Jane can see Will.

N M V N
Spot will see Mary.

M N V N
Will Jane spot Mary?

N M V N
Mary will pat Spot

Emission Probabilities

	N	M	V
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	1/2
Pat	0	0	1/4

N N M V N
Mary Jane can see Will.

N M V N
Spot will see Mary.

M N V N
Will Jane spot Mary?

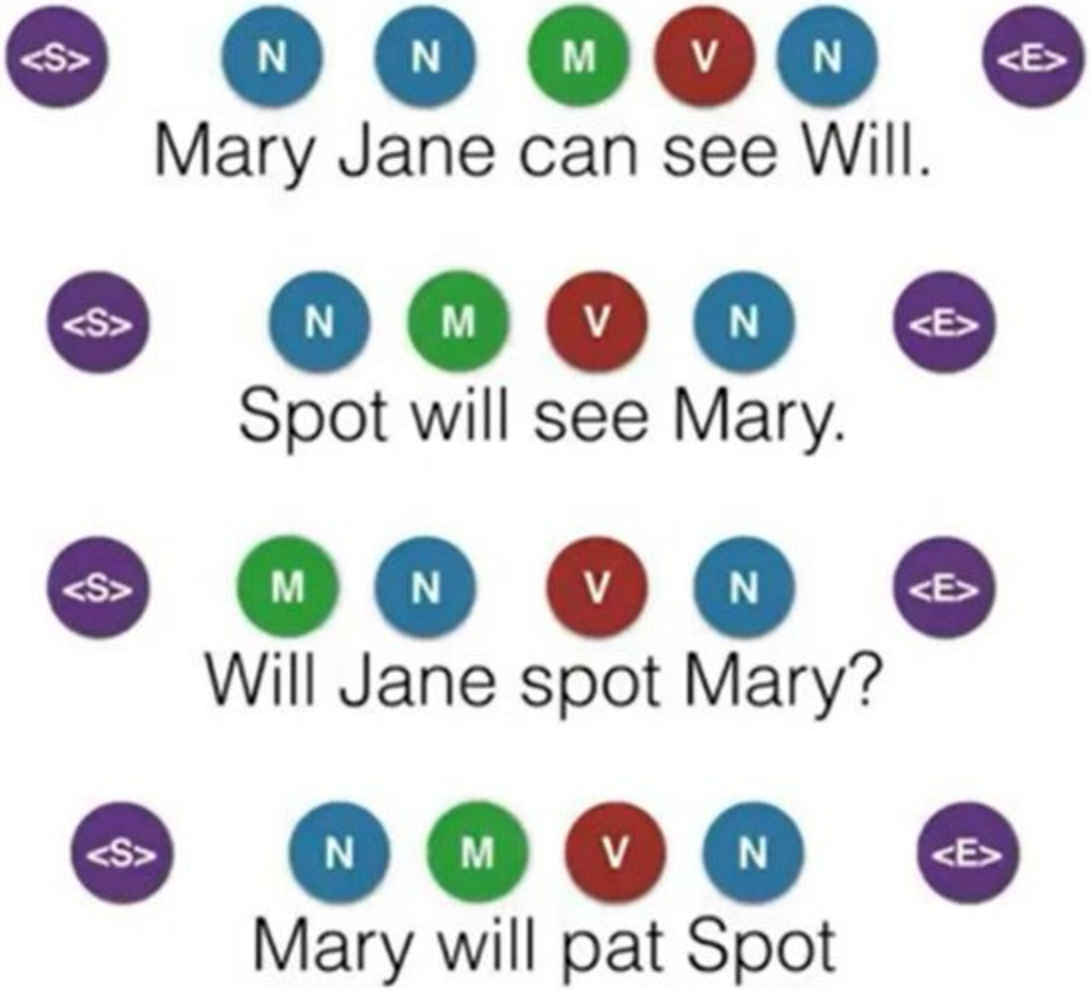
N M V N
Mary will pat Spot

	N	M	V
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

9 = Total no. of verbs that appear

Transition Probabilities

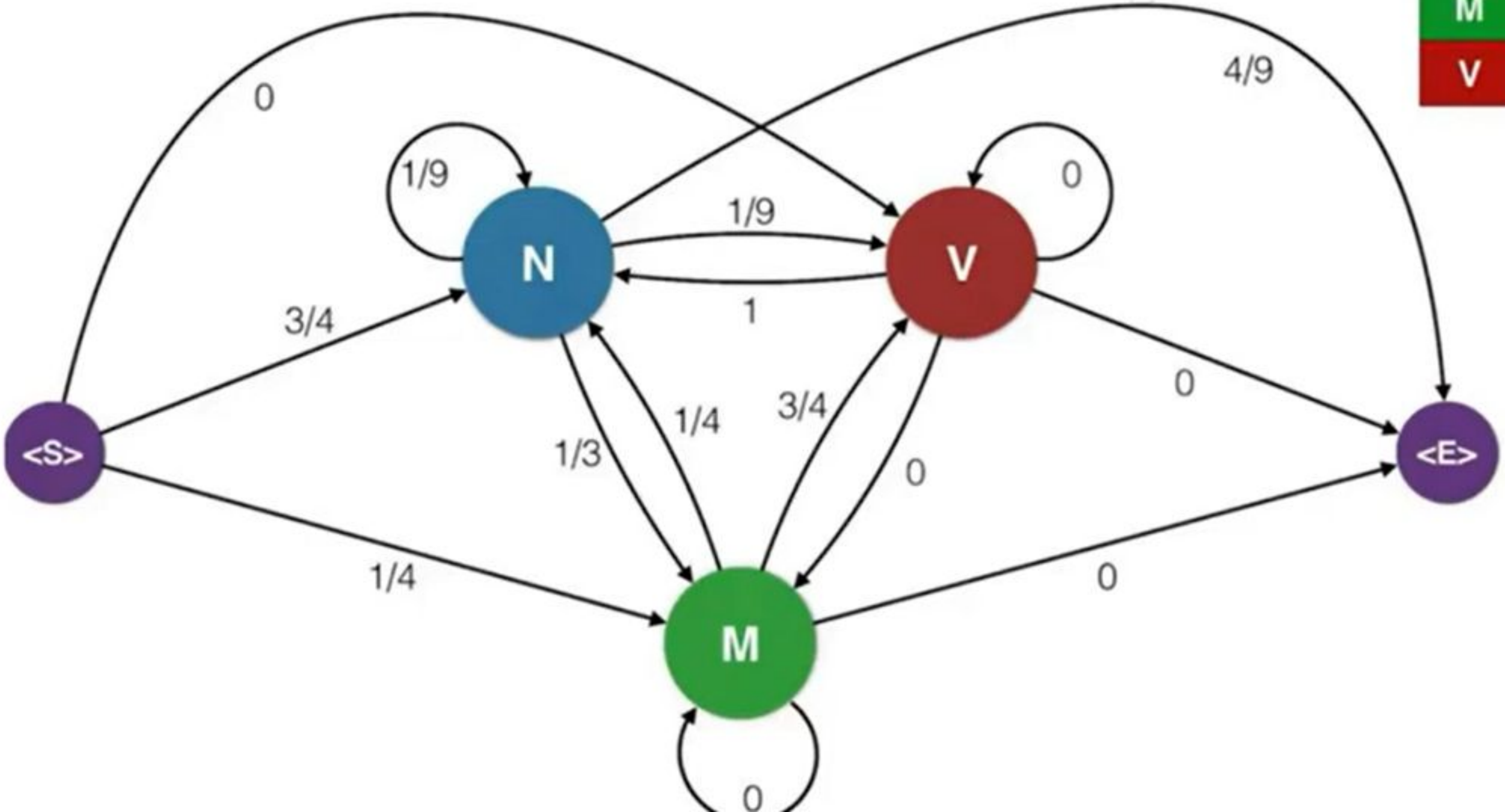
	N	M	V	<E>
<S>	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0



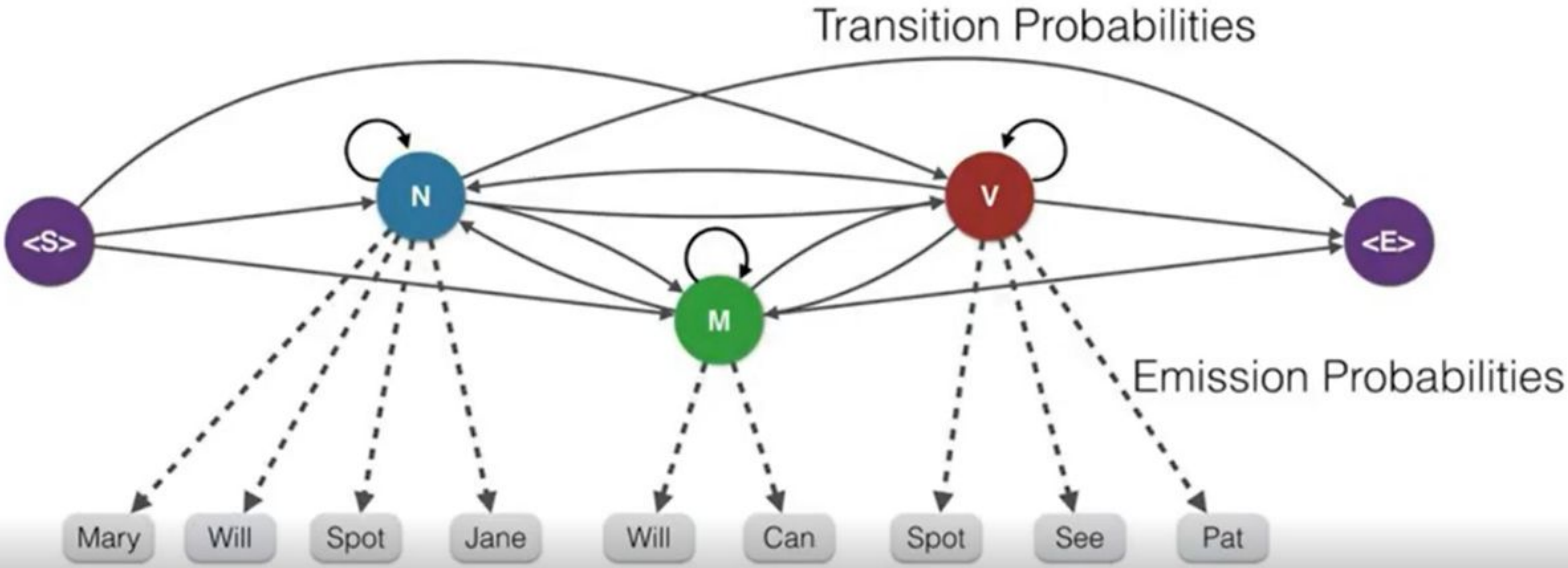
Hidden Markov Model

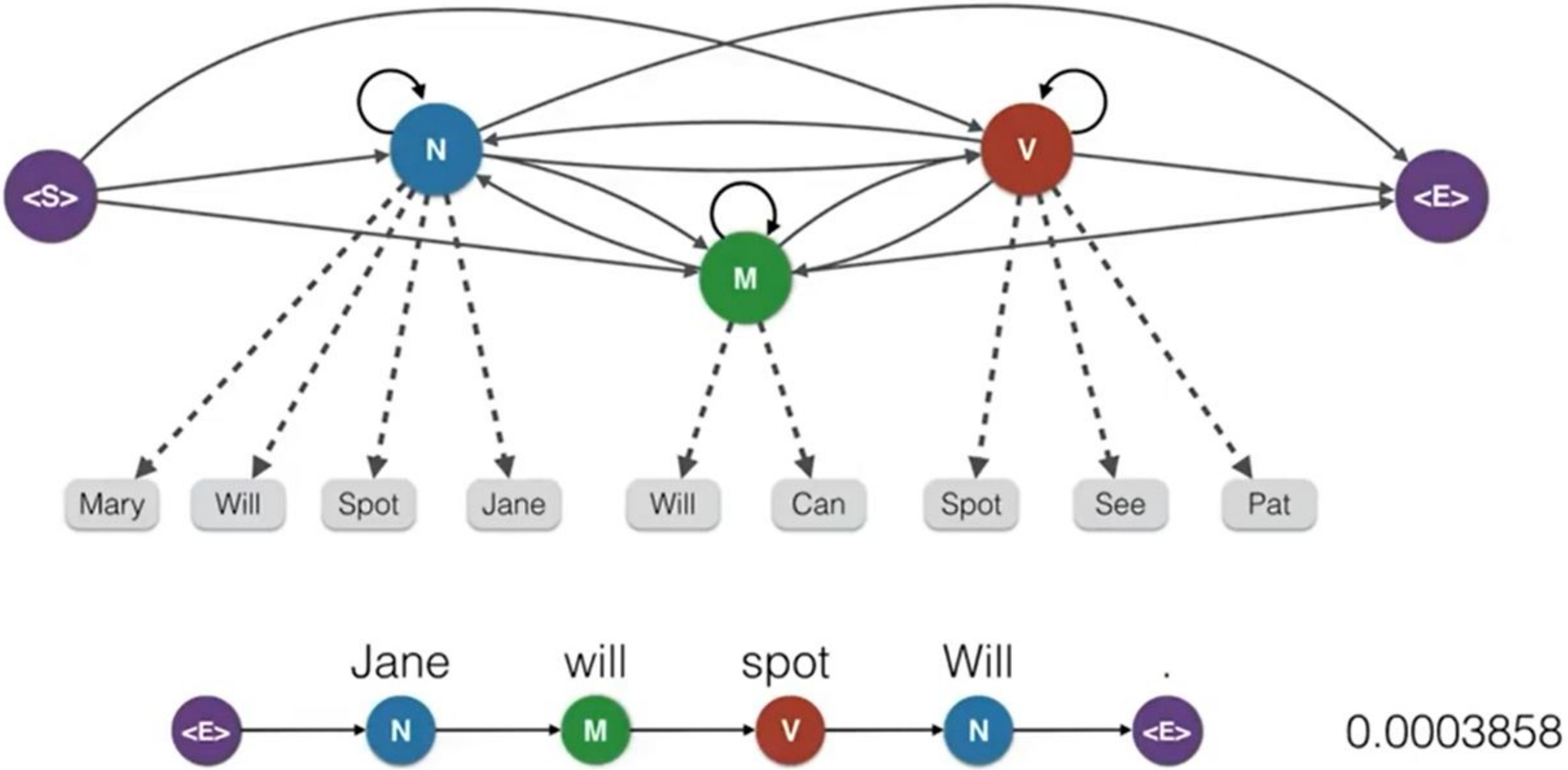
Transition Probabilities

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



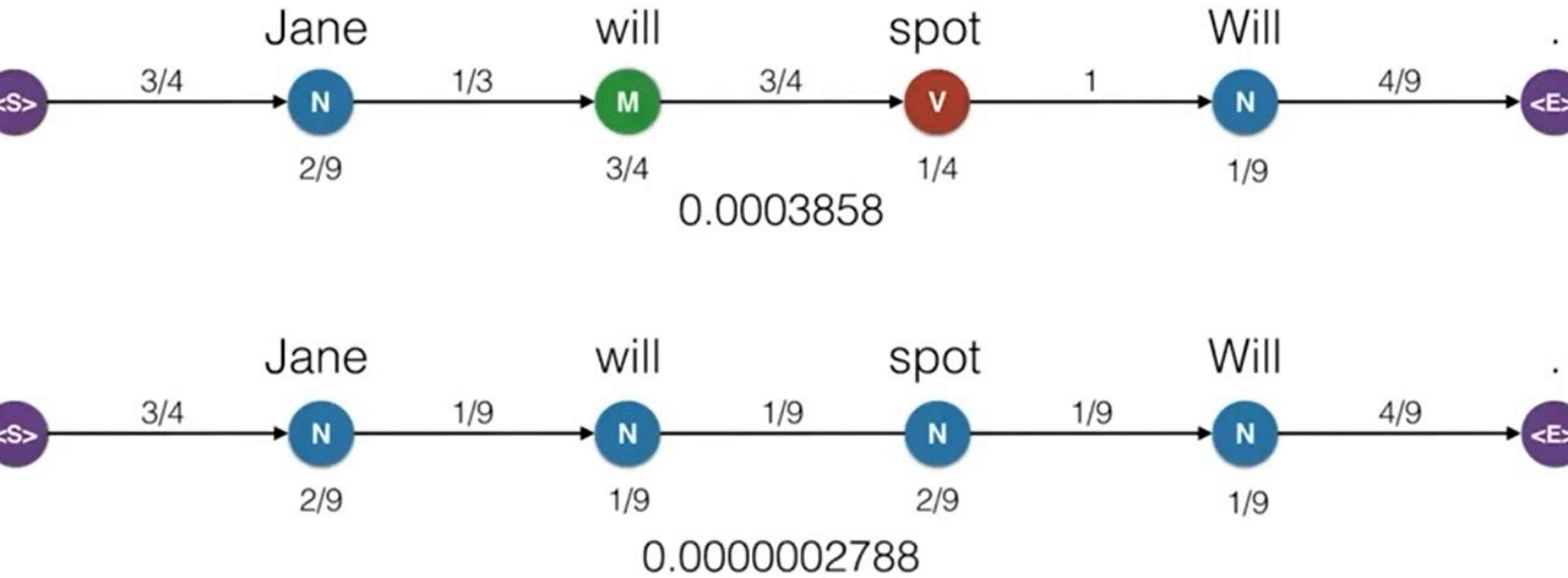
Hidden Markov Model





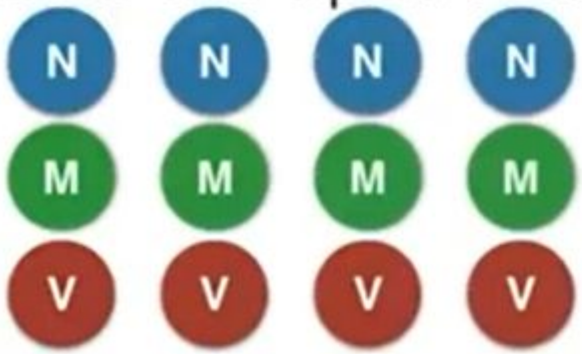
<S>
N
M
V
<E>
Mary
Will
Spot
Jane
Will
Can
Spot
See
Pat
Jane
will
spot
Will
.
<E>
N
M
V
N
<E>
0.0003858

Hidden Markov Model



Answer: 81 Possibilities

Jane will spot Will.



the working of Markov chains, refer to [this](#) link.

Also, have a look at the following example just to see how probability of the current state can be computed using the formula above, taking into account the Markovian Property.

- Exercise 1: Given that today is Sunny, what's the probability that tomorrow is Sunny and the next day Rainy?

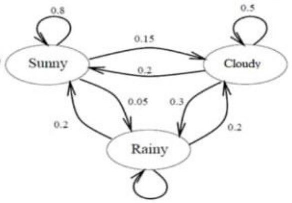
$P(q_2, q_3|q_1) = P(q_2|q_1)P(q_3|q_1, q_2)$

$= P(q_2|q_1)\ P(q_3|q_2)$

$= P(Sunny|Sunny)\ P(Rainy|Sunny)$

$= (0.8)(0.05)$

$= 0.04$



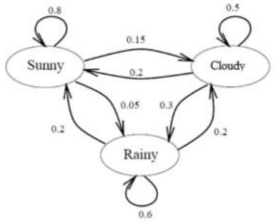
Apply the Markov property in the following example.

- Exercise 2: Assume that yesterday's weather was Rainy, and today is Cloudy, what is the probability that tomorrow will be Sunny?

$P(q_3|q_1, q_2) = P(q_3|q_2)$

$= P(Sunny|Cloudy)$

$= 0.2$



We can clearly see that as per the Markov

Solution to Q1

$$O = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$$

$$\begin{aligned}
&P(O \mid \text{Model}) \\
&\quad = P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{Model}) \\
&\quad = P(S_3)\ P(S_3|S_3)\ P(S_3|S_3)\ P(S_1|S_3) \\
&\qquad\quad P(S_1|S_1)\ P(S_3|S_1)\ P(S_2|\ S_3)\ P(S_3|S_2) \\
&\quad = \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\
&\quad = (1)(0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) \\
&\quad = 1.536 \times 10^{-4}
\end{aligned}$$

Solution to Q2

$$O = \{S_{i_1}, S_{i_2}, S_{i_3}, \ldots, S_{i_d}, S_{j_{d+1}} \neq S_i\}$$

$P(O \mid \text{Model},\ q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d)$

where $p_i(d)$ is the (discrete) PDF of duration d in state i.

Notice that $D_i \sim \text{geometric}(p)$, where $p = 1 - a_{ii}$ is the probability of success (exiting state i) and there are $d - 1$ failures before the first success.

Then $\overline{D_i} = \frac{1}{p} = \frac{1}{1-a_{ii}}$

Intuition: Consider a fair die. If the probability of success (a "1") is $p = 1/6$, it will take $1/p = 6$ rolls until a success.

the "math" way: $X \sim \text{geom}(p)$

$$\overline{X} = \sum_{k=1}^{\infty} k(1-p)^{k-1}p = p\sum_{k=1}^{\infty} k(1-p)^{k-1} = p\frac{1}{(1-(1-p))^2} = \frac{p}{p^2} = \frac{1}{p}$$

for $x \in \mathbb{R}, |x| \leq 1$, $\sum_{n=1}^{\infty} nx^{n-1} = \frac{1}{(1-x)^2}$

For example, the expected number of consecutive days of rainy weather is $1/a_{11} = 1/0.6 = 1.67$; for cloudy, 2.5; for sunny, 5.

Markov Model of Weather

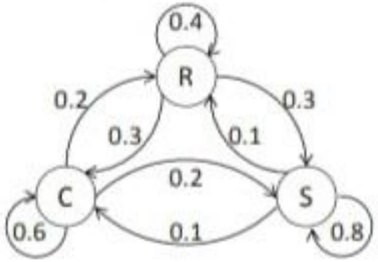
Once a day (e.g. at noon), the weather is observed as one of

state 1 : rainy state 2: cloudy state 3: sunny

The state transition probabilities are

$$A = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

(Notice that each row sums to 1.)



Questions:

1. Given that the weather on day 1 ($t = 1$) is sunny (state 3), what is the probability that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun"?
2. Given that the model is state i, what is the probability that it stays in state i for exactly d days? What is the expected duration in state i (also conditioned on starting in state i)?

Solution to Q1

$$O = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$$

$$\begin{aligned}
P(O \mid \text{Model}) \\
&= P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{Model}) \\
&= P(S_3)\ P(S_3|S_3)\ P(S_3|S_3)\ P(S_1|S_3) \\
&\qquad P(S_1|S_1)\ P(S_3|S_1)\ P(S_2|\ S_3)\ P(S_3|S_2) \\
&= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\
&= (1)(0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) \\
&= 1.536 \times 10^{-4}
\end{aligned}$$

Solution to Q2

$$O = \{\underset{1}{S_i}, \underset{2}{S_i}, \underset{3}{S_i}, \ldots, \underset{d}{S_i}, \underset{d+1}{S_j} \neq S_i\}$$

$$P(O \mid \text{Model},\ q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d)$$

where $p_i(d)$ is the (discrete) PDF of duration d in state i.

Notice that $D_i \sim \text{geometric}(p)$, where $p = 1 - a_{ii}$ is the probability of success (exiting state i) and there are $d - 1$ failures before the first success.

Then $\overline{D_i} = \frac{1}{p} = \frac{1}{1-a_{ii}}$

Intuition: Consider a fair die. If the probability of success (a "1") is $p = 1/6$, it will take $1/p = 6$ rolls until a success.

the "math" way: $X \sim \text{geom}(p)$

$$\overline{X} = \sum_{k=1}^{\infty} k(1-p)^{k-1}p = p\sum_{k=1}^{\infty} k(1-p)^{k-1} = p\frac{1}{(1-(1-p))^2} = \frac{p}{p^2} = \frac{1}{p}$$

for $x \in \mathbb{R}$, $|x| \leq 1$, $\sum_{n=1}^{\infty} nx^{n-1} = \frac{1}{(1-x)^2}$

For example, the expected number of consecutive days of rainy weather is $1/a_{11} = 1/0.6 = 1.67$; for cloudy, 2.5; for sunny, 5.