CBCS SCHEME

USN

21AI643

Sixth Semester B.E. Degree Examination, June/July 2024

Natural Language Processing

Time: 3 hrs. Max. Marks: 100

Note: ***Answer any FIVE full questions, choosing ONE full question from each module.***

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a. Define NLP. List and explain different phases of analysis in NLP with an example for each. **(10 Marks)**

b. Explain the following with suitable example :

i) $\overline{X}$ – theory

ii) Theta theory. **(10 Marks)**

OR

2 a. Define n-gram model. Derive expression for n-gram model. Solve the following test case sentence :

Training sentence :

<s> The Arabian Knights
<s> These are the fairy tales of the cast
<s>The stories of the Arabian Knights are translated in many languages.

Test sentence :

<s> The Arabian Knights are the fairy tales of the cast

Apply the bi-gram model for the test sentence and estimate the probability. **(10 Marks)**

b. What is LFG models? Explain the e-structure and f-structure concept used in language modeling. Find f-structure and c-structure for the following sentence'

she saw stars in the sky

CFG rules :

$S \rightarrow NP\ VP$
$VP \rightarrow V\ \{NP\}\ \{NP\}\ PP^{*}\ \{S'\}$
$PP \rightarrow P\ \ NP$
$NP \rightarrow Det\ \ N\{PP\}$
$S' \rightarrow comp\ S$. **(10 Marks)**

Module-2

3 a. Draw the NFA for the language consisting of all string containing only a's and b's and ending with baa. Draw the state transition table. Find R.E for the above language. **(10 Marks)**

b. Write the algorithm for minimum edit distance and compute the minimum edit distance between "Peaceful" and "Peaceful". **(10 Marks)**

OR

4 a. Write CYK parsing algorithm, Find the sequence of states created by CYK algorithm while parsing the sentence : "The man read this book". Consider the following simplified grammar in CNF.

S → NP VP	Det → that/ this /a / the
S → VP	Noun → book / flight / meal / man
VP → Verb NP	Verb → include /read
NP → Det Noun	AUx → does

(10 Marks)

b. Derive a top-down, depth – first, left – to – right parse tree for the given sentence : 'The angry bear chased the frightened little squirrel'.
Use the following grammar rules to create the parse tree :

S → NPVP	Det → the
NP → Det Nom	Adj → little / angry / frightened
VP → V NP	N → squirrel / bear
Nom → Adj Nom/N	V → chased

(10 Marks)

Module-3

5 a. With a neat diagram, explain the learning framework architecture. (10 Marks)

b. Write a note on :
i) Global security org
ii) Infact system. (10 Marks)

OR

6 a. Explain shortest dependency path hypothesis. Show various shortest dependency path among the relations in the "Jellise created an atmosphere of terror in the camp by killing, abusing and threatening the detainess'. (10 Marks)

b. What is Annotation? Explain the strategies used in active learning approach for acquiring labels using committee based classification scheme. (10 Marks)

Module-4

7 a. Explain latent semantic analysis feedback and topic models feedback system. (10 Marks)

b. With the neat diagram explain the evolutionary model for KDT(Knowledge Discovery from Text). (10 Marks)

OR

8 a. Describe the following with example :
i) iSTART
ii) cohesion and cohesion matrix. (10 Marks)

b. Explain SVM (Support Vector Machine) learning method in sequence model estimation. (10 Marks)

Module-5

9 a. Explain the architecture of an information retrieval system with a neat diagram. **(10 Marks)**

b. A user submitted a query to an IR system. Out of the first 15 documents returned by the system, those ranked 1, 2, 5, 8 and 12 were relevant compute non-interpolated average precision for this retrieval. Assume that the total number of relevant document is 6. **(10 Marks)**

OR

10 a. Define the following with respect to IR
i) wordNet
ii) frameNet. **(10 Marks)**

b. Explain the following classical model with example :
i) Boolean model
ii) Vector space model. **(10 Marks)**

* * * * *

Module 1

1.a What is NLP? Explain the different levels of NLP with eg. (10M) , 5 -Exp,5-Examples

Language is the Primary means of communication used by human beings. It is the tool used to express our ideas and emotions. It shapes thoughts, has a structure and carries meaning.

Natural Language Processing is concerned with development of computational models of aspects of human language processing.

The two main reasons for this development are:

- To develop automated tool for language processing.
- To gain a better understanding of human communication.

To build computational models with human language-processing abilities requires a knowledge of how human acquire, store, and process language. It also requires a knowledge of the world and of language. The two major approaches to NLP -

1. Rationalist Approach
2. Empiricist Approach

I. **Rationalist Approach:**

- Early NLP research is based on rationalist approach, which assumes the existence of some language faculty in human brain.
- Supporters argued that it is not possible for children to learn a complex thing like natural language from limited sensory input.

II. **Empiricist:**

- Empiricists do not believe in existence of language faculty.
- They believe in the existence of some general organization principles such as pattern recognition, generalization and association.
- Learning of detailed structures is therefore possible through the application of these principles on sensory inputs available to the child.

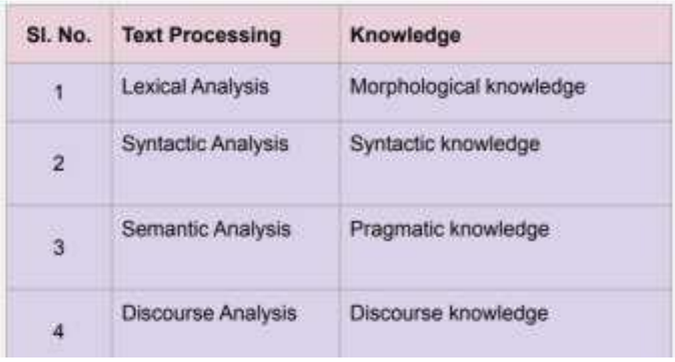
Relation between language and knowledge:

- Language is the medium of expression in which knowledge is deciphered.
- We are considering the text as the language and the content of it as knowledge.
- Language is the outer form of the content it expresses.
- The same content can be expressed in different languages.
- To process a language is to process the content of it.
- As computers are not able to understand natural language, methods are developed to map its content in a formal language

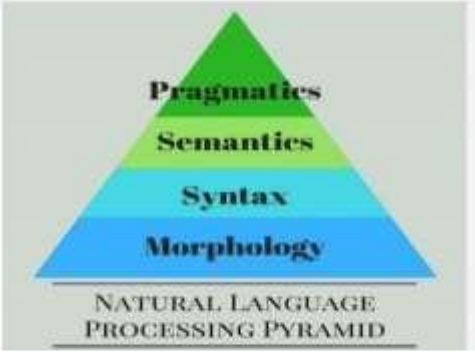
Language Processing

Language or text processing has different levels and each level involves different types of knowledge.

Various levels of processing text and the types of knowledge it involves



Sl. No.	Text Processing	Knowledge
1	Lexical Analysis	Morphological knowledge
2	Syntactic Analysis	Syntactic knowledge
3	Semantic Analysis	Pragmatic knowledge
4	Discourse Analysis	Discourse knowledge



I. Lexical Analysis:

- The simplest level of analysis is lexical analysis which involves analysis of words. (Lexical - relating to the words or vocabulary of a language)
- Words are the most fundamental unit of any natural language text. ➢ Word-level processing requires morphological knowledge.
- Morphological knowledge - knowledge about the structure and formation of words from basic units called morphemes.
- The rules for forming words from morphemes are language specific

Morphological Analysis:

- Morphological analysis is a field of linguistics that studies the structure of words.
- It identifies how a word is produced through the use of morphemes.
- Morphemes are the smallest meaningful words which cannot be divided further. It is the smallest element of a word that has grammatical function and meaning.
- Free morpheme and bound morpheme are the two types of morphemes.
- A single free morpheme can become a complete word. For instance, a bus, a bicycle, and so forth.
- A bound morpheme, on the other hand, cannot stand alone and must be joined to a free morpheme to produce a word. ing, un, and other bound morphemes are examples.

Unsuccessfull

un success ful
(prefix) (stem) (suffix)

II. Syntactic Analysis:

- Syntactic analysis considers a sequence of words as a unit, usually a sentence and finds its structure.
- Syntactic analysis decomposes a sentence by looking into its constituents (or words) and identifies how they relate to each other.
- It captures grammaticality or non-grammaticality of sentences by looking at constraints like word order, number, and case agreement.
- This level of processing requires syntactic knowledge. Syntactic Knowledge:
- Syntactic knowledge is knowledge about how words are combined to form larger units such as phrases and sentences and what constraints are imposed on them.
- Not every sequence of words results in a sentence - For example, ‘I went to the market’ is a valid sentence whereas ‘went the I market to’ is not

This level of analysis requires detailed knowledge about the rules of grammar

III. Semantic Analysis:

- Semantic analysis is associated with the meaning of the language.
- It is concerned with creating meaningful representation of linguistic inputs.
- Defining meaning components is difficult as grammatically valid sentences can be meaningless.
- Humans apply all sorts of knowledge - lexical, syntactic, semantic, discourse, pragmatic, and world knowledge to arrive at the meaning of a sentence.
- The starting point in semantic analysis is lexical semantics (meaning of words).
- A word can have any number of possible meanings associated with it. But in a given context, only one of those meanings will be applied.
- The meaning of a sentence cannot be composed solely on the basis of meaning of its words.

- Consider the following sentences:
  - Kabir and Ayan are married.
  - Kabir and Suha are married.
- Both sentences have identical structures, and the meaning of individual words are clear. But most of us end up with two different interpretations.
- We may interpret the second sentence to mean that Kabir and Suha are married to each other, but this interpretation does not occur from the first sentence.
- Syntactic structure and compositional semantics fail to explain these interpretations.
- Here we make use of pragmatic information.
- Hence semantic analysis requires pragmatic knowledge besides semantic and syntactic knowledge.
- Pragmatic analysis deals with outside word knowledge, which means knowledge that is external to the documents and/or queries.

IV. Discourse Analysis:

- Discourse-level processing attempts to interpret the structure and meaning of even larger units, e.g., at the paragraph and document level, in terms of words, phrases, clusters and sentences.
- It requires the resolution of anaphoric references and identification of discourse structure.
- It requires discourse knowledge - knowledge of how the meaning of a sentence is determined by preceding sentences. – example, how a pronoun refers to the preceding noun and how to determine the function of a sentence in the text
- Pragmatic knowledge may be needed for resolving the anaphoric references.
- In the following sentences, pragmatic knowledge is required to resolve the anaphoric reference - ‘they’. The district administration refused to give the trade union permission for the meeting because they feared violence. The district administration refused to give the trade union permission for the meeting because they oppose government.
- The highest level of processing is pragmatic analysis, which deals with the purposeful use of sentences in situations.
- It requires knowledge of the world i.e., knowledge that extends beyond the content of the text

1.b Explain the following with suitable example (10 M) , Exp-5, Eg-5

i) X bar theory:

The x bar theory is one of the central concepts in GB. Instead of defining several **phrase structures and the sentence structure** with separate sets of rules, Xbar theory defines them both as **maximal projections of some head.** Hence the entities defined become language independent.

Noun phrase (NP) is the maximal projection of Noun (N)
verb phrase (VP) is the maximal projection of Verb (V)
adjective phrase (AP) is the maximal projection of Adjective (A)
preposition phrase (PP) is the maximal projection of Preposition (P)
They can be represented as head X of their corresponding phrase (where X = {N, V, A, P} Even the sentence structure S’ - which is a projection of sentence can be regarded as the maximal projection of inflection (INFL).
GB envisages projection at 2 levels:

1. The projection of the head at semi-phrase level, denoted by X bar

2. The maximal projection at the phrase level, denoted by the X double barFor sentences – the first level projection is denoted as S and the second level maximal projection is defined as S'.

General Phrase and Sentential Structure:

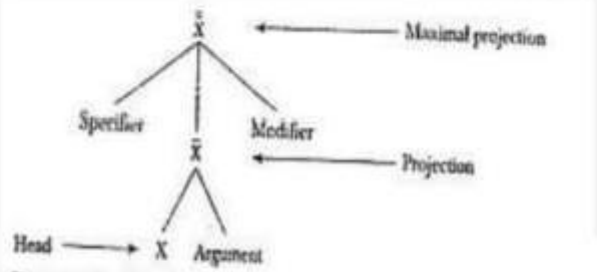
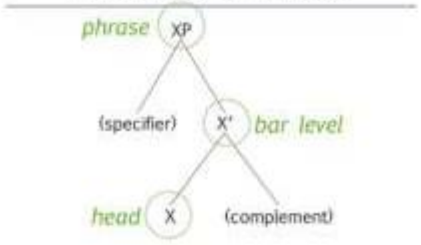


Figure 2.7(a) General phrase and sentential structure

How a phrase is organized

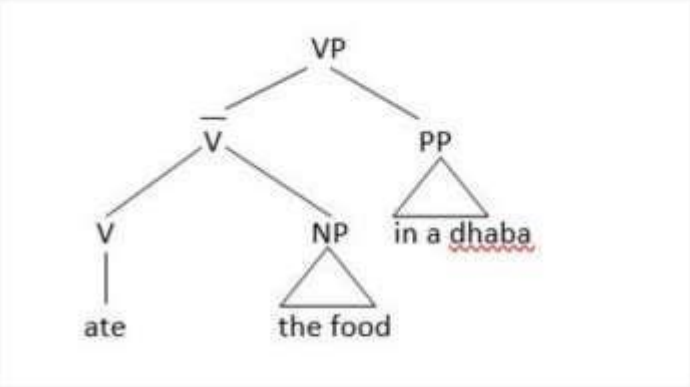


EXAMPLE

2.VP

ate the food in a dhaba

[VP [$_{\bar{V}}$ [$_V$ ate] [$_{NP}$ the food]] [$_{PP}$ in a dhaba]]



ii) Theta Theory

Theta Theory (θ-theory) or the Theory of Thematic Relations

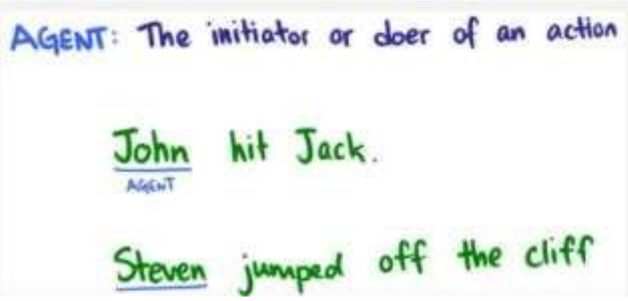
GB puts restrictions on lexical heads through which it **assigns certain roles to its arguments.** These roles are pre-assigned and cannot be violated at any syntactic level as per the projection principle. These role assignments are called **theta roles** and are related to 'semantic selection.

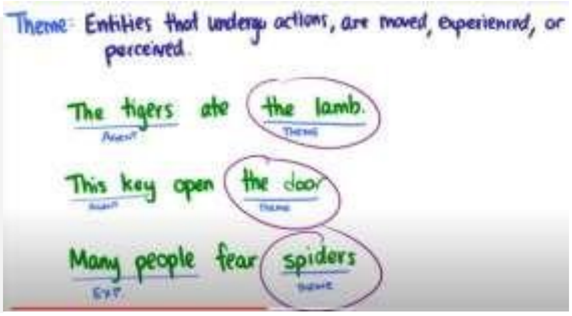
Theta-role and theta-criterion

There are certain thematic roles from which a head can select. These are called theta-roles and they are mentioned in the lexicon.

Ex: The verb 'eat' can take arguments with θ-roles '(Agent-Theme)'.

Agent is a special type of role which can be assigned by a head to outside arguments (external arguments) whereas other roles are assigned within its domain (internal arguments).


AGENT: The initiator or doer of an action
John hit Jack.
AGENT
Steven jumped off the cliff


Theme: Entities that undergo actions, are moved, experienced, or perceived.
The tigers ate the lamb.
Agent
Theme
This key open the door
Agent
Theme
Many people fear spiders
EXP
Theme

2.a Define n-gram model. Derive expression for n-gram model. Solve the below example using bi-gram: (10 M) Exp-5,types-5

- A statistical language model is a probability distribution P(s) over all possible word sequences (or any other linguistic unit like words, sentences, paragraphs, documents, or spoken utterances).
- Statistical Language Modelling is the development of probabilistic models that can predict the next word in the sequence of words given the words that precedes it.
- A number of statistical language models have been proposed in literature. The dominant approach in modelling is the n-gram model

The goal of a statistical language model is to estimate the probability of a sentence. This is achieved by decomposing sentence probability into a product of conditional probabilities using the chain rule as follows:

$P(s) = P(w_1, w_2, w_3, \ldots, w_n)$

$= P(w_1)\, P(w_2/w_1)\, P(w_3/w_1\, w_2)\, P(w_4/w_1\, w_2\, w_3) \ldots P(w_n/\, w_1\, w_2 \ldots w_{n-1})$

$= \prod\ P(w_i / h_i)$

i=1 to n where h_i is the history of the word w_i defined as $w_1\, w_2 \ldots w_{i-1}$

- To find the sentence probability, we need to calculate the probability of a word, given the sequence of words preceding it.
- The n-gram model simplifies the task by approximating the probability of a word given all the previous words by the conditional probability given previous n-1 words only.

- An n-gram model calculates the probability of a word by modeling language as the Markov model of order n-1, that is, by looking at previous n-1 words only.
- A model that limits the history to the previous one word only is termed a bi-gram (n = 1) model. A model that conditions the probability of a word to the previous two words, is called a tri-gram (n = 2) model.

Using bi-gram and tri-gram estimates, the probability of a sentence can be calculated as

$P(s)=\prod\ P\ (w_i\ /\ w_{i-1})$ i=1 to n

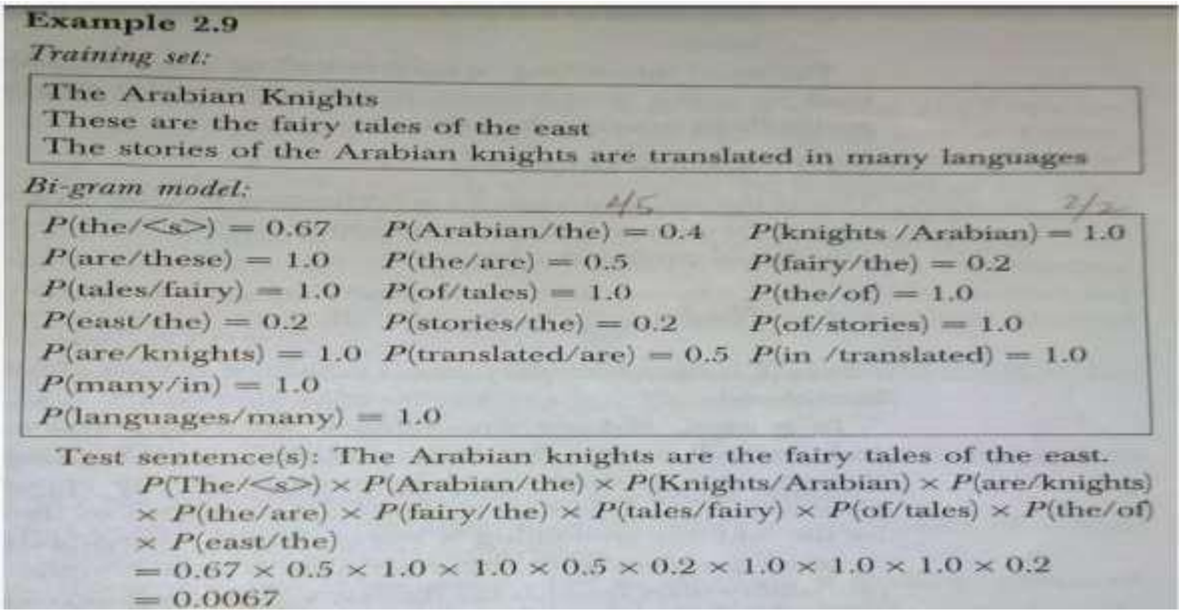
and

$P(s) =\ \prod\ P\ (w_i\ /w_{i-2}\ \ w_{i-1})$

- A special word (pseudoword) <s> is introduced to mark the beginning of the sentence in bi-gram estimation.
- The probability of the first word in the sentence is conditioned on <s>. Similarly, in tri-gram estimation, two pseudo words <s1><s2> are introduced.
- Estimation of probabilities is done by training the n-gram model on the training corpus. n-gram parameters are estimated using the Maximum Likelihood Estimation (MLE) technique, that is, using relative frequencies.
- We count a particular n-gram in the training corpus and divide it by the sum of all n-grams that share the same prefix.

$$\boxed{P\ (w_i\ /\ w_{i-n+1}, \ldots, w_{i-1}\ ) \quad = \quad C\ (w_{i-n+1}, \ldots, w_{i-1}, w_i)\ \ /\ C\ (w_{i-n+1}, \ldots, w_{i-1})}$$

Example:



Example 2.9

Training set:

The Arabian Knights
These are the fairy tales of the east
The stories of the Arabian knights are translated in many languages

Bi-gram model:

4/5 2/2

P(the/<s>) = 0.67	P(Arabian/the) = 0.4	P(knights /Arabian) = 1.0
P(are/these) = 1.0	P(the/are) = 0.5	P(fairy/the) = 0.2
P(tales/fairy) = 1.0	P(of/tales) = 1.0	P(the/of) = 1.0
P(east/the) = 0.2	P(stories/the) = 0.2	P(of/stories) = 1.0
P(are/knights) = 1.0	P(translated/are) = 0.5	P(in /translated) = 1.0
P(many/in) = 1.0		
P(languages/many) = 1.0		

Test sentence(s): The Arabian knights are the fairy tales of the east.

$$\begin{aligned} & P(\text{The}/\text{<s>}) \times P(\text{Arabian}/\text{the}) \times P(\text{Knights}/\text{Arabian}) \times P(\text{are}/\text{knights}) \\ & \times P(\text{the}/\text{are}) \times P(\text{fairy}/\text{the}) \times P(\text{tales}/\text{fairy}) \times P(\text{of}/\text{tales}) \times P(\text{the}/\text{of}) \\ & \times P(\text{east}/\text{the}) \\ & = 0.67 \times 0.5 \times 1.0 \times 1.0 \times 0.5 \times 0.2 \times 1.0 \times 1.0 \times 1.0 \times 0.2 \\ & = 0.0067 \end{aligned}$$

2.b What is LFG model? Explain c-structure and f-structure concept in language modelling. Find c-structure and f-structure for the following sentences (10 M) Exp-5, Eg-5

She saw stars in the sky

CFG rules to handle this sentence are:

S -> NP VP

VP -> V {NP} {NP} PP* {S'}

PP -> P NP

NP -> Det N {PP}

S' -> Comp S'

Lexical functional grammar (LFG) is a constraint-based grammar framework in theoretical linguistics. It posts two separate levels of syntactic structure
1. A phrase structure grammar representation of word order and constituency,
2. A representation of grammatical functions such as subject and object, similar to dependency grammar

The term 'lexical functional' is composed of two parts:
1. The lexical part - is derived from the fact that the lexical rules can be formulated to help define the structure of a sentence
2. The functional part - is derived from grammatical functions such as subject and object or roles played by various arguments in a sentence.
LFG represents sentences at two syntactic levels –
1. Constituent structure (c-structure)
2. Functional structure (f-structure)

C-Structure

The c-structure is derived from the phrase and sentence structure syntax.. C-Structure is used for encoding linear order constituency and hierarchical relations.

F-structure

f-structure encodes the information obtained from phrase and sentence structure rules and functional specifications. As the grammatical functional role cannot be derived directly from phrase and sentence structure, functional specifications are annotated as the nodes of c-structure, which when applied to sentences, results in f-structure.

Example: She saw stars in the sky

CFG rules to handle this sentence are:

S -> NP VP

VP -> V {NP} {NP} PP* {S'}

PP -> P NP

NP -> Det N {PP}

S’ -> Comp S

where

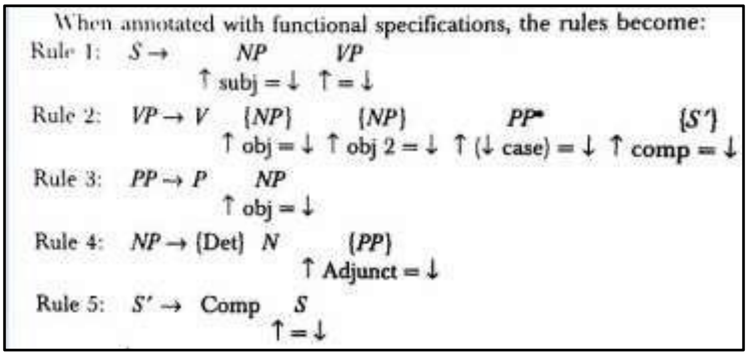
S: sentence; V: verb;

P: preposition, N: noun;

S’: clause; Comp: complement

{} optional;

*: Phrase can appear any number of times including blank.



When annotated with functional specifications, the rules become:

Rule 1: $S \rightarrow \underset{\uparrow \text{subj} = \downarrow}{NP} \quad \underset{\uparrow = \downarrow}{VP}$

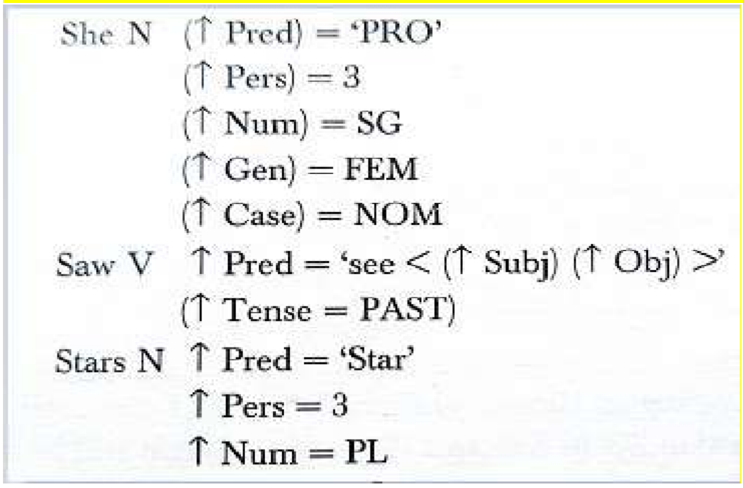
Rule 2: $VP \rightarrow V \quad \underset{\uparrow \text{obj} = \downarrow}{\{NP\}} \quad \underset{\uparrow \text{obj 2} = \downarrow}{\{NP\}} \quad \underset{\uparrow (\downarrow \text{case}) = \downarrow}{PP^{*}} \quad \underset{\uparrow \text{comp} = \downarrow}{\{S'\}}$

Rule 3: $PP \rightarrow P \quad \underset{\uparrow \text{obj} = \downarrow}{NP}$

Rule 4: $NP \rightarrow \{\text{Det}\} \quad N \quad \underset{\uparrow \text{Adjunct} = \downarrow}{\{PP\}}$

Rule 5: $S' \rightarrow \text{Comp} \quad \underset{\uparrow = \downarrow}{S}$

Lexical entries of the words in the sentence - **She saw Stars**



She N ($\uparrow$ Pred) = ‘PRO’
($\uparrow$ Pers) = 3
($\uparrow$ Num) = SG
($\uparrow$ Gen) = FEM
($\uparrow$ Case) = NOM

Saw V $\uparrow$ Pred = ‘see < ($\uparrow$ Subj) ($\uparrow$ Obj) >’
($\uparrow$ Tense = PAST)

Stars N $\uparrow$ Pred = ‘Star’
$\uparrow$ Pers = 3
$\uparrow$ Num = PL

C-Structure of the sentence - She saw stars

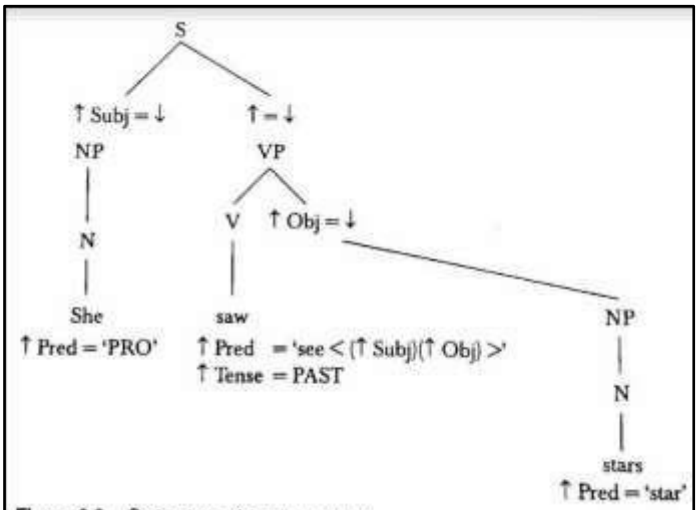
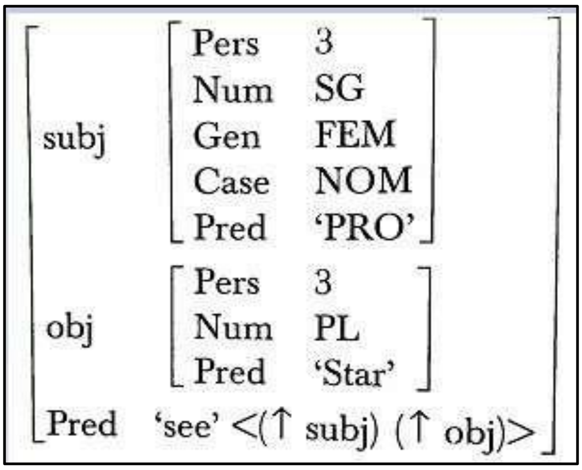


Figure 2.8 C-structure of sentence (2.4)

Finally, the f-structure is the set of attribute-value pairs, represented as



$$\begin{bmatrix} \text{subj} & \begin{bmatrix} \text{Pers} & 3 \\ \text{Num} & \text{SG} \\ \text{Gen} & \text{FEM} \\ \text{Case} & \text{NOM} \\ \text{Pred} & \text{‘PRO’} \end{bmatrix} \\ \text{obj} & \begin{bmatrix} \text{Pers} & 3 \\ \text{Num} & \text{PL} \\ \text{Pred} & \text{‘Star’} \end{bmatrix} \\ \text{Pred} & \text{‘see’} <(\uparrow \text{subj})\ (\uparrow \text{obj})> \end{bmatrix}$$

Module 2

3.a Draw the NFA for the language consisting of all strings containing only 'a's and 'b's and ending with 'baa'. Draw the state transition table. Find the Regular Expression (R.E.) for the above language. (10 Marks)

1. **NFA Construction:**

Let's construct an NFA (Non-deterministic Finite Automaton) for the language that consists of all strings containing only 'a's and 'b's and ending with "baa".

States:

- q0q_0q0: Start state
- q1q_1q1: State after reading 'b'
- q2q_2q2: State after reading 'ba'
- q3q_3q3: Final state after reading 'baa'

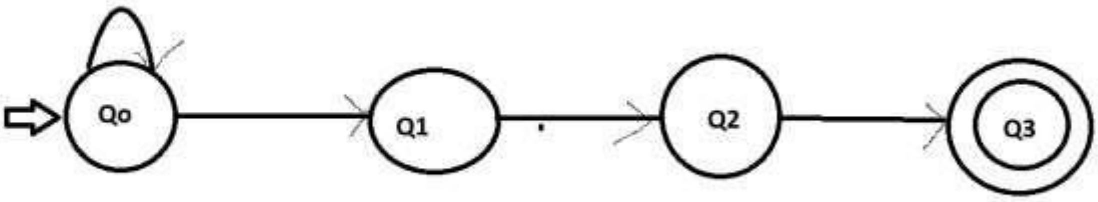
Transitions:

- From q0q_0q0:
  - On 'a', stay in q0q_0q0 (loop)
  - On 'b', move to q1q_1q1
- From q1q_1q1:
  - On 'a', move to q2q_2q2
  - On 'b', stay in q1q_1q1 (loop)
- From q2q_2q2:
  - On 'a', move to q3q_3q3 (final state)
  - On 'b', move to q1q_1q1
- From q3q_3q3:
  - On 'a' or 'b', move back to q0q_0q0 (optional loop, depending on interpretation)

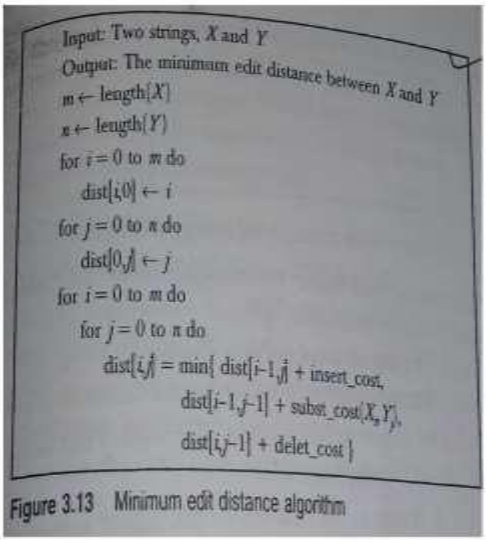
The regular expression for the language can be derived as:

```
R.E. = (a|b)*baa
```

This denotes any combination of 'a's and 'b's followed by the specific string "baa".



3.b Write the algorithm for minimum edit distance and compute the minimum edit distance between "Peaceful" and "Peacefull". (10 Marks)



```
Input: Two strings, X and Y
Output: The minimum edit distance between X and Y
m ← length(X)
n ← length(Y)
for i = 0 to m do
    dist[i,0] ← i
for j = 0 to n do
    dist[0,j] ← j
for i = 0 to m do
    for j = 0 to n do
        dist[i,j] = min{ dist[i-1,j] + insert_cost,
                         dist[i-1,j-1] + subst_cost(X_i,Y_j),
                         dist[i,j-1] + delet_cost }
```

Figure 3.13 Minimum edit distance algorithm

Table for Minimum Edit Distance between "peaceful" and "peaceful":

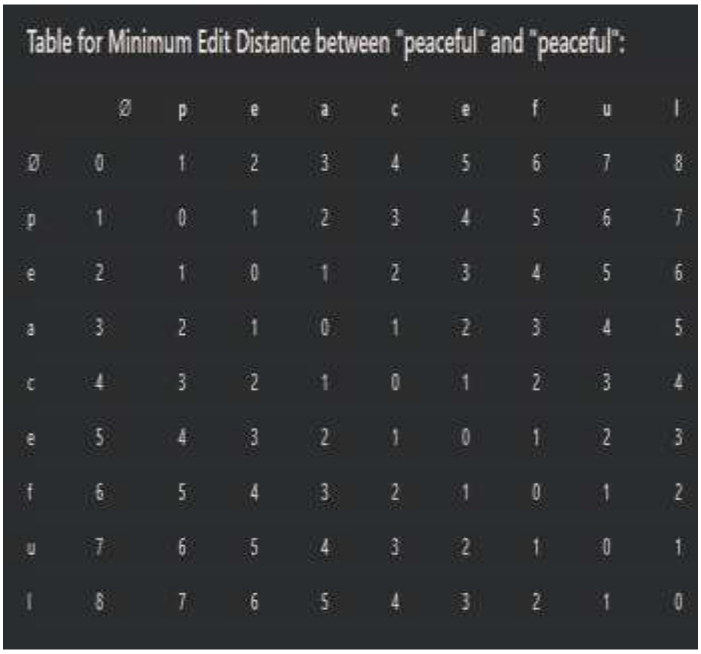
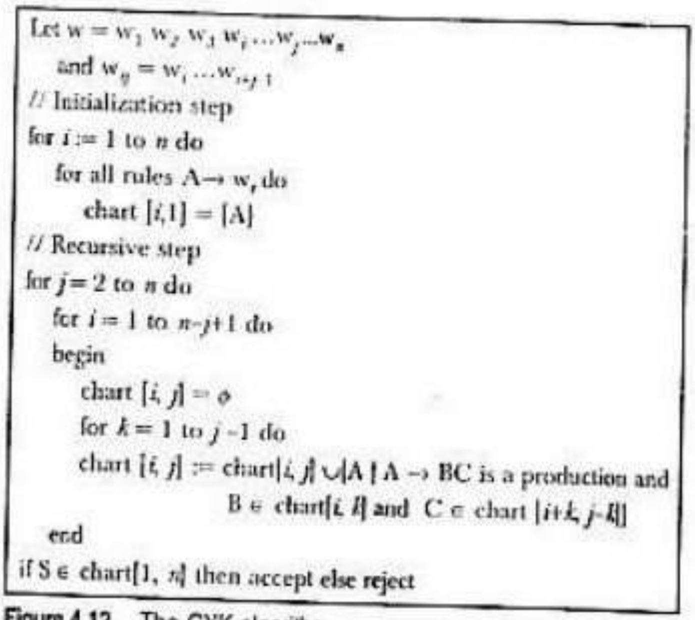


Table for Minimum Edit Distance between "peaceful" and "peaceful":

	Ø	p	e	a	c	e	f	u	l
Ø	0	1	2	3	4	5	6	7	8
p	1	0	1	2	3	4	5	6	7
e	2	1	0	1	2	3	4	5	6
a	3	2	1	0	1	2	3	4	5
c	4	3	2	1	0	1	2	3	4
e	5	4	3	2	1	0	1	2	3
f	6	5	4	3	2	1	0	1	2
u	7	6	5	4	3	2	1	0	1
l	8	7	6	5	4	3	2	1	0

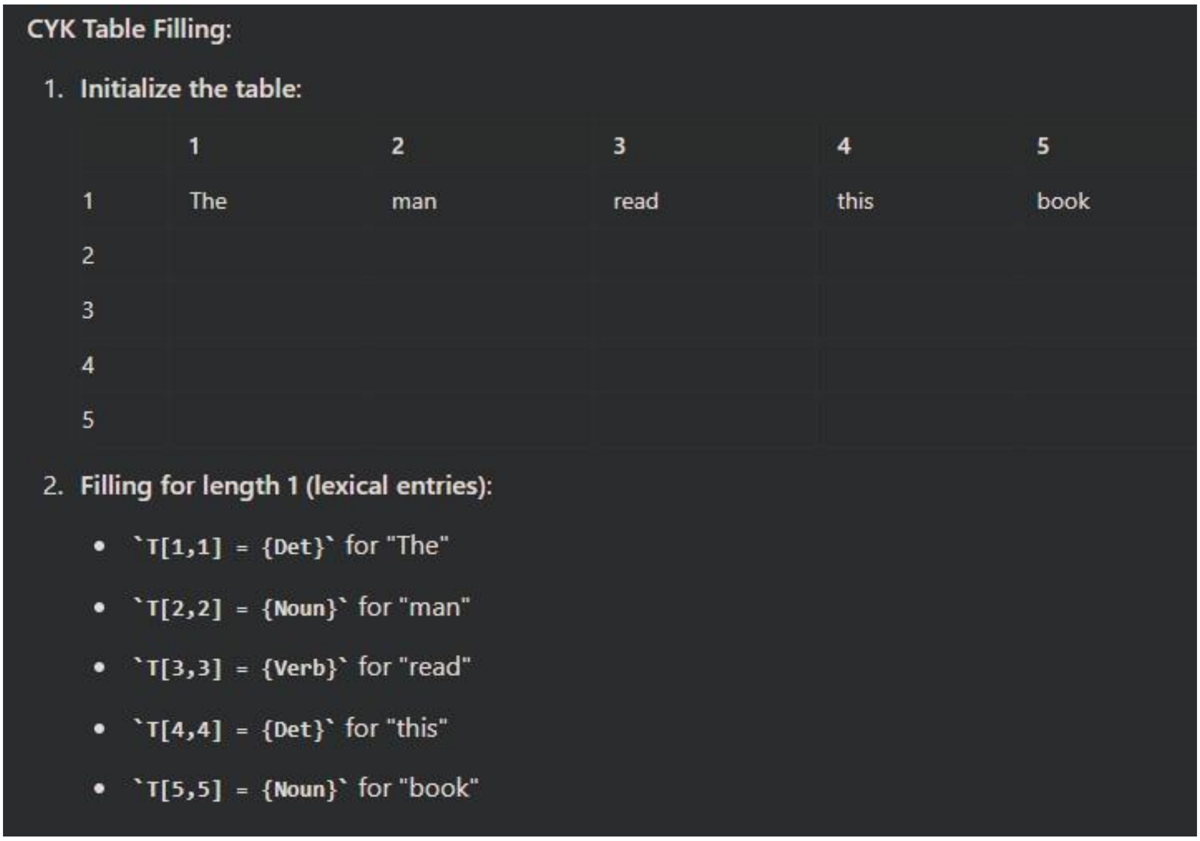
- The first row and column are initialized with values representing the number of edits needed to transform an empty string into the first iii or jjj characters of the respective string.
- Since the two strings are identical, every diagonal value (i.e., dp[i][j]dp[i][j]dp[i][j] when comparing the same characters) is taken from the previous diagonal value without adding any additional cost, hence all these cells are 0.
- The rest of the table entries show the cumulative cost of editing, which is 0 in this case.

4(a): Write the CYK parsing algorithm. Find the sequence of states created by the CYK algorithm while parsing the sentence: "The man read this book". Consider the following simplified grammar in CNF. (10 M)



```
Let $w = w_1\ w_2\ w_3\ w_i \ldots w_j \ldots w_n$
  and $w_{ij} = w_i \ldots w_{i+j-1}$
// Initialization step
for $i := 1$ to $n$ do
  for all rules $A \rightarrow w_i$ do
    chart $[i,1] = \{A\}$
// Recursive step
for $j = 2$ to $n$ do
  for $i = 1$ to $n-j+1$ do
  begin
    chart $[i, j] = \phi$
    for $k = 1$ to $j-1$ do
    chart $[i, j] :=$ chart$[i, j] \cup \{A \mid A \rightarrow BC$ is a production and
                    $B \in$ chart$[i, k]$ and $C \in$ chart $[i+k, j-k]\}$
  end
if $S \in$ chart$[1, n]$ then accept else reject
```

Figure 4.12 The CYK algorithm



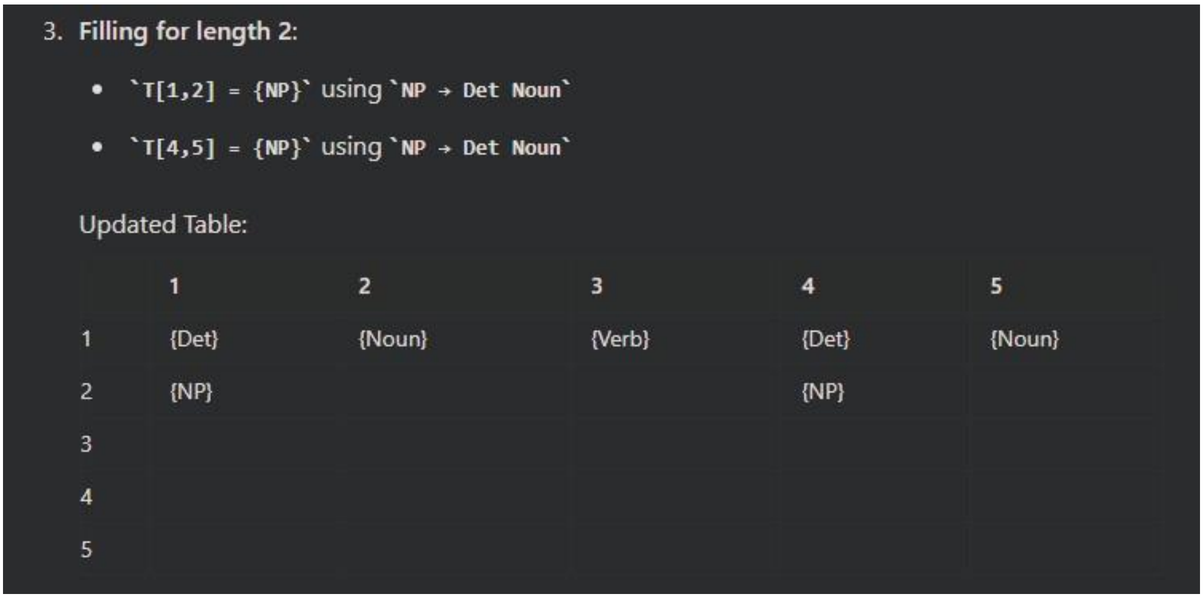
CYK Table Filling:

1. **Initialize the table:**

	1	2	3	4	5
1	The	man	read	this	book
2					
3					
4					
5					

2. **Filling for length 1 (lexical entries):**

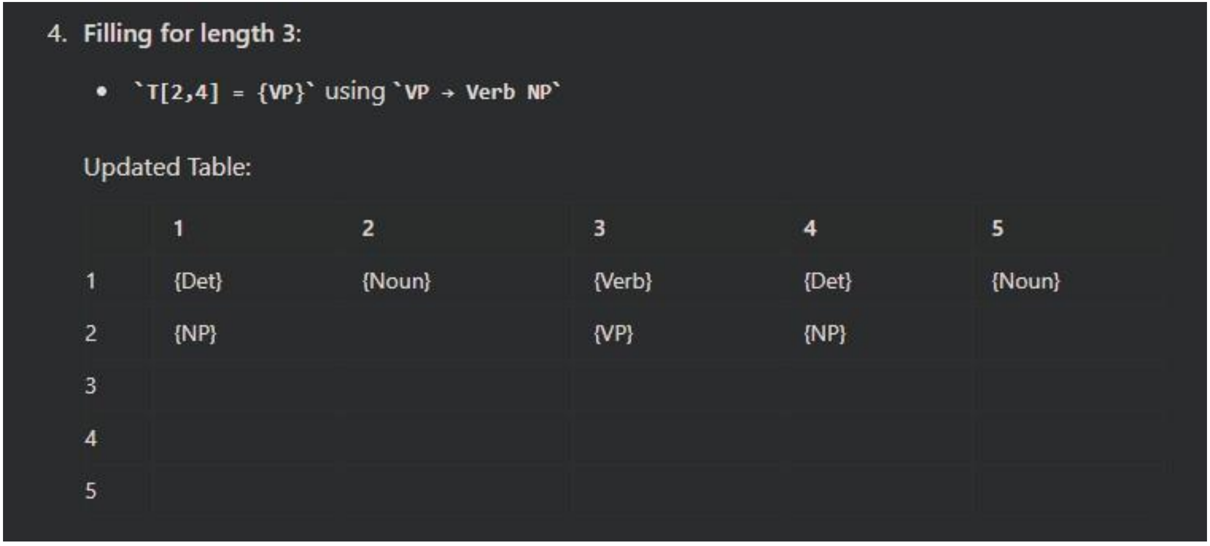
- `T[1,1] = {Det}` for "The"
- `T[2,2] = {Noun}` for "man"
- `T[3,3] = {Verb}` for "read"
- `T[4,4] = {Det}` for "this"
- `T[5,5] = {Noun}` for "book"



3. **Filling for length 2:**
   - `` `T[1,2] = {NP}` `` using `` `NP → Det Noun` ``
   - `` `T[4,5] = {NP}` `` using `` `NP → Det Noun` ``

   Updated Table:

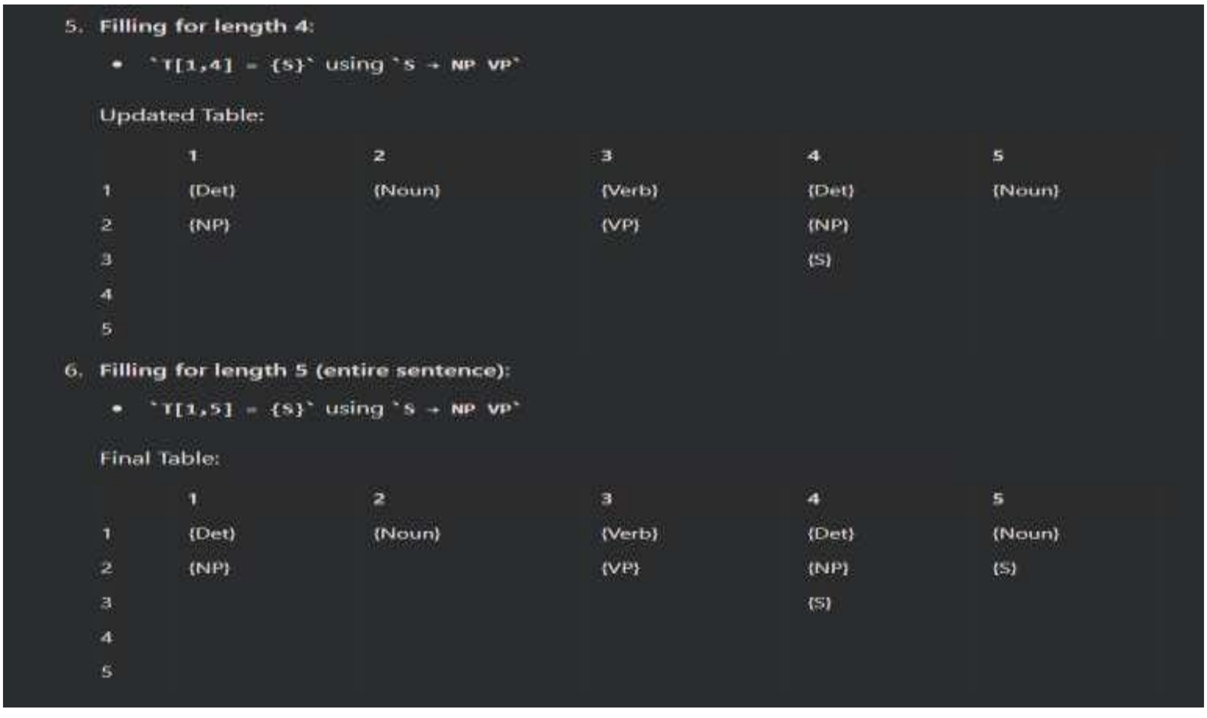
	1	2	3	4	5
1	{Det}	{Noun}	{Verb}	{Det}	{Noun}
2	{NP}			{NP}	
3					
4					
5					



4. **Filling for length 3:**
   - `` `T[2,4] = {VP}` `` using `` `VP → Verb NP` ``

   Updated Table:

	1	2	3	4	5
1	{Det}	{Noun}	{Verb}	{Det}	{Noun}
2	{NP}		{VP}	{NP}	
3					
4					
5					



5. **Filling for length 4:**
   - `` `T[1,4] = {S}` `` using `` `S → NP VP` ``

   Updated Table:

	1	2	3	4	5
1	{Det}	{Noun}	{Verb}	{Det}	{Noun}
2	{NP}		{VP}	{NP}	
3				{S}	
4					
5					

6. **Filling for length 5 (entire sentence):**
   - `` `T[1,5] = {S}` `` using `` `S → NP VP` ``

   Final Table:

	1	2	3	4	5
1	{Det}	{Noun}	{Verb}	{Det}	{Noun}
2	{NP}		{VP}	{NP}	{S}
3				{S}	
4					
5					

4(b): Derive a top-down, depth-first, left-to-right parse tree for the given sentence: "The angry bear chased the frightened little squirrel." Use the following grammar rules to create the parse tree: (10M)

Grammar Rules:

S → NP VP

NP → Det Nom

VP → V NP

Nom → Adj Nom | N

Det → the

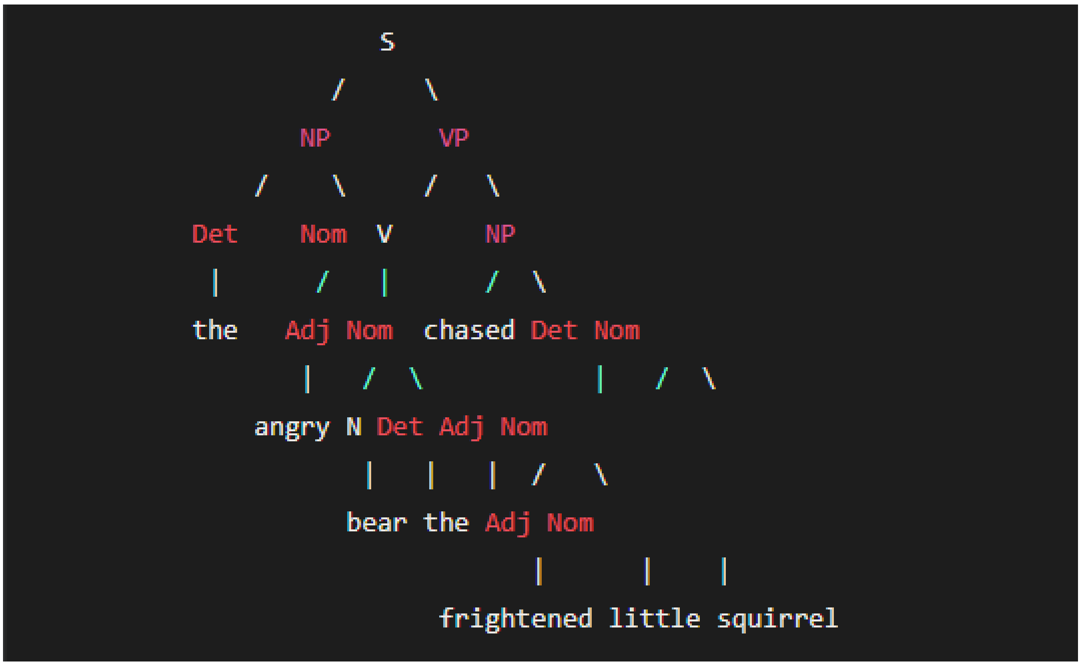
Adj → little | angry | frightened

N → squirrel | bear

V → chased

Parse Tree Construction:

1. Start with the sentence: "The angry bear chased the frightened little squirrel."
2. Apply Grammar Rules:
   - S → NP VP
     - NP → Det Nom (for "The angry bear")
       - Det → the
       - Nom → Adj Nom (for "angry bear")
         - Adj → angry
         - Nom → N (for "bear")
           - N → bear
     - VP → V NP (for "chased the frightened little squirrel")
       - V → chased
       - NP → Det Nom (for "the frightened little squirrel")
         - Det → the
         - Nom → Adj Nom (for "frightened little squirrel")
           - Adj → frightened
           - Nom → Adj Nom (for "little squirrel")
             - Adj → little
             - Nom → N (for "squirrel")
               - N → squirrel



```
            S
         /     \
       NP       VP
    /    \     /   \
Det    Nom  V      NP
 |      /   |      /  \
the   Adj Nom  chased Det Nom
       |   /  \           |   /  \
    angry N Det Adj Nom
           |   |   |  /   \
          bear the Adj Nom
                      |      |    |
                frightened little squirrel
```

Module 3

5 a. With a neat diagram, explain the learning framework architecture.(10 M)

ANS-

Only the preparation of documents is performed outside of this framework

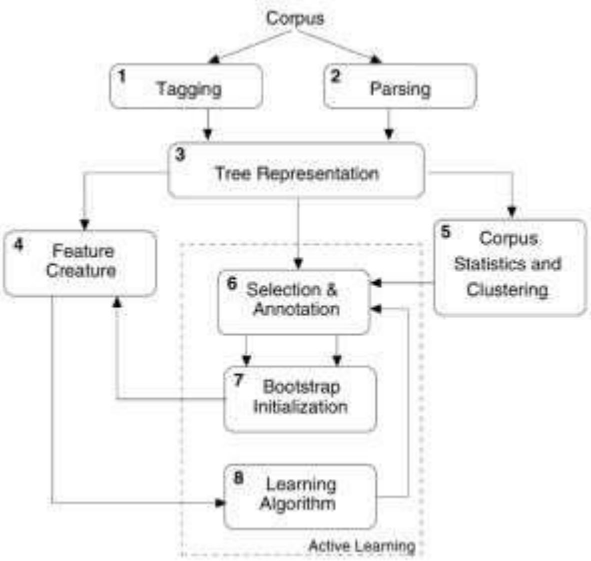


Fig. 4.5. The Learning Framework Architecture.

- **Document Preparation:**

The documents are official diagnostic reports hierarchically structured in several sections and subsections, written by using MS-Word, extracting text from such documents, while preserving the content structure, is a difficult task. A program has been written which that reads the XML document tree and for each section with a specified label extracts the pure text and stores it in a new XML.

- **Tagging:**

The part-of-speech (POS) tagger that we used is a probabilistic tagger with parameter files for tagging several languages: German, English, French, or Italian.

The more problematic issue is that of words marked with the stem <unknown>.

Their POS is usually correctly induced, but we are specifically interested in the stem information.

The two reasons for an <unknown> label are:

a) the word has been misspelled

b) the word is domain specific, and as such not seen during the training of the tagger.

On the positive side, selecting the words with the <unknown> label directly create the list of domain specific words, useful in creating a domain lexicon. Before trying to solve the problem of providing stems for words with the label, one should determine whether the stemming information substantially contributes to the further processing of text. Since we could not know that in advance, we manually provided stems for all words labelled as. Then, during the learning

process we performed a set of experiments, where:

a) no stem information at all was used

b) all words had stem information (tagger + manually created list of stems).

- **Parsing:**

Syntactical parsing is one of the most important steps in the learning framework since the produced parse trees serve as input for the creation of features used for learning. Since we are interested in getting qualitative parsing results, three different parsers are used and they are: the Stanford parser, the BitPar parser and the Sleepy parser.

What these parsers have in common is that they all are based on unlexicalized probabilistic context free grammars (PCFG) trained on the same corpus of German, Negra.

- **Stanford Parser –**

The Stanford parser is an ambitious project that tackles the task of generating parse trees from unlabelled data independently of the language. For the moment, the parser is distributed with parameter files for parsing

English, German, and Chinese. We tested the parser on our data and noticed that the POS tags were often inaccurate induced (in the sentence with only 8 words of), which then resulted in inaccurate parse trees. But, in those cases when the tagging was performed correctly, the parse trees were also correct. Still, the parser could not parse long sentences,

perhaps since it was trained in the part of the Negra corpus with sentences having up to 10 words. Trying the parser with long English sentences instead, produced excellent results.

- **BitPar Parser –**

This parser is composed of two parts, the parser itself and the parameter files (chart rules, lexicon, etc.). Published experimental results claim robust performance, due to the use of sophisticated annotation and transformation

schemata for modelling grammars. Another advantage of the parser is that its lexicon can be extended very

easily with triples of domain dependent words, their tags, their frequency counts in a corpus, thus avoiding the tagging errors typical for unlexicalized parsers. These tagging errors damage the parse results, as can be seen from

the results of the Stanford parser. Our critique for the described BitPar is that it usually produces trees with

more nodes than the other parsers and the annotation of nodes contains specialized linguistic information, not very appropriate for creating features for learning.

- **Sleepy Parser-**

This parser has been specifically tuned for the German language, and while it is a statistical parser like the others, it uses different annotation schemas and incorporates grammatical functions (SB-subject, OC-clausal object, MO-

modifier, HD-head, etc.) or long- distance dependencies between terms. In contrast to the two other parsers, it also has a highly tuned suffix analyser for guessing POS tags (8), which contributes to more accurate tagging results

than the other parsers, although some domain-dependent words are not always correctly tagged. Erroneous parsing is also encountered for very long sentences.

- **Tree Representation:**

The bracketed parse tree and the stem information of tagging serve as input for the step of creating a tree data structure. The tree is composed of terminals (leaf nodes) and non-terminals (internal nodes), all of them known as constituents of the tree. For export purposes as well as for performing exploration or annotation of the corpus.

The tree data structures are stored in XML format. The created tree, when visualized in Tiger Search. The terminals are labelled with their POS tags and contain the corresponding words and stems; the inside nodes are labelled with. their phrase types (NP, PP, etc.), and the branches have labels, too, corresponding to the grammatical functions of the nodes.

- **Feature Creation:**

Features are created from the parse tree of a sentence. A feature vector is created for every constituent of the tree, containing some features unique to the constituent, some features common to all constituents of the sentence, and some others calculated with respect to the target constituent (the predicate verb).

- **Annotation:**

To perform the manual annotation, the Salsa annotation tool (publicly available) is used. The Salsa annotation tool reads the XML representation of a parse tree and displays it . The user can add frames and roles as well as to attach them to a desired target verb. Such an assignment can be easily performed using point-and-click. After this process, an element is added to the XML. representation of the sentence, containing information about the frame.

- **Active Learning:**

Research in IE has indicated that using an active learning approach for acquiring labels from a human annotator has advantages over other approaches of selecting instances for labelling.

5,b. Write a note on: (10 M)

i) **Global security org:**

InFact has been the search behind the GlobalSecurity.org site (www.globalsecurity.org).

GlobalSecurity.org is the most authoritative online destination for those in need of both reliable background information and breaking news.

GlobalSecurity.org's unique positioning enables it to reach both a targeted and large diversified audience.

The content of the website is updated hourly, as events around the world develop, providing in depth coverage of complicated issues.

The breadth and depth of information on the site ensures a loyal repeat audience. This is supplemented by GlobalSecurity.org's unique visibility in the mass media, which drives additional growth. The director of GlobalSecurity.org, John Pike, regularly provides commentary and analysis on space and security issues to PBS, CNN, MSNBC, Fox, ABC, CBS, NBC, BBC, NPR, and numerous print and online publications.

In powering this site, InFact serves the information search needs of a well-established user community of 100,000, consisting of news reporters, concerned citizens, subject matter experts, senior leaders, and junior staff and interns.

Usability Considerations:

In deploying InFact on the GlobalSecurity.org site, our goal was to serve the information needs of a wide community of users, the majority of which are accustomed to straightforward keyword search. Therefore, on this site, by default, InFact acts as a keyword search engine. However, we also started experimenting with ways to progressively migrate users away from keyword search and towards natural language search or "fact search."

Analysis of Query Logs :

We wish to quantify the relative popularity of natural language (Fact) search versus keyword search. In addition, we wish to compare the relative success of alternative strategies we adopted to overcome usability issues. This study of log data reflects four ways users submit a natural language query to InFact:

1) they click on the Hot Search link.

2) they click on a keyword tip.

3) they click on an example in the Query Generator or Help page.

4) they attempt to type an explicit relationship or fact search using the IQL syntax.

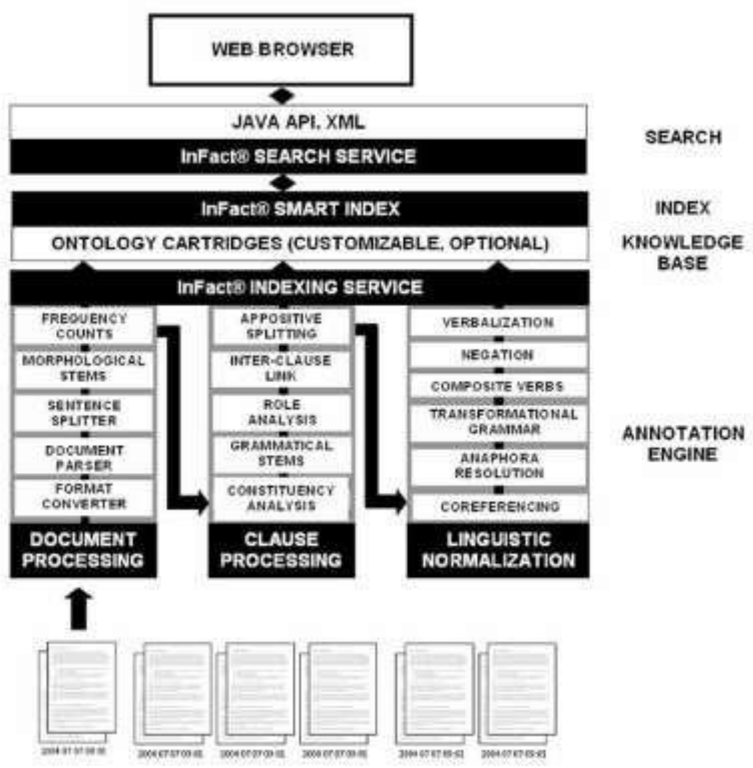
ii) InFact System:

InFact consists of an indexing and a search module.

InFact models text as a complex multivariate object using a unique combination of deep parsing, linguistic normalization and efficient storage.

The storage schema addresses the fundamental difficulty of reducing information contained in parse trees into generalized data structures that can be queried dynamically.

InFact handles the problem of linguistic variation by mapping complex linguistic structures into semantic and syntactic equivalents.



In Fact’s Indexing Service performs in order:

i) Document Processing

The first step in document processing is format conversion, which is handled through native format converters, which can convert 370 different input file types.

Customized document parsers address the issue that a Webpage may not be the basic unit of content, but it may consist of separate sections with an associated set of relationships and metadata.

Next we apply logic for sentence splitting in preparation for clause processing.

Last, we extract morphological stems and compute frequency counts, which are then entered in the index.

ii) Clause Processing

The indexing service takes the output of the sentence splitter and feeds it to a deep linguistic parser.

Indices are created automatically, without using predefined extraction rules, and it captures all information, not just predefined patterns.

The parser performs a full constituency and dependency analysis, extracting part-of-speech (POS) tags and grammatical roles for all tokens in every clause.

Next it captures inter-clause links, through:

○ Explicit tagging of conjunction or pronouns that provide the link between the syntactic structures for two adjacent clauses in the same sentence

○ pointing to the list of annotated keywords in the antecedent and following sentence

iii) Linguistic Normalization

Apply normalization rules at the syntactic, semantic, or even pragmatic level.

Our approach to coreferencing and anaphora resolution make use of syntactic agreement and/or binding theory constraints.

Binding theory places syntactic restrictions on the possible coreference relationships between pronouns and their antecedents.

Example:

■ “John works by himself,” “himself” must refer to John,

■ “John bought him a new car,” “him” must refer to some other individual mentioned in a previous sentence.

Coreferencing and anaphora resolution models also benefit from preferential weighting based on dependency attributes. Apply a transformational grammar to map multiple surface structures into an equivalent deep structure.

A common example is the normalization of a dependency structure involving a passive verb form into the active, and recognition of the deep subject of such clause.

At the more pragmatic level, apply rules to normalize composite verb expressions, capture explicit and implicit negations, or to verbalize noun or adjectives

For instance, the sentences “Bill did not visit Jane,” which contains an explicit negation, and “Bill failed to visit Jane,” where the negation is rendered by a composite verb expression, are mapped to the same structure.

6 a. Explain shortest dependency path hypothesis. Show various shortest dependency path among the relations in the “Jellise created an atmosphere of terror in the camp by killing, abusing and threatening the detainees” (10M)

If e1 and e2 are two entities mentioned in the same sentence such that they are observed to be in a relationship R, then the contribution of the sentence dependency graph to establishing the relationship R (e1, e2) is almost exclusively concentrated in the shortest path between e1 and e2 in the undirected version of the dependency graph.

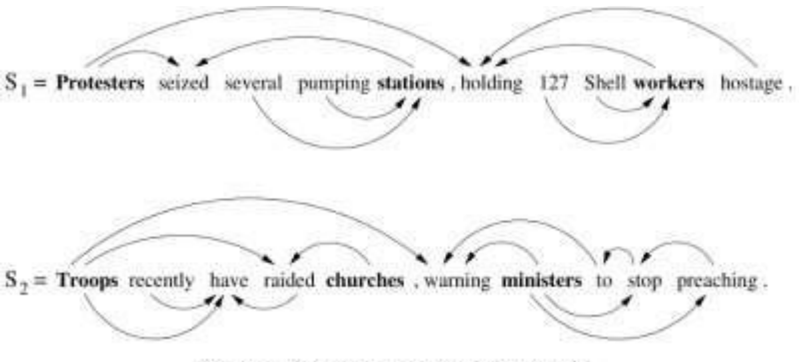

S_1 = Protesters seized several pumping stations , holding 127 Shell workers hostage .
S_2 = Troops recently have raided churches , warning ministers to stop preaching .

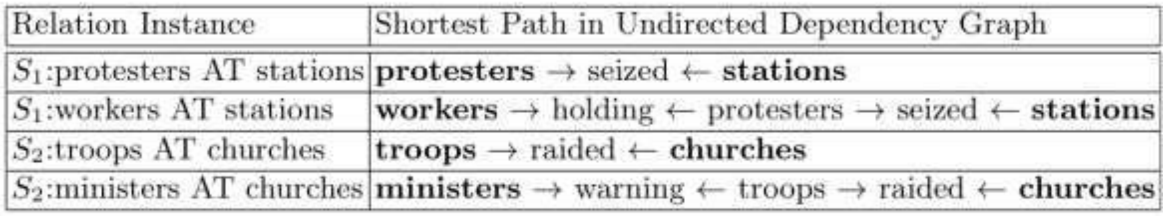
Fig. 3.4. Sentences as dependency graphs.

If entities e1 and e2 are arguments of the same predicate, then the shortest path between them will pass through the predicate, which may be connected directly to the two entities, or indirectly through prepositions.

If e1 and e2 belong to different predicate-argument structures that share a common argument, then the shortest path will pass through this argument.

This is the case with the shortest path between ‘stations’ and ‘workers’, passing through ‘protesters,’ which is an argument common to both predicates ‘holding’ and ‘seized’.

Table 3.1. Shortest Path representation of relations.

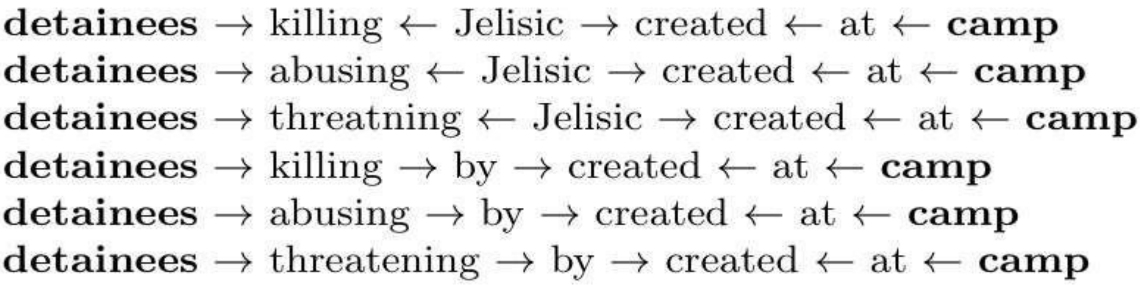


Relation Instance	Shortest Path in Undirected Dependency Graph
S_1:protesters AT stations	**protesters** → seized ← **stations**
S_1:workers AT stations	**workers** → holding ← protesters → seized ← **stations**
S_2:troops AT churches	**troops** → raided ← **churches**
S_2:ministers AT churches	**ministers** → warning ← troops → raided ← **churches**

All these paths support the Located relationship.

For the first path, it is reasonable to infer that if a Person entity (e.g., ‘protesters’) is doing some action (e.g., ‘seized’) to a Facility entity (e.g., ‘station’), then the Person entity is Located at that Facility entity.

The second path captures the fact that the same Person entity (e.g., ‘protesters’) is doing two actions (e.g., ‘holding’ and ‘seized’) , one action to a Person entity (e.g., ‘workers’), and the other action to a Facility entity (e.g., 'station'). A reasonable inference in this case is that the 'workers' are Located at the 'station'.



detainees → killing ← Jelisic → created ← at ← **camp**
detainees → abusing ← Jelisic → created ← at ← **camp**
detainees → threatning ← Jelisic → created ← at ← **camp**
detainees → killing → by → created ← at ← **camp**
detainees → abusing → by → created ← at ← **camp**
detainees → threatening → by → created ← at ← **camp**

To find the shortest dependency paths among the relations in the sentence "Jellisc created an atmosphere of terror in the camp by killing, abusing, and threatening the detainees," we need to identify the relationships between the entities (e.g., "Jellisc," "atmosphere of terror," "camp," "killing," "abusing," "threatening," "detainees") and find the shortest paths in the dependency tree that connect them.

In example there exist multiple shortest paths in the dependency graph between the same two entities– there are actually two different paths, with each path replicated into three similar paths due to coordination. Our current approach considers only one of the shortest paths, nevertheless it seems reasonable to investigate using all of them as multiple sources of evidence for relation extraction.

There may be cases where e1 and e2 belong to predicate-argument structures that have no argument in common. However, because the dependency graph is always connected, we are guaranteed to find a shortest path between the two entities. In general, we shall find a shortest sequence of predicate-argument structures with target predicates P1,P2,...,Pn such that e1 is an argument of P1, e2 is an argument of Pn, and any two consecutive predicates Pi and Pi+1 share a common argument (where by "argument" we mean both arguments and complements).

6 b. What is Annotation? Explain the strategies used in active learning approach for acquiring labels using committee based classification scheme. (10M)

Annotation refers to the process of labeling text data with frames (semantic structures) and roles (the functions that entities play within those frames), which helps in understanding the relationships and meanings within the text.

Research in IE has indicated that using an active learning approach for acquiring labels from a human annotator has advantages over other approaches of selecting instances for labeling.

The possibilities for designing an active learning strategy are manifold; the one we have implemented uses a committee-based classification scheme that is steered by corpus statistics.

The strategy consists of the following steps:

a) Divide the corpus in clusters of sentences with the same target verb. If a cluster has fewer sentences than a given threshold, group sentences with verbs evoking the same frame into the same cluster.

b) Within each cluster, group the sentences (or clauses) with the same parse sub tree together.

c) Select sentences from the largest groups of the largest clusters and present them to the user for annotation.

d) Bootstrap initialization: apply the labels assigned by the user to groups of sentences with the same parse sub-tree. e) Train all the classifiers of the committee on the labeled instances; apply each trained classifier to the unlabeled sentences.

f) Get a pool of instances where the classifiers of the committee disagree and present to the user the instances belonging to sentences from the next largest clusters not yet manually labeled.

g) Repeat steps d)–f) a few times until a desired accuracy of classification is achieved.

Steps a), b), c): In these steps, statistics about the syntactical structure of the corpus are created, with the intention of capturing its underlying distribution, so that representative instances for labeling can be selected.

Step d): This step has been regarded as applicable to our corpus, due to the nature of the text. Our corpus contains repetitive descriptions of the same diagnostic measurements on electrical machines, and often, even the language used has a repetitive nature. Actually, this does not mean that the same words are repeated. Rather, the kind of sentences used to describe the task has the same syntactic structure.

Step e): The committee of classifiers consists of a maximum entropy (MaxEnt) classifier from Mallet, a Winnow classifier from SNoW, and a memory-based learner (MBL) from TiMBL. For the MBL, we selected k=5 as the number of the nearest neighbours. The classification is performed as follows: if at least two classifiers agree on a label, the label is accepted. If there is disagreement, the cluster of labels from the five nearest neighbours is examined. If the cluster is not homogenous (i.e., it contains different labels), the instance is included in the set of instances to be presented to the user for manual labeling.

Step f): If one selects new sentences for manual annotation only based on the output of the committee-based classifier, the risk of selecting outlier sentences is high. Thus, from the instances' set created by the classifier, we select those belonging to large clusters not manually labeled yet.

Module 4

7.a) Explain latent semantic analysis feedback and topic models feedback system. (10M).

Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA) uses statistical computations to extract and represent the meaning of words. Meanings are represented in terms of their similarity to other words in a large corpus of documents. LSA begins by finding the frequency of terms used and the number of co-occurrences in each document throughout the corpus and then uses a powerful mathematical transformation to find deeper meanings and relations among words, very short documents may not be able to receive the full benefit of LSA.

To construct an LSA corpus matrix, a collection of documents are selected. A document may be a sentence, a paragraph, or larger unit of text. A term-document frequency (TDF) matrix X is created for those terms that appear in two or more documents. The row entities correspond to the words or terms (hence the W) and the column entities correspond to the documents (hence the D). The matrix is then analysed using Singular Value

Decomposition (SVD), that is the TDF matrix X is decomposed into the product of three other matrices

(a) vectors of derived orthogonal factor values of the original row entities W.

(b) vectors of derived orthogonal factor values of the original column entities D.

(c) scaling values (which is a diagonal matrix) S.

The product of these three matrices is the original TDF matrix.

{X} = {W}{S}{D}

The dimension (d) of {S} significantly affects the effectiveness of the LSA space for any particular application.

The similarity of terms is computed by taking the cosine of the corresponding term vectors. A term vector is the row entity of that term in the matrix W. In iSTART, the documents are sentences from texts and trainees' explanations of those sentences. These documents consist of terms, which are represented by term vectors; hence, the document can be represented as a document vector which is computed as the sum of the term

vectors of its terms:

$$D_i = \sum_{t=1}^{n} T_{ti}$$

The similarity between two documents (i.e., the cosine between the two document vectors) is computed as:

$$Sim(D1, D2) = \frac{\sum_{i=1}^{d}(D1_i \times D2_i)}{\sum_{i=1}^{d}(D1_i)^2 \times \sum_{i=1}^{d}(D2_i)^2}$$

The rating is based on formulae that use weighted sums of the four LSA cosines between the explanation and each of the four benchmarks.

The four benchmarks include:

a) the words in the title of the passage (“title”).

b) the words in the sentence (“current sentence”).

c) words that appear in prior sentences in the text that are causally related to the sentence (“prior text”), and

d) words that did not appear in the text but were used by two or more subjects who explained the sentence during experiments (“world knowledge”).

when LSA was combined with a fully automated word-based system and we found that world knowledge benchmark could be dropped. Hence, only three benchmarks are used for LSA-based factors:

1) the words in the title of the passage.

2) the words in the sentence, and

3) the words in the two immediately prior sentences.

From the word-based values we include:

4) the number of content words matched in the target sentence.

5) the number of content words matched in the prior sentences.

6) the number of content words matched in the subsequent sentences, and

7) the number of content words that were not matched in 4, 5, or 6.

Topic models

Topic modeling is a technique in natural language processing (NLP) used to identify the underlying topics within a collection of documents. It helps in understanding the structure and themes present in large text datasets. Here are a few examples of topic modeling algorithms:

1. **Latent Dirichlet Allocation (LDA):**

- LDA is a generative probabilistic model that assumes documents are mixtures of topics, and topics are mixtures of words. It helps in discovering hidden topics in a set of documents.
- **Example:** Given a set of news articles, LDA might identify topics such as politics, sports, technology, and health, with each topic represented by a set of words like:
    - Politics: election, government, policy, vote
    - Sports: game, team, player, match
    - Technology: software, computer, internet, device
    - Health: doctor, disease, treatment, medicine

2. **Non-Negative Matrix Factorization (NMF):**
    - NMF factorizes the term-document matrix into two non-negative matrices: one representing the document-topic distribution and the other representing the topic-word distribution.
    - **Example:** In a collection of research papers, NMF might identify topics like machine learning, data science, and biology, with each topic characterized by specific terms:
        - Machine Learning: algorithm, model, training, neural
        - Data Science: data, analysis, statistical, visualization
        - Biology: cell, DNA, protein, organism
3. **Latent Semantic Analysis (LSA):**
    - LSA uses singular value decomposition (SVD) to decompose the term-document matrix and capture the latent relationships between terms and documents.
    - **Example:** Analyzing customer reviews, LSA might reveal topics such as product quality, customer service, and delivery experience, with each topic identified by relevant terms:
        - Product Quality: durable, reliable, excellent, value
        - Customer Service: helpful, support, response, friendly
        - Delivery Experience: fast, timely, package, tracking

7.b)With a neat diagram explain the evolutionary model for KDT(Knowledge discovery from Text). (10M)

Ans-

Semantically guided model for evolutionary Text Mining which is domain-independent but genre-based. Unlike other approaches to Knowledge Discovery from Texts (KDT), this approach does not rely on external resources or descriptions hence its domain independent. Instead, it performs the discovery only using information from the original corpus of text documents and from the training data generated from them.

Genetic Algorithms (Gas) are the central approach to KDT. However, for proper GA-based KDT there are important issues to be addressed including representation and guided operations to ensure that the produced offspring are semantically coherent.

In order to deal with issues regarding representation and new genetic operations so to produce an effective KDT process, our working model has been divided into two phases. The first phase is the preprocessing step aimed to produce both training information for further evaluation and the initial population of the GA. The second phase constitutes the knowledge discovery itself, in particular this aims at producing and evaluating explanatory unseen hypotheses.

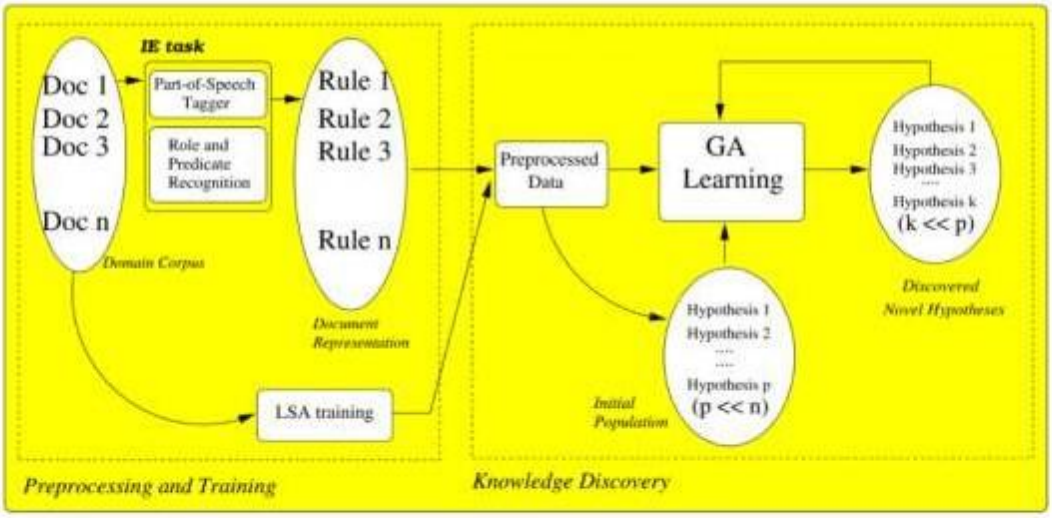


Fig. 9.1. The Evolutionary Model for Knowledge Discovery from Texts

In order to generate an initial set of hypotheses, an initial population is created by building random hypotheses from the initial rules, that is, hypotheses containing predicate and rhetorical information from the rules are constructed. The GA then runs for a number of generations until a fixed number of generations is achieved. At the end, a small set of the best hypotheses are obtained.

8.a)Describe the following with example (10M)

i) iSTART

ii) Cohesion and Cohesion Matrix

i) iSTART

· iSTART (Interactive Strategy Training for Active Reading and Thinking) is an intelligent tutoring system designed to help students improve their reading comprehension skills. A word matching feedback system in iSTART can provide immediate feedback to users based on their input. Here's an algorithm for such a system:

Algorithm: Word Matching Feedback System

1. **Input**:

    o User input text (user_text)

    o Reference text (reference_text)

2. **Preprocessing**:

    o Tokenize user_text and reference_text into words.

    o Convert all tokens to lowercase to ensure case insensitivity.

    o Remove any punctuation or special characters from the tokens.

3. **Word Matching**:

    o Initialize an empty list feedback to store feedback messages.

    o For each word in the reference_text tokens:

        ▪ Check if the word exists in the user_text tokens.

        ▪ If the word exists, append a positive feedback message to feedback.

        ▪ If the word does not exist, append a corrective feedback message to feedback.

4. **Additional Feedback (Optional)**:

    o Calculate the similarity score between user_text and reference_text using a similarity metric (e.g., cosine similarity, Jaccard similarity).

    o Provide additional feedback based on the similarity score (e.g., overall similarity percentage, areas for improvement).

5. **Output**:

    o Return or display the feedback list to the user.

<u>**Psuedocode**</u>

```
def word_matching_feedback_system(user_text, reference_text): # Preprocessing

 user_tokens = preprocess_text(user_text)
 reference_tokens = preprocess_text(reference_text)


 # Word Matching
 feedback = []
 for word in reference_tokens:
 if word in user_tokens:
 feedback.append(f"Good job! You used the word '{word}'.")  else:
 feedback.append(f"Try to include the word '{word}'.")
 # Additional Feedback (Optional)
 similarity_score = calculate_similarity(user_text, reference_text)  feedback.append(f"Overall
similarity score: {similarity_score:.2f}")
 return feedback
def preprocess_text(text):
 # Tokenize, convert to lowercase, and remove punctuation
tokens = text.lower().split()
 tokens = [token.strip('.,!?') for token in tokens]
 return tokens
def calculate_similarity(text1, text2):
 # Calculate similarity score between two texts (e.g., using cosine similarity)  # This is a
placeholder implementation
 return 0.85 # Example similarity score
# Example Usage
user_text = "The cat chased the mouse."
reference_text = "The cat was chasing a mouse."
feedback = word_matching_feedback_system(user_text, reference_text) for message in
feedback:
 print(message)
```

ii) Cohesion and Cohesion Matrix

Cohesion refers to the degree to which elements within a text or set of texts are semantically related or stick together in a meaningful way. In topic modeling, cohesion usually pertains to the semantic similarity of words within a topic. Higher cohesion indicates that the words within a topic are closely related in meaning, which makes the topics more interpretable.

Example:

- A topic with high cohesion might include words like "doctor," "nurse," "hospital," and "patient," all related to the theme of healthcare.
- A topic with low cohesion might include unrelated words like "doctor," "car," "software," and "economy," making it harder to interpret the theme.

Cohesion Matrix

A **cohesion matrix** (or coherence matrix) is used to evaluate the quality of the topics generated by a topic model. It provides a numerical representation of the cohesion between different elements (e.g., words within a topic).

Types of Cohesion Measures:

1. **Pointwise Mutual Information (PMI):**
   - PMI measures the association between pairs of words within a topic. Higher PMI values indicate that words co-occur more frequently than would be expected by chance.
   - **Example:** If the words "doctor" and "patient" frequently co-occur in documents about healthcare, their PMI would be high, indicating strong cohesion.
2. **Cosine Similarity:**
   - Cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space, indicating how similar the vectors are.
   - **Example:** In a vector space model, if the word vectors for "computer" and "software" are close to each other, their cosine similarity would be high, suggesting they belong to a cohesive topic.
3. **Word Co-occurrence:**
   - This measures how often words appear together within a certain context window in the text. High co-occurrence rates indicate strong cohesion.
   - **Example:** In a collection of articles about technology, the words "internet" and "browser" might frequently appear together, indicating a cohesive topic.

Applications of Cohesion and Cohesion Matrix:

- **Model Evaluation:** Assessing the quality and interpretability of the topics generated by a topic model.
- **Parameter Tuning:** Helping in the selection of optimal parameters for topic modeling algorithms, such as the number of topics.

- **Comparison of Models:** Comparing different topic modeling approaches or algorithms to determine which provides more coherent and meaningful topics.

8.b)Explain SVM(Support Vector Machine) learning in sequence model estimation.(10 M)

In the case of determining the conditioned word probabilities p(w|x, y), words that have been observed in the training data would be assigned lower probabilites than the maximum likelihood estimates, whereas unobserved words would be assigned higher probabilities than their maximum likelihood estimates. Statistical learning methods, e.g., [10, 11], uti?lizing methods of regularization theory, allow us to determine the tradeoff between memorization and generalization more principled than the smoothing techniques mentioned above. The learning method adopted here for estimating the sequence model is a Support Vector Machine[10] (SVM). It is commonly known that Support Vector Machines are well suited for text applications given a small number of train?ing examples [12]. This is an important aspect for the commercial use of the system, since the process of gathering, preparing, and cleaning up training examples is time consuming and expensive.

Support Vector Machines solve a binary classification problem. The SVM score associated with an instance of the considered events is its signed distance to the separating hyperplane in units of the SVM margin. In order to solve multiclass problems, a series of Support Vector Machines have to be trained, e.g., in the case of a one-vs-all training schema, the number of SVMs trained is given by the number of classes. The scores between these different machines are not directly compara?ble and the scores must be calibrated such that at least for a given classification instance the scores are on an equal scale. In this application, the scores not only must be comparable between classes for a given classification instance (page), but also between different classification instances (pages), i.e., the SVM scores must be mapped to probabilities. Platt [13] uses SVM scores that are calibrated to class membership probabilities by adopting the interpretation of the score being propor?tional to the logarithmic ratio of class membership probability. He determines the class membership probability as a funcion of the SVM score by fitting a sigmoid function to the empirically observed class membership probabilities as a function of the SVM score. The fit parameters are the slope of the sigmoid function and/or a translational offset. The latter parameter, given the interpretation of the SVM scores discussed above, is the logarithmic ratio of the class prior probabilities. The method used here [14] fixes the translational offset and only fits the slope parame?ter. In addition, the Support Vector Machines are trained using cost factors for the positive as well as for the negative class and optimize the two costs independently. Empirical studies performed by the authors showed that cost factor optimization in conjunction with fitting the slope parameter of the mapping function from SVM scores to probabilities yields superior probability estimates than fitting the slope and the translational offset without cost factor optimization, fitting the slope and the translational offset with cost factor optimization, and fitting the slope only.

Module 5

9.a) Explain the architecture of an information retrieval system with a neat diagram.(10M)

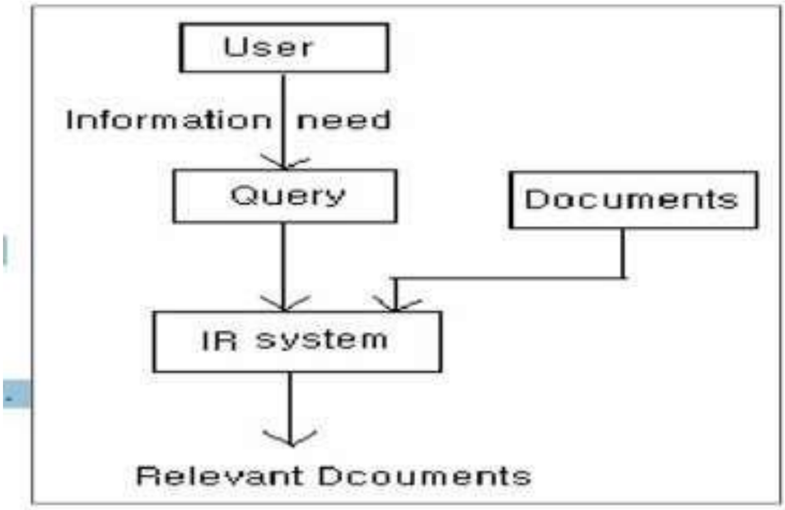
● The process of IR begins with the user’s information need.

● Based on the need the user formulates a query.

● The IR system returns documents that seem relevant to the query.

● The retrieval is performed by matching the query representation with document representation.

● The actual text of the document is not used in the retrieval process. Instead documents in a collection are frequently represented through a set of index terms or keywords.



approaches used in Information Retrieval-

a) Indexing

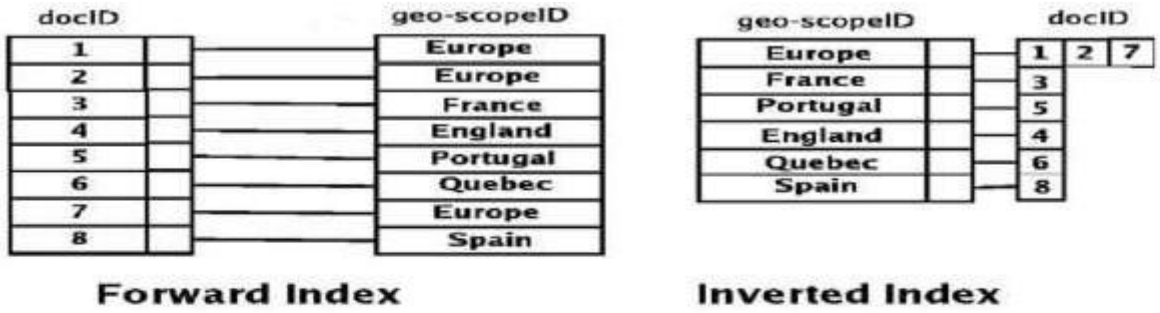
- In a small collection of documents, an IR system can access a document to decide its relevance to a query. However, in a large collection of documents, the same technique poses practical problems.
- A collection of raw documents is usually transformed into an easily accessible representation. This process of transforming a collection of raw documents into an easily accessible representation is known as indexing.
- Indexing technique involves identifying good document descriptors such as keywords or terms.
- Indexing is simply the representation of text (both query and document) as a set of terms whose meaning is equivalent to some content of the original text.
- Indexing is the process of creating data structures that allow for efficient retrieval of information from a large dataset. In the context of information retrieval, indexing involves organizing documents or data in a way that makes it easier and faster to search and retrieve relevant information.

- **Inverted Index:** An inverted index maps terms to the documents that contain them. This allows for quick lookup of documents based on the presence of specific terms.

Algorithm:

- **Tokenization:** Break down documents into individual terms or tokens.
- **Normalization:** Convert tokens to a standard form (e.g., lowercase).
- **IndexCreation:** Create a dictionary where each term points to a list of documents (and optionally, positions within the documents) that contain the term.

Example



Forward Index

Inverted Index

<u>**b) Stop words elimination**</u>

- Stop words elimination is the process of removing common words that are unlikely to be useful in retrieving relevant documents. These words (e.g., "the," "is," "in," "and") occur frequently but do not carry significant meaning in the context of a search query.

Algorithm:

1. **Create a List of Stop Words:** This list contains words to be removed during preprocessing.
2. **Tokenization:** Break down documents into individual terms.
3. **Stop Words Removal:** Remove terms that are present in the stop words list.

Example: Given the document:

"Information retrieval is fun and important"

● Stop words list: ["is", "and"]

● After removal: "Information retrieval fun important"

c) Stemming

- Stemming is the process of reducing words to their base or root form. The goal is to treat words with the same root as identical for retrieval purposes, which helps in matching documents to queries more effectively.

Algorithm:

Step1:Removecommonsuffixes (e.g., "ing," "ed," "s"). **Step2:**Applytransformations to handle plural forms and past tense. ○

Step3: Further reductions to handle derivational suffixes.

Step4: Final cleanup to ensure the word is in its simplest form

Example:

Given the words: "running," "runner," "runs"

Stemmer reduces these to: "run," "run," "run"

9.b) A user submitted a query to an IR system. Out of the first 15 documents returned by the system,those ranked 1,2,5,8,12 were relevant compute non-interpolated average precision for this retrieval. Assume that total number of relevant document is 6.(10 M)

To compute the **non-interpolated average precision (AP)**, we need to consider the positions of the relevant documents in the retrieved list. The formula for non-interpolated AP is:

AP=1total number of relevant documents×∑k=1nPrecision@k×rel(k)\text{AP} = \frac{1}{\text{total number of relevant documents}} \times \sum_{k=1}^{n} \text{Precision@k} \times \text{rel}(k)AP=total number of relevant documents1×k=1∑n Precision@k×rel(k)

where:

- Precision@k\text{Precision@k} , Precision@k is the precision at rank k,
- rel(k)\text{rel}(k) ,rel(k) is a binary function that equals 1 if the document at rank k is relevant, and 0 otherwise,
- nn is the total number of documents retrieved (15 in this case),
- The total number of relevant documents is given as 6.

The relevant documents are at ranks: 1, 2, 5, 8, and 12.

Steps:

1. **Compute Precision@k** for each relevant document:
    - Precision at rank 1: 11=1\frac{1}{1} = 111=1

- Precision at rank 2: 22=1\frac{2}{2} = 122=1
- Precision at rank 5: 35=0.6\frac{3}{5} = 0.653=0.6
- Precision at rank 8: 48=0.5\frac{4}{8} = 0.584=0.5
- Precision at rank 12: 512≈0.4167\frac{5}{12} \approx 0.4167125≈0.4167

2. **Sum of precisions for relevant documents:**

1+1+0.6+0.5+0.4167≈3.51671 + 1 + 0.6 + 0.5 + 0.4167 \approx 3.51671+1+0.6+0.5+0.4167≈3.5167

3. **Compute non-interpolated AP:**

AP=16×3.5167≈0.5861\text{AP} = \frac{1}{6} \times 3.5167 \approx 0.5861AP=61 ×3.5167≈0.5861

The non-interpolated average precision for this retrieval is approximately **0.5861**.

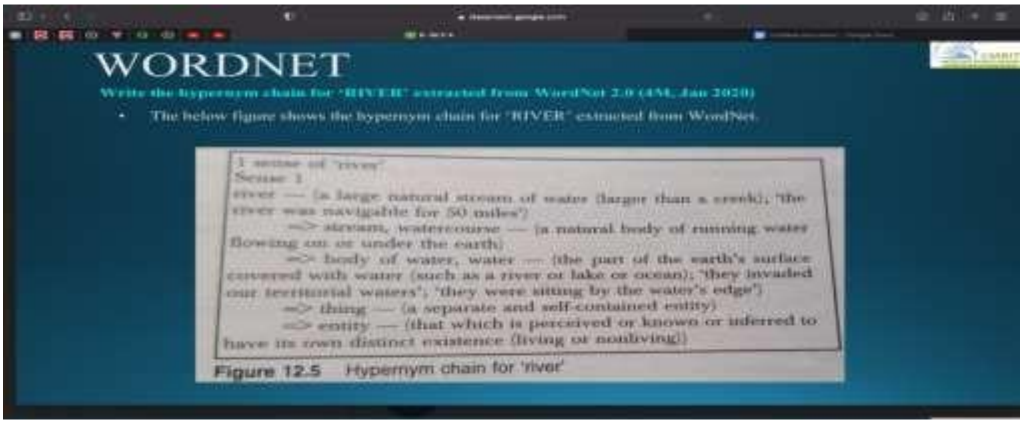
10.a) Define the following with respect to IR

(10M) i)Wordnet

ii)Framenet

<u>i)Wordnet</u>

WordNet is a large lexical database for the English language, developed and maintained at the Cognitive Science Laboratory, Princeton University, under the direction of George A. Miller. Inspired by psycholinguistic theories, WordNet organizes information into three databases: one for nouns, one for verbs, and one for both adjectives and adverbs. The database is structured around sets of synonymous words called synsets, each representing a base concept. These synsets are interconnected by lexical and semantic relations, where lexical relations occur between word forms (senses) and semantic relations occur between word meanings. These relations include synonymy, hypernymy/hyponymy, antonymy, meronymy/holonymy, and troponymy. A word can appear in multiple synsets and parts of speech, with its different meanings, or senses, each associated with a specific synset. WordNet's sense-entries consist of a set of synonyms and a gloss, providing a comprehensive representation of word meanings and their relationships.



WORDNET

Write the hypernym chain for 'RIVER' extracted from WordNet 2.0 (4M, Jan 2020)

- The below figure shows the hypernym chain for 'RIVER' extracted from WordNet.

1 sense of 'river'
Sense 1
river — (a large natural stream of water (larger than a creek); "the river was navigable for 50 miles")
=> stream, watercourse — (a natural body of running water flowing on or under the earth)
=> body of water, water — (the part of the earth's surface covered with water (such as a river or lake or ocean); "they invaded our territorial waters"; "they were sitting by the water's edge")
=> thing — (a separate and self-contained entity)
=> entity — (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Figure 12.5 Hypernym chain for 'river'

ii) Framenet

FrameNet is a large database of semantically annotated English sentences. It is based on principles of frame semantics. The basic philosophy involved is that each word evokes a particular situation with particular participants. FrameNet aims at capturing these situations through case-frame representation of words (verbs, adjectives, and nouns). It defines a tagset of semantic roles called frame elements. The word that invokes a frame is called the target word or predicate, and the participant entities are defined using semantic roles, which are called frame elements. Sentences from the British National Corpus are tagged with these frame elements. Each frame contains a main lexical item as the predicate and associated frame-specific semantic roles, such as AUTHORITIES, TIME, and SUSPECT in the ARREST frame. The FrameNet ontology can be viewed as a semantic level representation of predicate argument structure.

Example 1: The sentence below is annotated with the semantic roles AUTHORITIES and SUSPECT.

- [Authorities The police] nabbed [Suspect the snatcher].

Example 2: The COMMUNICATION frame has the semantic roles ADDRESSEE, COMMUNICATOR, TOPIC, and MEDIUM.

- A JUDGEMENT frame contains roles such as JUDGE, EVALUEE, and REASON.

Example 3: The sentence below is annotated with the roles JUDGE, EVALUEE, and REASON.

- [Judge She] [Evaluee blames the police] [Reason for failing to provide enough protection].

Example 4: A frame may inherit roles from another frame. For example, a STATEMENT frame may inherit from a COMMUNICATION frame and contain roles such as SPEAKER, ADDRESSEE, and MESSAGE.

- [Speaker She] told [Addressee me] [Message ‘I’ll return by 7:00 pm today’].

FrameNet data can be used for:

- Automatic semantic parsing
- Information extraction
- Question answering system
- Information retrieval
- Machine translation
- Text summarization

10.b) Explain the following classical models with example (10M)

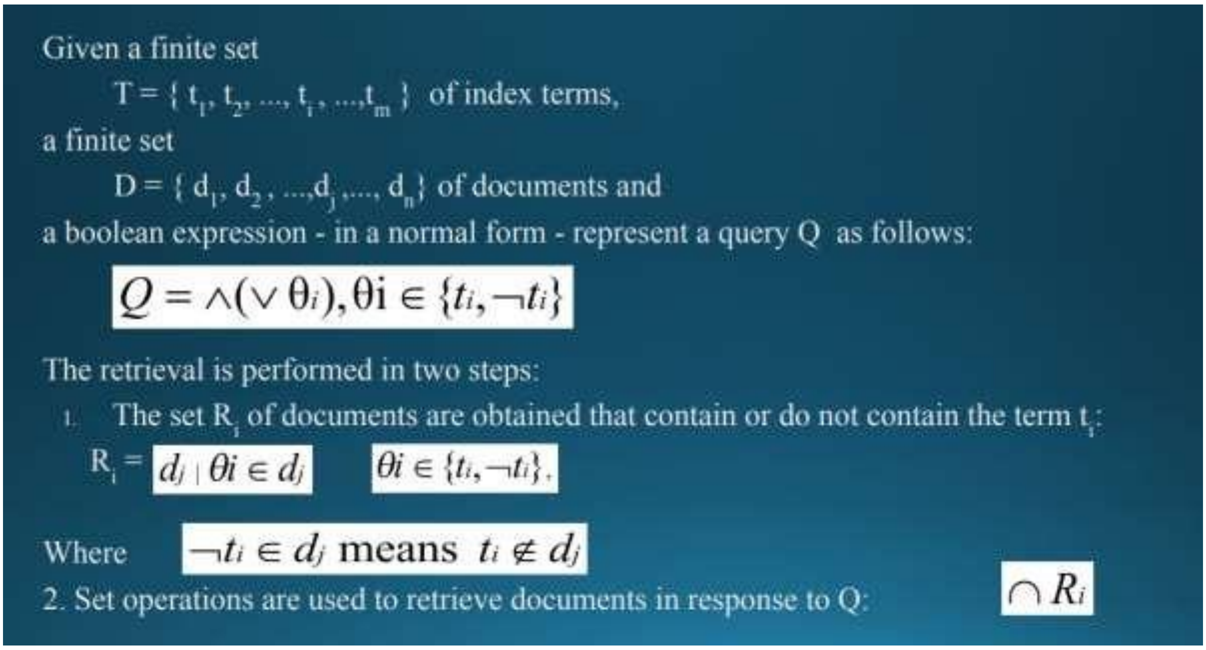
i) Boolean Model

Boolean model is the oldest of the three classical models. It is based on Boolean logic and classical set theory.

• In this model documents are represented as a set of keywords, usually stored in an inverted file.

• An inverted file is a list of keywords, and identifiers of the documents in which they occur.
• Users are required to express their queries as a boolean expression consisting of keywords connected with boolean logical operators (AND, OR, NOT).

• Retrieval is performed based on whether or not document contains the query terms



Given a finite set

$T = \{ t_1, t_2, ..., t_i, ..., t_m \}$ of index terms,

a finite set

$D = \{ d_1, d_2, ..., d_j, ..., d_n \}$ of documents and

a boolean expression - in a normal form - represent a query Q as follows:

$$\boxed{Q = \wedge(\vee \theta_i), \theta i \in \{t_i, \neg t_i\}}$$

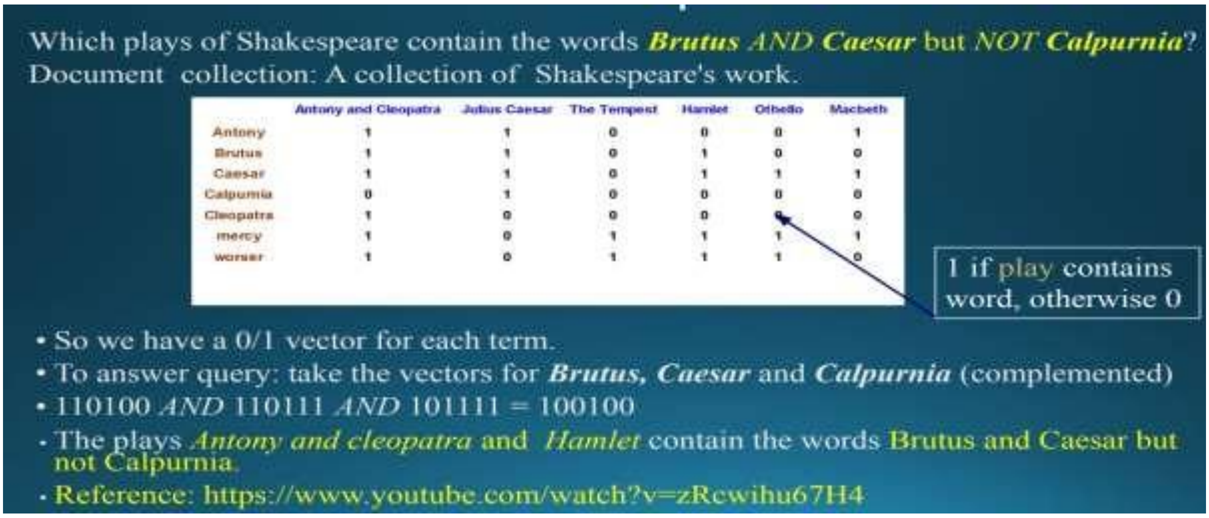
The retrieval is performed in two steps:

1. The set R_i of documents are obtained that contain or do not contain the term t_i:

$R_i = \boxed{d_j \mid \theta i \in d_j}$ $\boxed{\theta i \in \{t_i, \neg t_i\}}$,

Where $\boxed{\neg t_i \in d_j \text{ means } t_i \notin d_j}$

2. Set operations are used to retrieve documents in response to Q: $\boxed{\cap R_i}$



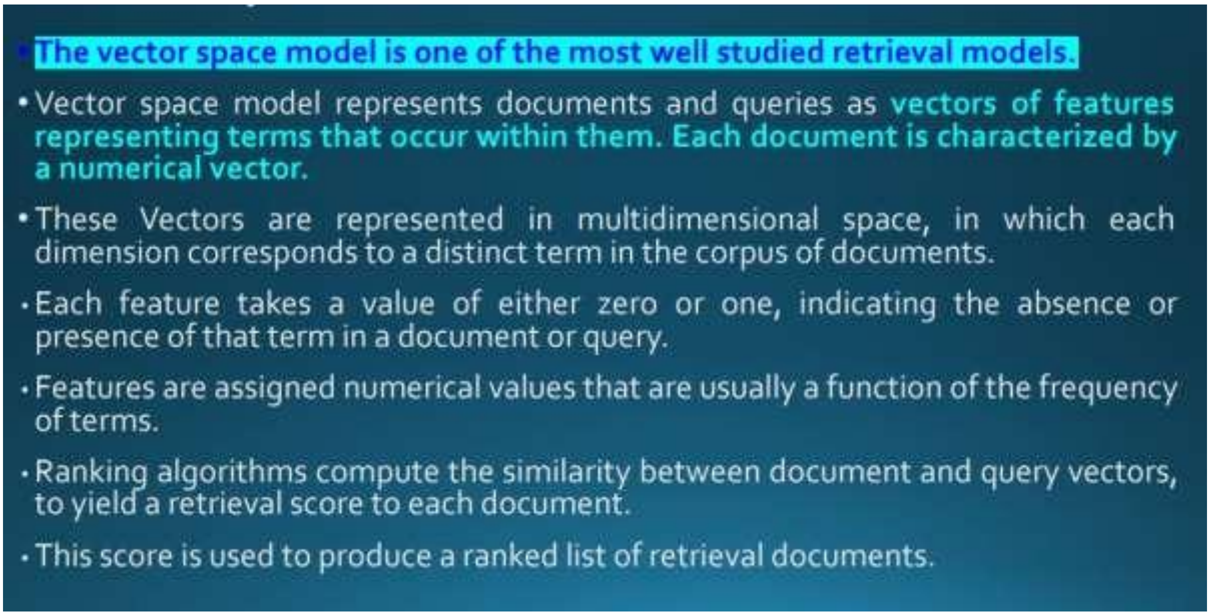
Which plays of Shakespeare contain the words ***Brutus*** *AND* ***Caesar*** but *NOT* ***Calpurnia***?
Document collection: A collection of Shakespeare's work.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

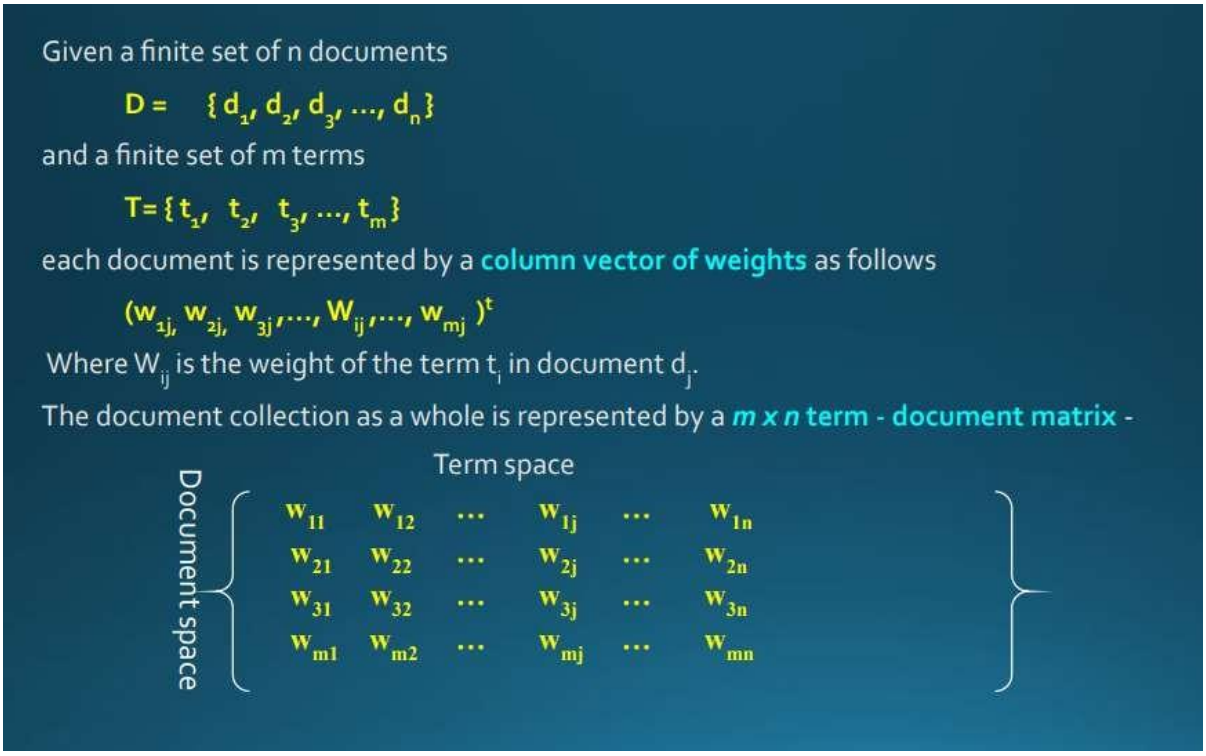
1 if play contains word, otherwise 0

- So we have a 0/1 vector for each term.
- To answer query: take the vectors for ***Brutus, Caesar*** and ***Calpurnia*** (complemented)
- 110100 *AND* 110111 *AND* 101111 = 100100
- The plays *Antony and cleopatra* and *Hamlet* contain the words Brutus and Caesar but not Calpurnia.
- Reference: https://www.youtube.com/watch?v=zRcwihu67H4

ii) Vector Space model



- The vector space model is one of the most well studied retrieval models.
- Vector space model represents documents and queries as vectors of features representing terms that occur within them. Each document is characterized by a numerical vector.
- These Vectors are represented in multidimensional space, in which each dimension corresponds to a distinct term in the corpus of documents.
- Each feature takes a value of either zero or one, indicating the absence or presence of that term in a document or query.
- Features are assigned numerical values that are usually a function of the frequency of terms.
- Ranking algorithms compute the similarity between document and query vectors, to yield a retrieval score to each document.
- This score is used to produce a ranked list of retrieval documents.



Given a finite set of n documents

$D = \{d_1, d_2, d_3, \dots, d_n\}$

and a finite set of m terms

$T = \{t_1, t_2, t_3, \dots, t_m\}$

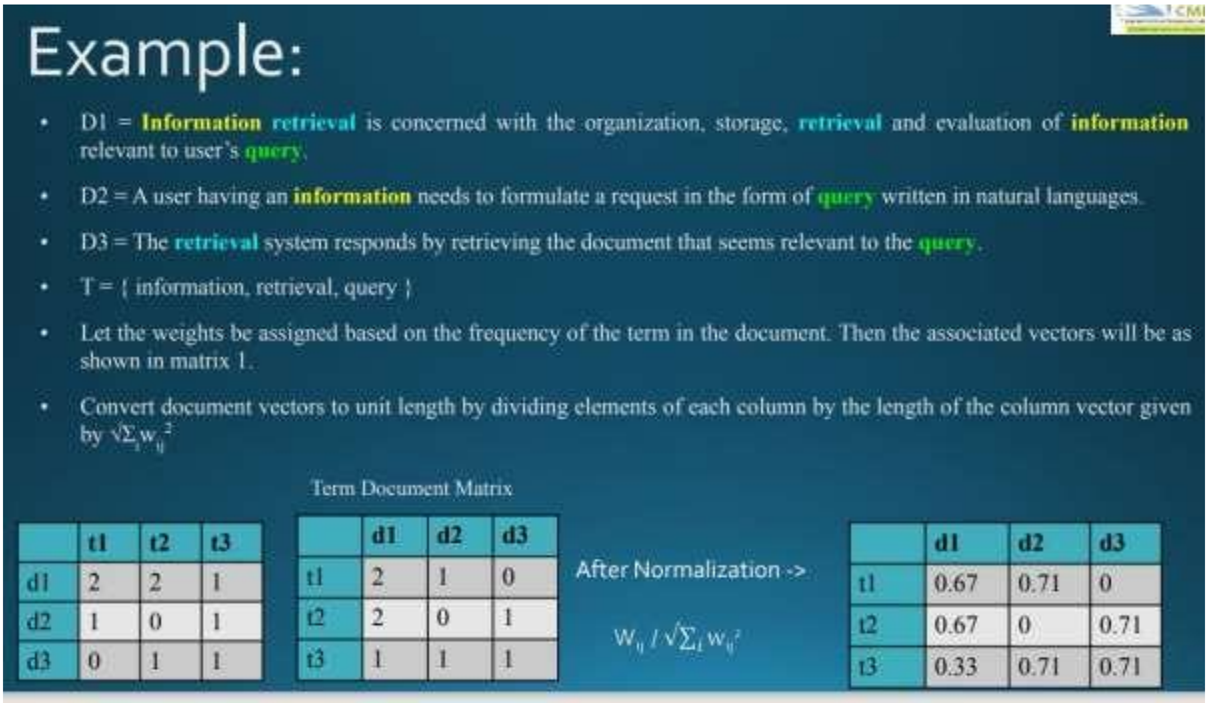
each document is represented by a column vector of weights as follows

$(w_{1j,}\ w_{2j,}\ w_{3j}, \dots, W_{ij}, \dots, w_{mj})^t$

Where W_{ij} is the weight of the term t_i in document d_j.

The document collection as a whole is represented by a *m x n* term - document matrix -

$$\begin{array}{c|c} & \text{Term space} \\ \text{Document space} & \left\{\begin{array}{cccccc} w_{11} & w_{12} & \cdots & w_{1j} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2j} & \cdots & w_{2n} \\ w_{31} & w_{32} & \cdots & w_{3j} & \cdots & w_{3n} \\ w_{m1} & w_{m2} & \cdots & w_{mj} & \cdots & w_{mn} \end{array}\right\} \end{array}$$



Example:

- D1 = **Information** **retrieval** is concerned with the organization, storage, **retrieval** and evaluation of **information** relevant to user's **query**.
- D2 = A user having an **information** needs to formulate a request in the form of **query** written in natural languages.
- D3 = The **retrieval** system responds by retrieving the document that seems relevant to the **query**.
- T = { information, retrieval, query }
- Let the weights be assigned based on the frequency of the term in the document. Then the associated vectors will be as shown in matrix 1.
- Convert document vectors to unit length by dividing elements of each column by the length of the column vector given by $\sqrt{\Sigma_i w_{ij}^2}$

	t1	t2	t3
d1	2	2	1
d2	1	0	1
d3	0	1	1

Term Document Matrix

	d1	d2	d3
t1	2	1	0
t2	2	0	1
t3	1	1	1

After Normalization ->

$W_{ij} / \sqrt{\sum_i w_{ij}^2}$

	d1	d2	d3
t1	0.67	0.71	0
t2	0.67	0	0.71
t3	0.33	0.71	0.71