SAP ID : 60009200027
Name: Aditya Potdar
Div./Batch : K/K1

Page No.
Date

Computational Linguistics

Assignment No. 1

Q.1 Explain the stages of Natural language Processing

There are five stages of NLP :-

① Morphological Analysis :-
This is the first stage of NLP, which involves breaking down the text into individual words or tokens. This is done by identifying spaces between words, punchuation marks, and special characters.

② Syntactic Analysis :- It is used to check grammar, word arrangements, and shows the relationship among words.
For eg :- Agra goes to the Poonam
In the real world, Agra goes to the Poonam does not make any sense, so this sentence is rejected by the Syntactic Analyzer

③ Semantic Analysis :- This stage involves analyzing the meaning of the text. The goal is to understand the context of the text and the relationship between the words. This involves identifying entities, relationships and concepts within the text.
For eg :- She drank some milk sounds correct,
She drank some books sounds incorrect.

Page No.
Date

④ Discourse Analysis :- This stage involves analyzing the structure of the text as a whole, rather than just individual sentences. The goal is to understand the relationships between different sentences and how they contribute to the overall meaning of text.

For eg :- Monkeys eat Banana, when they wake up

Who is they here? Monkeys

⑤ Pragmatic Analysis :- This is the fifth and last phase of NLP. It helps to discover the intended effect by applying a set of rules that characterize cooperative dialogues.

For eg :- "Open the door" is interpreted as a request instead of an order.

Q.2.

Stemming	Lemmatization
① Stemming is faster because it chops words without knowing the context of the word in given sentences.	① lemmatization is slower as compared to stemming but it knows the context of the word before proceeding.
② It is a rule-based approach	② It is a dictionary-based approach.
③ Accuracy is less.	③ Accuracy is more.
④ When we convert any word into root-form then stemming may create the non-existence meaning of word.	④ lemmatization always gives the dictionary meaning word while converting into root-form

Here, we dont know whether Dipesh loves his cat or Rahul's cat

⑤ Stemming is preferred when the meaning of the word is not important for analysis. Eg:- Spam detection	⑤ Lemmatization would be recommended when the meaning of the word is important for analysis. Eg :- Question Answer
⑥ For eg:- studies → studi	⑥ For eg. Studies → study

Q.3. Explain the various types of ambiguities in Natural language Processing.

① Lexical Ambiguity :- The ambiguity of a single word is called lexical ambiguity. It occurs when a word has multiple meanings.
For eg:- 'bank', could refer to a financial institution or the edge of a river.

② Syntactical Ambiguity :- This type of ambiguity occurs when a sentence is parsed in different ways.
For eg:- The sentence, "The man saw the girl with the telescope". It is ambiguous whether the man saw the girl carrying a telescope or he saw her through his telescope.

③ Semantic Ambiguity :- When a sentence has more than one meaning then it is called as semantic ambiguity.
For eg:- Rahul loves his cat and Dipesh does too.
Here, we dont know whether Dipesh loves his cat or Rahul's cat

Page No.
Date

(4) Anaphoric Ambiguity :- This type of ambiguity occurs when a pronoun refers to an ambiguous noun or phrase. For eg :- the sentence, 'John saw the cat and he fed it" is ambiguous because it is unclear whether "he" refers to John or someone else.

(5) Pragmatic Ambiguity :- Such kind of ambiguity refers to the situation where the context of a phrase gives it multiple interpretations.
For eg :- 'You are late", it is unclear that the intention of speaker is informative or criticizing.

Q.4. (A) I saw the girl with the binocular.
Syntactial Ambiguity

(B) Rahul loves his cat and Dipesh does too.
Semantic Ambiguity

(C) Monkeys eat Banana, when they wake up.
Anaphoric Ambiguity

(D) Monkeys eat Banana, when they are ripe.
Anaphoric Ambiguity

(E) You are late. Pragmatic Ambiguity

Page No.
Date

Q.5 Explain Challenges in NLP.

① Multilingualism :- There are many languages spoken around the world, each with its own unique grammar and vocabulary. Developing NLP systems that can understand and process multiple languages is a significant challenge.

② Ellipsis refers to removal of one or more words from a clause because they are understood from the context. It requires systems to infer missing information based on context and other clues.

For eg :- "Mary went to the store, and John did too", the verb "went" is elided in the second clause because it is understood from the context. We know that John went to the store because of previous clause but an NLP system would need to infer this information from the context.

③ Punctuational Ambiguity :- This ambiguity arises from the incorrect use or absence of punctuation marks, which can cause confusion in the meaning of a sentence. It is challenging for NLP systems as it requires them to correctly interpret the intended meaning of a sentence based on the context.

Page No.
Date

Q.6. Solution:-

① <s> Julia
② Julia is
③ is visiting
④ visiting the
⑤ the museum
⑥ museum </s>
~~⑦ Ja <s> Julia~~
⑧ Julia ,
⑨ , Grover
⑩ Grover and
⑪ and Natasha
⑫ Natasha are
⑬ are friends
⑭ friends </s>
⑮ <s> Zoe
⑯ Zoe and
~~⑰ and Natasha~~
⑱ Natasha will
⑲ will meet
⑳ meet Julia
㉑ Julia in
㉒ in the
~~㉓ the museum~~
~~㉔~~

~~23~~ 20

Page No.
Date

Q7. Solution:-

$C(\text{"Computational"}) = 1200$

$P(\text{"linguistics"} \mid \text{"Computational"}) = 0.25$

$$P(\omega_n \mid \omega_{n-1}) = \frac{C(\omega_{n-1}\omega_n)}{C(\omega_{n-1})}$$

$C(\text{"linguistics" "Computational"}) = 300$

$$\therefore P_{Add-2}(\text{"linguistics"} \mid \text{"Computational"}) = \frac{300 + 2}{1200 + 2(4400)}$$

$$= \frac{\cancel{1202}}{\cancel{9100}} = \frac{1}{20}$$

$$P_{Add-2} = \cancel{0.1321} = 0.05$$

Q.8. Solution:-

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2 ; \text{ if } S_1(i) \neq S_2(j) \\ 0 ; \text{ if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

G	7	8	7	6	5	6	5
N	6	7	6	5	4	5	4
I	5	6	5	4	3	4	5
T	4	5	4	3	2	3	4
T	3	4	3	2	2	3	4
I	2	3	2	3	4	5	6
S	1	2	3	4	5	6	7
#	0	1	2	3	4	5	6
	#	K	I	T	T	E	N

For $D(\cancel{K}, \cancel{S}) \equiv D(1,1) = \min \begin{cases} D(0,1) + 1 \\ D(1,0) + 1 \\ D(0,0) + 2; \; S \neq K \end{cases}$

$= \min \begin{cases} 2 \\ 2 \\ 2 \end{cases} = \underline{2}$

Similarly for others,

Distance $= \underline{5}$

Page No.
Date

Q.11. Write the algorithm for finding minimum edit distance using dynamic programming.

Initialization :- $D(i,0) = i$; $D(0,j) = j$

Recurrence Relation :-

For each $i = 1$ to M
For each $j = 1$ to N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + 2; \text{ if } X(i) \neq Y(j) \\ \quad + 0; \text{ if } X(i) = Y(j) \end{cases}$$

Termination :- $D(N, M)$ is distance.

Q.12. Write the algorithm for finding Weighted Minimum Edit Distance.

Initialization :- $D(0,0) = 0$;
$D(i,0) = D(i-1,0) + del[X(i)]$; $1 < i \leq N$
$D(0,j) = D(0,j-1) + ins[Y(j)]$; $1 < j \leq M$

Recurrence Relation :-

$$D(i,j) = \min \begin{cases} D(i-1,j) + del[X(i)] \\ D(i,j-1) + ins[Y(j)] \\ D(i-1,j-1) + sub[X(i), Y(j)] \end{cases}$$

Termination :- $D(N, M)$ is distance.

Page No.
Date

Q.13 Explain evaluating language models using perplexity

The best language model is one that best predicts an unseen test set.

The perplexity of a language model on a test set is the inverse probability of the test set, normalized by the number of words.

Minimizing perplexity is the same as maximizing probability.

The perplexity PP of a test set $W = w_1 w_2 \dots w_n$ is

$$PP(W) = \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 \dots w_{i-1})}} \quad \text{by chain rule}$$

The perplexity PP for bigrams :

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

Page No.
Date

Q.14. Write short note on Add-1 Smoothing.

To keep a language model from assigning zero probability to these unseen events, we remove a bit of probability mass from some more frequent events and give it to the events we have never seen. This modification is called smoothing.

The simplest way to do smoothing is to add one to all the counts, before we normalize them into probabilities. This algorithm is called Laplace smoothing (Add-1 smoothing).

Laplace Smoothing for Unigrams :-
The unsmoothed maximum likelihood estimate of the unigram probability of the word w_i is its count c_i normalized by the total number of word tokens N.

$$P(w_i) = c_i / N$$

Laplace Smoothing adds one to each count. Since there are V words in the vocabulary and each one was incremented, we also need to adjust the denominator to take into account the extra V observations.

$$P_{laplace}(w_i) = \frac{(c_i + 1)}{(N + V)}$$

Laplace Smoothing for Bigrams :-
The normal bigram probalities are computed by normalizing each bigram counts by the unigram count

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{C(w_{n-1})}$$

Page No.
Date

Add-one smoothed bigram probabilities :-

$$P^*_{Laplace}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{\sum_w (C(w_{n-1}w)+1))} = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

Q.15. Explain Steps in Noisy Channel Model for non-word error.

Step 1: Selection of language Model

Step 2: Instead of checking the probability $P(x|w)$ for all words w in the vocabulary, we will calculate probability for only suggested words that are 1 or 2 edit distance from x.

Step 3: For calculating $P(x|w)$ we first build an error confusion matrix that tells us for a pair of letters how likely an edit (deletion/insertion/substitution/transposition) to happen.

Step 4: Get the probability : $P(x|w)$

For deletion: deletion(x, w)/ count(xw)

For insertion: insert(x, w)/ count(w)

For substitution : substitution(x, w)/ count(w)

For transposition: transposition(x, w)/ count(xw)

Step 5: Calculation for w'

Once we have the $P(x|w)$ for the candidate suggestions (step 2), we select w' based on

$w' = \text{argmax}_{-w}\ P(x|w) * P(w)$

Page No.
Date

Q16- Explain Free Morphemes and Bound Morphemes with the help of an example.

Free morphemes are morphemes that can stand alone as a word, and they carry their own meaning. Examples of free morphemes includes words such as "book", "cat" and "run". These words can stand alone as independent words in a sentence, and they carry a complete meaning.

Bound morphemes, on the other hand, are morphemes that cannot stand alone as a word. They must be attached to a free morpheme to create a meaningful word. Bound morphemes include prefixes, suffixes, and infixes. Examples of bound morphemes includes the suffix "s" in the word "books".

Q.17. Explain open class words and closed class words with the help of example.

Open class words are words that can be added to or changed easily. These are typically nouns, verbs, adjectives and adverbs. Examples include "computer", "email", "like", "dance", "funny", "quickly", etc.
Closed class words, on the other hand, are words that are fixed in number and rarely added to or changed. These words include determiners, pronouns, prepositions, conjunctions and certain auxillary verbs. Examples include "the", "a", "this", "that", "who", "or", "have", "was", etc.

Page No.
Date

Q.18. Difference between Inflectional morphology and derivational morphology

Inflectional Morphology	Derivational Morphology
① The study of the modification of words to fit into different grammatical contexts.	① The study of the formation of new words that either differ in syntactic category or in meaning from their bases.
② Inflectional morphemes are added to the end of the word.	② Derivational morphemes can be added to the beginning or end of the word.
③ Inflectional morphemes are highly productive, meaning that they can be added to almost any word in the language.	③ Derivational morphemes are less productive and are often restricted to certain word classes or word formations.
④ Eg:- Adding suffix "s" in the word "book" changes the word from singular to plural, without changing the meaning of the word.	④ Eg:- Adding prefix "un" to the word "happy" ~~at~~ has the opposite meaning of word "happy".

Page No.
Date

Q.19. Explain Substitutions and Capture Group in Regular expression with the help of example.

Substitutions are a way to replace certain parts of a string with new text using regular expressions. In regular expression a substitution is defined using the syntax 's/pattern/replacement/' where 'pattern' is the regular expression pattern to match and 'replacement' is the text to replace it with.

For eg:- if input is 'the 35 boxes' and regular expression syntax is 's/([0-9]+)/<\1>/' <> will be placed between any number given in the input. '\1' refers to contents of register

Capture groups are a way to extract and reuse parts of a matched pattern in a regular expression. Capture groups are defined using parentheses '()' around the part of the pattern that we want to capture.

If regular expression is: "/the (.*)er they (.*), the \1er we \2/"

This expression matches "the faster they ran, the faster we ran"

But it doesn't match "the faster they ran, the faster we ate".

Page No.
Date

Q.20. Write the regular expression that matches a string that has an 'a' followed by zero or more 'b'. ab^*

Q.21. Write the regular expression for given language
$L = \{E, a, aa, b, bb, ab, ba, aba, bab, \ldots\}$
any combination of a and b. $(a|b)^*$

Q.23. Write the regular expression for the language
$L = \{a, aa, aaa, \ldots\}$. $a+$

Q.24. Write the regular expression for the language
$L = \{E, 0, 1, 00, 11, 10, 100, \ldots\}$. $(1|0)^*$

Q.22. Write the regular expression for the language
$L = \{a, aba, aab, aaa, abab, \ldots\}$ $(a|ab)^*$

Q.25. Write the regular expression for the language accepting all the string which are starting with 1 and ending with 0, over $\Sigma = \{0, 1\}$.
$1(0|1)^*0$

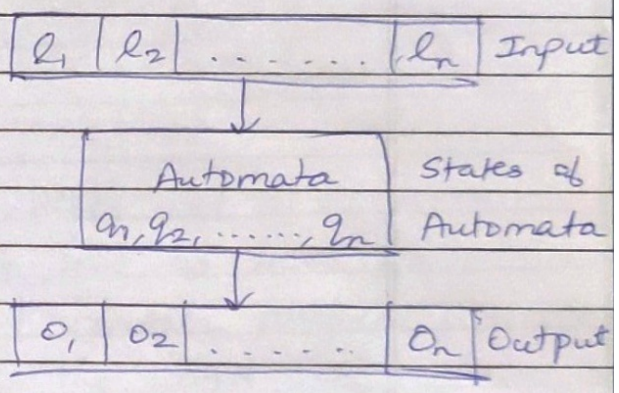
Page No.
Date

Q.26. What is finite state automata?

Finite automata is the simplest machine to recognize patterns. The finite automata is an abstract machine that has five elements or tuples. It has set of states and rules for moving from one state to another, but it depends upon the applied input symbol.

Finite automata have 2 states, Accept state or Reject state. When the input string is processed successfully, and the automata reached its final state, then it will accept.

The given figure shows following features of automata:
Input, Output, States of automata, State relation, Output relation

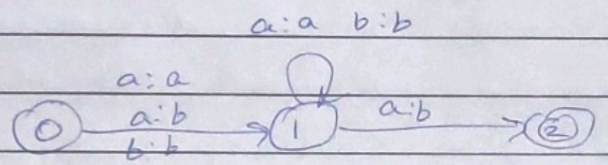


A Finite Automata consists of the following:-
$Q \rightarrow$ Finite set of states.
$\Sigma \rightarrow$ Set of input symbols
$q \rightarrow$ Initial state
$F \rightarrow$ set of Final states
$\delta \rightarrow$ Transition Function

Q.27. Explain finite state transducers with the help of example.

Finite state transducers is finite automation that works on 2 or more tapes. The most common way to think about transducers is as a kind of "translating machine".

They read from one tape and write onto other.



babba → babbb
aba → bbb
aba → abb

The machine moves between states taking an input symbol, while it outputs the corresponding output signal.

It encodes a relation or mapping from a set to another.

Page No.
Date

Q.26. Two Level Morphology :-

Given an input cats, we would like to output cat + N + PL, telling us that cat is a plural noun.

This is done via 2-Level morphology.
It represents the correspondence between lexical and surface levels.
FST is used to map between 2 levels.

FST reads from one tape (input) and writes on another tape (output).

Lexical :	C	A	T	N	PL
Surface	C	A	T	S	

Q.39. Explain the working of Porter Stemmer algorithm.

The Porter Stemmer algorithm is a widely used algorithm for stemming English words.

The algorithm works as follows :-

① The word is first split into individual characters or groups of characters called syllables.

② The Porter Stemmer applies a set of rules and heuristics to remove common suffixes from the word. These suffixes can include plurals, past participles and gerunds.

③ The algorithm then applies another set of rules to further reduce the word to its base or root form. These rules can include removing certain prefixes or changing vowels or consonants in the word.

④ Finally, the Porter Stemmer algorithm checks to see if the resulting stem is a valid word in the English language. If it is not, the stem is discarded and the original word is used instead.

Page No.
Date

Q.30. 'm' is the no. of Vowel - Consonant (VC) pairs in the given word.

Tree → m = 0
Troubles → m = 2
Recommendation → m = 5
Highlighting → m = 3
Calling → m = 2

Page No.
Date

Q.31. Explain levels of details of POS tag and its types

The level of details of POS tagging refers to the extent to which different types of grammatical information are included in the tagging process.

There are 2 main levels of details of POS tagging:

1. Coarse-grained :- This level of detail involves assigning a general grammatical category to each word such as noun, verb, adjective or adverb. For example, we can identify a word as noun but we cannot tell whether it is Singular Noun or Plural Noun.

2. Fine-Grained :- This level of detail involves assigning more specific grammatical categories to each word, such as noun (singular), noun (plural), verb (past tense), verb (present participle) and so on.

Q.32. Justify with the help of example that POS tagging is disambiguition task.

POS tagging is a disambiguous task because it involves identifying the correct grammatical category of a word in a sentence, which can often be ambiguous based on context.

For example, book can be a verb (book that flight) or a noun (hand me that book)

Page No.
Date

Q.33. Explain Rule Based POS tagging with the help of example.

Rule-based taggers used dictionary or lexicon for getting possible tags for tagging each word. If the word has more than one possible tag, then rule-based taggers use hand-written rules to identify the correct tag.

Disambiguation can also be performed in rule-based tagging by analyzing the linguistic features of a word along with its preceding as well as following words.
For eg :- If the preceding word of a word is article, then word must be noun.

Rule-based POS tagging has its two-stage architecture :-

(a) In the first stage, it uses a dictionary to assign each word a list of potential parts-of-speech.
(b) In the second stage, it uses large lists of hand-written disambiguation rules to sort down the list to a single part-of-speech for each word.

For example :- I want to reak a book.

Here, "book" can be a noun or verb.

Rule: Before the word "book", determiner "a" is present, therefore "book" is a noun.

Page No.
Date

Q.34. Explain components of Markov Chain models

Components of Markov chain models include :-

$Q = q_1 q_2 \ldots q_N$ → a set of N states

$A = a_{11} a_{12} \ldots a_{N1} \ldots a_{NN}$ → a transition probability matrix A, each a_{ij} representing the probability of moving from state i to state j such that $\sum_{j=1}^{n} a_{ij} = 1 \ \forall i$

$\pi = \pi_1, \pi_2, \ldots, \pi_N$ → an initial probability distribution over states.
π_i is the probability that the Markov chain will start in state i.
Some states j may have $\pi_j = 0$, meaning that they cannot be initial states.
Also, $\sum_{i=1}^{N} \pi_i = 1$.

Page No.
Date

Q.35. Explain Components of Hidden Markov models

Components of Hidden Markov Models include :-

$Q = q_1 q_2 \ldots q_N$ → a set of N states

$A = a_{11} \ldots a_{ij} \ldots a_{NN}$ → a transition probability matrix A, each a_{ij} representing the probability of moving from state i to state j, such that $\sum_{j=1}^{N} a_{ij} = 1 \ \forall i$

$O = O_1 O_2 \ldots O_T$ → a sequence of T observations, each one drawn from a vocabulary $V = v_1, v_2, \ldots, v$.

$B = b_i(O_t)$ → a sequence of observation likelihoods, also called emission probabilities, each expressing the probability of an observation O_t being generated from a state q_i

$\pi = \pi_1, \pi_2, \ldots, \pi_N$ → an initial probability distribution over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states.
Also, $\sum_{i=1}^{n} \pi_i = 1$.

Page No.
Date

Q.36. Write algorithm for Vertibri HMM.

function Viterbi (observations of len T, state-graph of len N)
returns best-path,
path-prob

create a path probability matrix viterbi [N, T]
for each state s from 1 to N do ; initialization step
viterbi [s, 1] ← $\pi_s * b_s(o_1)$
backpointer [s, 1] ← 0
for each time step t from 2 to T do ; recursion step
for each state s from 1 to N do
viterbi [s, t] ← $\max_{s'=1}^{N}$ viterbi [s', t-1] * $a_{s',s}$ * $b_s(o_t)$

backpointer [s, t] ← $\underset{s'=1}{\overset{N}{\operatorname{argmax}}}$ viterbi [s', t-1] * $a_{s',s}$ * $b_s(o_t)$

bestpathprob ← $\max_{s=1}^{N}$ viterbi [s, T] ; termination step

bestpathpointer ← $\underset{s=1}{\overset{N}{\operatorname{argmax}}}$ viterbi [s, T] ; termination step

bestpath ← the path starting at state bestpathpointer, that follows backpointer[] to states back in time

return bestpath, bestpathprob

$v_{t-1}(i)$: the previous Viterbi path probability from the previous timestep
a_{ij} : the transition probability from previous state q_i to current state q_j.
$b_j(o_t)$: the state observation likelihood of the observation symbol o_t given the current state j.

Page No.
Date

Q.37. Calculate the transition and emission probabilities for the set of sentences below.

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

<S> Mary Jane can see Will <E>
N N M V N

<S> Spot will see Mary <E>
N M V N

<S> Will Jane spot Mary ? <E>
M N V N

<S> Mary will pat Spot <E>
N M V N

	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

Emission Probability

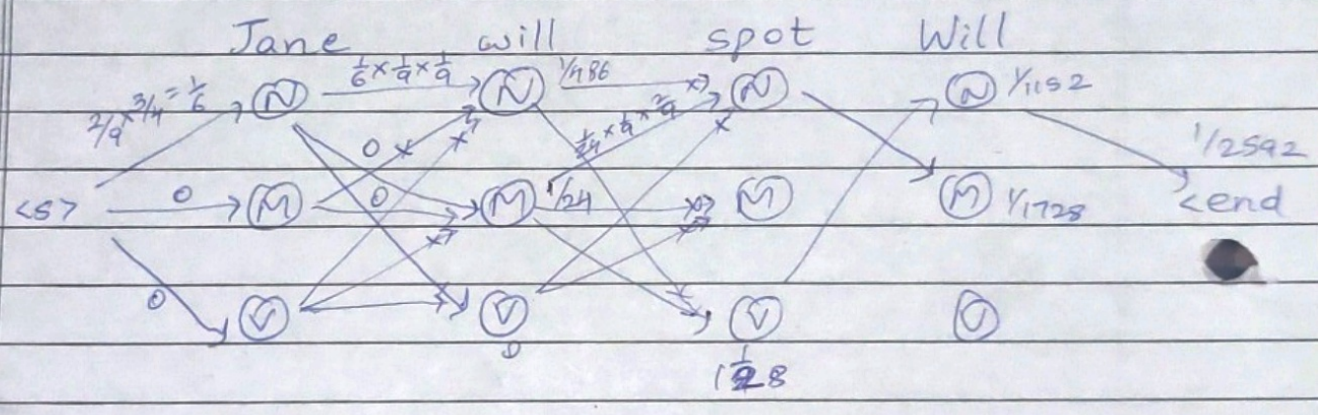
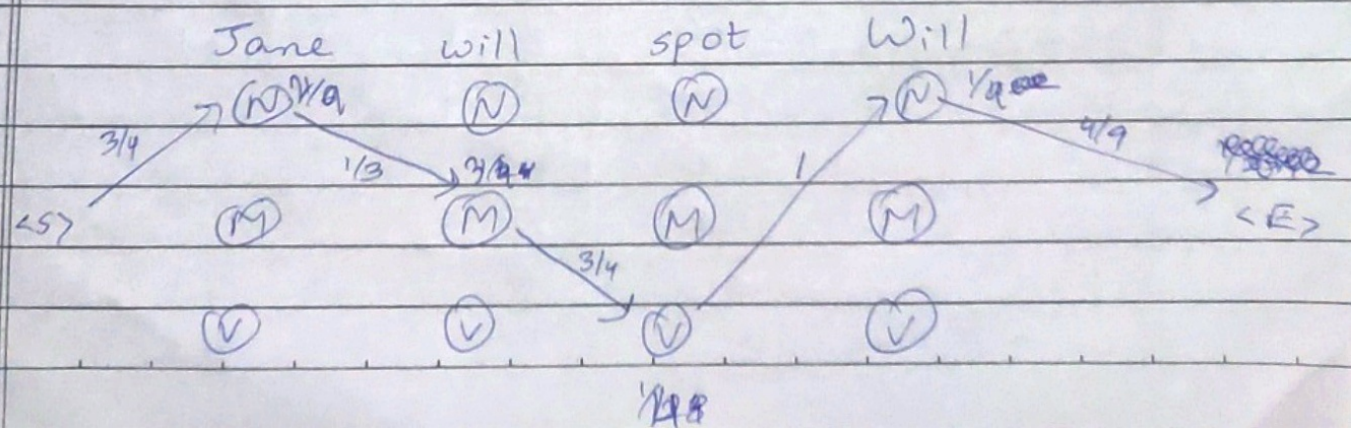
	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	3/9	1/9	4/9
M	1/4	0	3/4	0
V	1	0	0	0

Transition Probability

Q.39. Find the probability of Jane will spot Will (Noun Model - Verb Verb Noun) using the above emission and transition probability matrix.

$$\hat{t}_{1:n} = \underset{t_1, \dots, t_n}{\operatorname{argmax}} \; P(t_1 \dots t_n \mid w_1 \dots w_n)$$

$$\approx \underset{t_1 \dots t_n}{\operatorname{argmax}} \prod_{i=1}^{n} \underbrace{P(w_i \mid t_i)}_{\text{emission}} \cdot \underbrace{P(t_i \mid t_{i-1})}_{\text{transition}}$$



Final Path :- $P = 0.000385$