

MODULE-1

Introduction & Language Modelling

- **Introduction:** What is Natural Language Processing? Origins of NLP, Language and Knowledge, The Challenges of NLP, Language and Grammar, Processing Indian Languages, NLP Applications.
- **Language Modelling:** Statistical Language Model - N-gram model (unigram, bigram), Paninian Framework, Karaka theory.

Textbook 1: Tanveer Siddiqui, U.S. Tiwary, “Natural Language Processing and Information Retrieval”, Oxford University Press. Ch. 1, Ch. 2.

1. INTRODUCTION

1.1 What is Natural Language Processing (NLP)

Language is the primary means of communication used by humans and tool to express the greater part of our ideas and emotions. It shapes thought and has a structure, and carries meaning. To express a thought, content helps represent the language in real-time.

NLP is concerned with *development of computational models* of aspects of human language processing, there are two main reasons:

1. To develop automated tools for language processing
2. To gain a better understanding of human communication

Building computational models with human language-processing abilities requires a knowledge of how humans acquire, store, and process language.

Historically, there have been **two major approaches to NLP**:

1. Rationalist approach
2. Empiricist approach

Rationalist Approach: Early approach, assumes the existence of some language faculty in the human brain. Supporters of this approach argue that it is not possible to learn a complex thing like natural language from limited sensory inputs.

Empiricist approach: Do not believe in existence of a language faculty. Believe in the existence of some general organization principles such as pattern recognition, generalization, and association.

Learning of detailed structures takes place through the application of these principles on sensory inputs available to the child.

1.2 Origins of NLP

The NLP includes speech processing and sometimes mistakenly termed natural language understanding-originated from machine translation research. Natural language processing includes both understanding (interpretation) and generation (production). We are concerned with text processing only - The area of *computational linguistics and its application*.

Computational linguistics: is similar to theoretical- and psycho-linguistics, but uses different tools. While theoretical linguistics is more about the *structural rules of language*, psycho-linguistics focuses on *how language is used and processed* in the mind.

Theoretical linguistics explores the abstract rules and structures that govern language. It investigates universal grammar, syntax, semantics, phonology, and morphology. Linguists create models to explain how languages are structured and how meaning is encoded. Eg. Most languages have constructs like noun and verb phrases. Theoretical linguists identify rules that describe and restrict the structure of languages (grammar).

Psycho-linguistics focuses on the psychological and cognitive processes involved in language use. It examines how individuals acquire, process, and produce language. Researchers study language development in children and how the brain processes language in real-time. Eg. Studying how children acquire language, such as learning to form questions ("What's that?").

Computational Linguistics Models:

Computational linguistics is concerned with the study of language using computational models of linguistic phenomena. It deals with the *application of linguistic theories and computational techniques* for NLP. In computational linguistics, representing a language is a major problem; Most knowledge representations tackle only a small part of knowledge. Representing the whole body of knowledge is almost impossible.

Computational models may be broadly classified under *knowledge-driven* and *data-driven categories*. Knowledge-driven systems rely on explicitly coded linguistic knowledge, often expressed as a set of handcrafted grammar rules. Acquiring and encoding such knowledge is difficult and is the main bottleneck in the development of such systems.

Data-driven approaches presume the existence of a large amount of data and usually employ some machine learning technique to learn syntactic patterns. Performance of these systems is dependent on the quantity of the data and usually adaptive to noisy data.

Main objective of the models is to achieve a balance between semantic (knowledge-driven) and data-driven approaches on one hand, and between theory and practice on the other.

With the unprecedented amount of information now available on the web, NLP has become one

of the leading techniques for processing and retrieving information. NLP has become one of the leading techniques for processing and *retrieving information*.

Information retrieval includes a number of information processing applications such as information extraction, text summarization, question answering, and so forth. It includes multiple modes of information, including speech, images, and text.

1.3 Language & Knowledge

Language is the medium of expression in which knowledge is deciphered. We are here considering the *text form of the language* and the *content of it as knowledge*.

Language, being a medium of expression, is the outer form of the content it expresses. The same content can be expressed in different languages.

Hence, to process a language means 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.

The language and speech community considers a language as a set of sounds that, through combinations, conveys meaning to a listener. However, we are concerned with representing and processing text only. Language (text) processing has different levels, each involving different types of knowledge.

1.3.1 lexical analysis

- Analysis of words.
- Word-level processing requires morphological knowledge, i.e., knowledge about the structure and formation of words from basic units (morphemes).
- The rules for forming words from morphemes are language specific.

1.3.2 Syntactic analysis

- Considers a sequence of words as a unit, usually a sentence, and finds its structure.
- Decomposes a sentence 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*** (How words are combined to form larger units such as phrases and sentences)
- For example:
 - 'I went to the market' is a valid sentence whereas 'went the I market to' is not.
 - 'She is going to the market' is valid, but 'She are going to the market' is not.

1.3.3 Semantic analysis

- It is associated with the meaning of the language.
- Semantic analysis is concerned with *creating meaningful representation* of linguistic inputs.
- Eg. 'Colorless green ideas sleep furiously' - syntactically correct, but semantically anomalous.

- A word can have a number of possible meanings associated with it. But in a given context, only one of these meanings participates.



- Finding out the correct meaning of a particular use of word is necessary to find meaning of larger units.
- Eg. *Kabir and Ayan are married.*
Kabir and Suha are married.
- Syntactic structure and compositional semantics fail to explain these interpretations.
- This means that *semantic analysis requires pragmatic knowledge* besides semantic and syntactic knowledge.
- Pragmatics** helps us understand how meaning is influenced by context, social factors, and speaker intentions.

1.3.4 Discourse Analysis

- 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.



Anamorphic Reference

- Pragmatic knowledge may be needed for resolving anaphoric references.

Example: The district administration refused to give the trade union permission for the meeting because they feared violence. (a)

The district administration refused to give the trade union permission for the meeting because they oppose government. (b)

- For example, in the above sentences, resolving the anaphoric reference 'they' requires pragmatic knowledge.

1.3.5 Pragmatic analysis

- The highest level of processing, deals with the purposeful use of sentences in situations.
- It requires knowledge of the world, i.e., knowledge that extends beyond the contents of the text.

1.4 The Challenges of NLP

- Natural languages are highly ambiguous and vague, achieving precise representation of content can be difficult.
- The inability to capture all the required knowledge.
- Identifying its semantics.
- A language keeps on evolving. New words are added continually and existing words are introduced in new context. (eg. 9/11 - terrorist act on WTC)

Solution: The only way machines can learn is by considering its context, context of a word is defined by co-occurring words.

- The frequency of a word being used in a particular sense also affects its meaning.
- Idioms, metaphor, and ellipses add *more complexity* to identify the meaning of the written text.
 - Example: “*The old man finally kicked the bucket*” ⑦ “*kicked the bucket*” is a well-known **Idiom**, meaning is to “*to die*.” ○ “*Time is a thief*.” ⑦ **Metaphor** suggests “time robs you of valuable moments or experiences in life”.

- “I’m going to the store, and you’re going to the party, right?”

“Yes, I am...”

Ellipses refer to the omission of words or phrases in a sentence. (represented by "...")

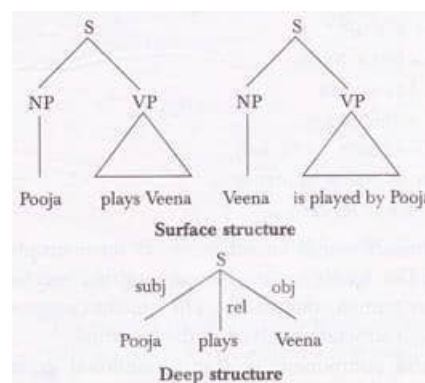
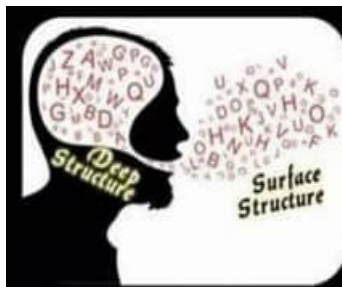
- The **ambiguity** of natural languages is another difficulty (explicit as well as implicit sources of knowledge).
 - Word Ambiguity: Example: 'Taj' - a monument, a brand of tea, or a hotel.
 - “Can” – ambiguous in its part-of-speech. ('Part-of-speech tagging' algorithm)
 - “Bank” is ambiguous in its meaning. ('word sense disambiguation' algorithm)
 - Structural ambiguity - A sentence may be ambiguous
 - 'Stolen rifle found by tree.'
 - Verb sub-categorization may help to resolve
 - Probabilistic parsing - statistical models to predict the most likely syntactic structure.
- A number of grammars have been proposed to describe the structure of sentences. ○ It is almost impossible for grammar to capture the structure of all and only meaningful text.

1.5 Language and Grammar

- **Language Grammar:** Grammar defines language and consists of rules that allow parsing and generation of sentences, serving as a foundation for natural language processing.
- **Syntax vs. Semantics:** Although syntax and semantics are closely related, a separation is made in processing due to the complexity of world knowledge influencing both language structure and meaning.

- **Challenges in Language Specification:** Natural languages constantly evolve, and the numerous exceptions make language specification challenging for computers.
- **Different Grammar Frameworks:** Various grammar frameworks have been developed, including transformational grammar, lexical functional grammar, and dependency grammar, each focusing on different aspects of language such as derivation or relationships.
- **Chomsky's Contribution:** Noam Chomsky's generative grammar framework, which uses rules to specify grammatically correct sentences, has been fundamental in the development of formal grammar hierarchies.

Chomsky argued that phrase structure grammars are insufficient for natural language and proposed *transformational grammar* in Syntactic Structures (1957). He suggested that each sentence has two levels: *a deep structure and a surface structure* (as shown in Fig 1), with transformations mapping one to the



S : sentence	Vt : transitive verb
NP : noun phrase	Vi : intransitive verb
VP : verb phrase	VL : linking verb
N : noun	Comp : complement
V : verb	Prep : preposition
D or Det : determiner	Pres : present
Pron : pronoun	PP : prepositional phrase
Prop N : proper name	M : modal
Pred : predicate	Be : the verb "be"
M : modal	Adv. P : adverb phrase
Be : the verb "be"	Adj. P : adjective phrase

other.

Fig 1. Surface and Deep Structures of sentence

- Chomsky argued that an utterance is the surface representation of a 'deeper structure' representing its meaning.
- The deep structure can be transformed in a number of ways to yield many different surface-level representations.
- Sentences with different surface-level representations having the same meaning, share a common deep-level representation.

Pooja plays veena.
Veena is played by Pooja.

Both sentences have the same meaning, despite having different surface structures (roles of subject and object are inverted).

Transformational grammar has three components:

1. Phrase structure grammar: Defines the basic *syntactic structure* of sentences.
2. Transformational rules: Describe how *deep structures can be transformed* into *different surface structures*.
3. Morphophonemic rules: Govern the relationship structure of a sentence (its syntax) influences the form of the words in terms of *sound and pronunciation* (phonology).

Phrase structure grammar consists of rules that generate natural language sentences and assign a structural description to them. As an example, consider the following set of rules:

Eg: Veena is played by Pooja.

S → NP + VP	Det → the, a, an, ...
VP → V + NP	Verb → catch, write, eat, ...
NP → Det + Noun	Noun → police, snatcher, ...
V → Aux + Verb	Aux → will, is, can, ...

S : sentence	Vt : transitive verb
NP : noun phrase	Vi : intransitive verb
VP : verb phrase	VL : linking verb
N : noun	Comp : complement
V : verb	Prep : preposition
D or Det : determiner	Pres : present
Pron : pronoun	PP : prepositional phrase
Prop N : proper name	M : modal
Pred : predicate	Be : the verb "be"
M : modal	Adv. P : adverb phrase
Be : the verb "be"	Adj. P : adjective phrase

Transformation rules, transform one phrase-marker (underlying) into another phrase-marker (derived).

These rules are applied on the terminal string generated by phrase structure rules. It transforms one surface representation into another, e.g., an active sentence into passive one.

Consider the active sentence: "The police will catch the snatcher."

Eg. [NP1 - Aux - V - NP2] → [NP2 - Aux + be + en - V - by + NP1]

The application of phrase structure rules will assign the structure shown in Fig 2 (a)

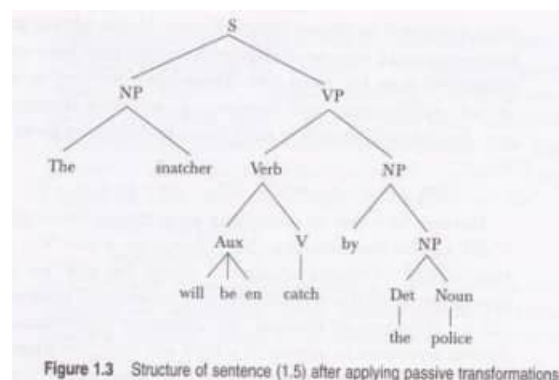
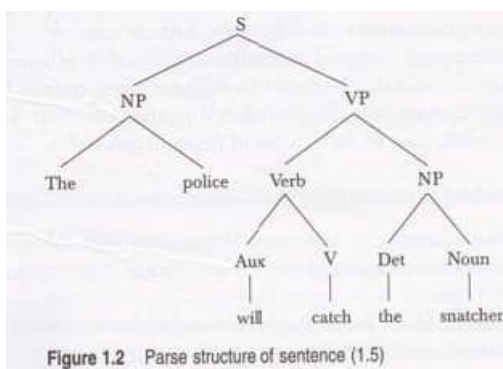


Fig. 2: (a) Phrase structure

(b) Passive Transformation

The passive transformation rules will convert the sentence into

The + culprit + will + be + en + catch + by + police

Morphophonemic Rule: Another transformational rule will then reorder 'en + catch' to 'catch + en' and subsequently one of the *morphophonemic rules* will convert 'catch + en' to 'caught'.

Note: Long distance dependency refers to syntactic phenomena where a verb and its subject or object can be arbitrarily apart. *Wh-movement* are a specific case of these types of dependencies.

E.g.

"I wonder who John gave the book to" involves a long-distance dependency between the verb *"wonder"* and the object *"who"*. Even though *"who"* is not directly adjacent to the verb, the syntactic relationship between them is still clear.

The *problem in the specification of appropriate phrase structure rules* occurs because these phenomena cannot be localized at the surface structure level.

1.6 Processing Indian Languages

There are a number of differences between Indian languages and English:

- Unlike English, Indic scripts have a non-linear structure.
- Unlike English, Indian languages have SOV (Subject-Object-Verb) as the default sentence structure.
- Indian languages have a free word order, i.e., words can be moved freely within a sentence without changing the meaning of the sentence.
- Spelling standardization is more subtle in Hindi than in English.
- Indian languages have a relatively rich set of morphological variants.
- Indian languages make extensive and productive use of complex predicates (CPs).
- Indian languages use post-position (Karakas) case markers instead of prepositions.
- Indian languages use verb complexes consisting of sequences of verbs,
 - e.g., गा रहा है (ga raha hai-singing) and खेल रही है (khel rahi hai-playing).
 - The auxiliary verbs in this sequence provide information about tense, aspect, modality, etc

Paninian grammar provides a framework for Indian language models. These can be used for computation of Indian languages. The grammar focuses on extraction of relations from a sentence.

1.7 NLP Applications

1.7.1 Machine Translation

This refers to automatic translation of text from one human language to another. In order to carry out this translation, it is necessary to have an understanding of words and phrases, grammars of the two languages involved, semantics of the languages, and word knowledge.

1.7.2 Speech Recognition

This is the process of mapping acoustic speech signals to a set of words. The difficulties arise due to wide variations in the pronunciation of words, homonym (e.g. dear and deer) and acoustic ambiguities (e.g., in the rest and interest).

1.7.3 Speech Synthesis

Speech synthesis refers to automatic production of speech (utterance of natural language sentences). Such systems can read out your mails on telephone, or even read out a storybook for you.

1.7.4 Information Retrieval

This focuses on identifying relevant documents for a user's query using NLP techniques in information retrieval. Methods like indexing, word sense disambiguation, query modification, and knowledge bases improve IR performance, including query expansion. Lexical resources like WordNet, LDOCE, and Roget's Thesaurus enhance these systems. These tools help refine search results and improve accuracy.

1.7.5 Information Extraction

An information extraction system captures and outputs factual information contained within a document. Query is specified as pre-defined templates. System identifies a subset of information within a document that fits the pre-defined template.

1.7.6 Question Answering

Given a question and a set of documents, a question answering system attempts to find the precise answer, or at least the precise portion of text in which the answer appears. A question answering system requires more NLP than an information retrieval system or an information extraction system. It requires not only precise analysis of questions and portions of texts but also semantic as well as background knowledge to answer certain type of questions.

1.7.7 Text Summarization

This deals with the creation of summaries of documents and involves syntactic, semantic, and discourse level processing of text.

1.8 Some Successful Early NLP Systems

1.8.1 ELIZA (Weizenbaum 1966)

ELIZA is one of the earliest natural language understanding programs. It uses syntactic patterns to mimic human conversation with the user. Here is a sample conversation.

Eliza: Hello. I am ELIZA. How may I help you?

User: I am feeling a little bit sleepy.

Eliza: How long have you been feeling a little bit sleepy?

User: For almost half an hour.

Eliza: Please go on.

1.8.2 SysTran (System Translation)

The first SysTran machine translation system was developed in 1969 for Russian-English translation. SysTran also provided the first on-line machine translation service called Babel Fish, which is used by AltaVista search engines for handling translation requests from users.

1.8.3 TAUM METEO

This is a natural language generation system used in Canada to generate weather reports. It accepts daily weather data and generates weather reports in English and French.

1.8.4 SHRDLU (Winograd 1972)

This is a natural language understanding system that simulates actions of a robot in a block world domain. It uses syntactic parsing and semantic reasoning to understand instructions. The user can ask the robot to manipulate the blocks, to tell the blocks configurations, and to explain its reasoning.

1.8.5 LUNAR (Woods 1977)

This was an early question answering system that answered questions about moon rock.

1.9 Information Retrieval

The availability of vast amounts of electronic text has made it challenging to find relevant information. Information retrieval (IR) systems aim to address this issue by providing *efficient access to relevant content*. Unlike 'entropy' in communication theory, which measures uncertainty, information here refers to the *content or subject matter of text*, not digital communication or data transmission. Words serve as carriers of information, and text is seen as the message encoded in natural language.

In IR, "retrieval" refers to accessing information from computer-based representations, requiring processing and storage. Only relevant information, based on a *user's query*, is retrieved. IR involves *organizing, storing, retrieving, and evaluating information that matches a query*, working with

unstructured data. Retrieval is based on content, not structure, and systems typically return a ranked list of relevant documents.

IR has been integrated into various systems, including database management systems, bibliographic retrieval systems, question answering systems, and search engines. Approaches for accessing large text collections fall into two categories: one builds topic hierarchies (e.g., Yahoo), requiring manual classification of new documents, which can be cost-ineffective; the other ranks documents by relevance, offering more scalability and efficiency for large collections

Major issues in designing and evaluating Information Retrieval (IR) systems include selecting appropriate document representations. Current models often use keyword-based representation, which suffers from problems like polysemy, homonymy, and synonymy, as well as ignoring semantic and contextual information. Additionally, vague or inaccurate user queries lead to poor retrieval performance, which can be addressed through query modification or relevance feedback.

Matching *query representation* to document representation is another challenge, requiring effective similarity measures to rank results. Evaluating IR system performance typically relies on recall and precision, though relevance itself is subjective and difficult to measure accurately. Relevance frameworks, such as the situational framework, attempt to address this by considering context and time. Moreover, varying user needs and document collection sizes further complicate retrieval, requiring specialized methods for different scopes.

2. LANGUAGE MODELLING

To create a general model of any language is a difficult task. There are two approaches for language modelling.

1. To *define a grammar* that can handle the language.
2. To *capture the patterns in a grammar* language statistically.

2.1 Introduction

Our purpose is to understand and generate natural languages from a computational viewpoint.

1st approach: Try to *understand every word and sentence* of it, and then come to a conclusion (has not succeeded).

2nd approach: To study the grammar of various languages, compare them, and if possible, arrive at reasonable models that facilitate our understanding of the problem and designing of natural-language tools.

Language Model: A model is a description of some complex entity or process. Natural language is a complex entity and in order to process it through a computer-based program, we need to build a representation (model) of it.

Two categories of language modelling approaches:

Grammar-based language model:

- Uses the grammar of a language to create its model.
- It attempts to represent the syntactic structure of language.
- Hand-coded rules defining the *structure and ordering* of various constituents appearing in a linguistic unit.

Eg. A sentence usually consists of noun phrase and a verb phrase. The grammar-based approach attempts to utilize this structure and also the relationships between these structures.

Statistical language modelling:

- Creates a language model by training it from a corpus.
- To capture regularities of a language, the training corpus needs to be sufficiently large.
- Fundamental tasks in many NLP applications, including speech recognition, spelling correction, handwriting recognition, and machine translation.
- Information retrieval, text summarization, and question answering.
- Most popular - n-gram models.

2.2 Various Grammar-Based Language Models

- Generative Grammars
- Hierarchical Grammar
- Government and Binding (GB)
- Lexical Functional Grammar (LFG) Model
- Paninian Framework

2.2.1 Generative Grammars

- We can generate sentences in a language if we know a collection of words and rules in that language (Noam Chomsky).
- Sentences that can be generated as per the rules are grammatical and had dominated computational linguistics.
- Addressed syntactical structure of language.
- But Language is a relation between the sound (or the written text) and its meaning.

2.2.2 Hierarchical Grammar

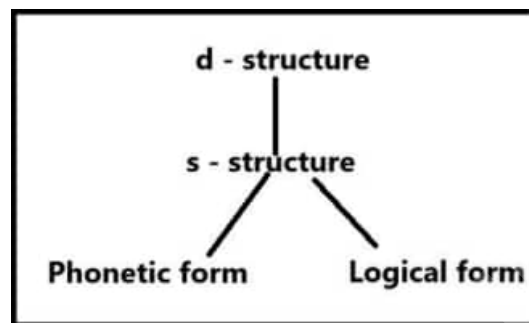
- Chomsky (1956) described classes of grammars in a hierarchical manner, where the top layer contained the grammars represented by its sub classes.
- Hence, Type 0 (or unrestricted) grammar contains Type 1 (or context-sensitive grammar), which in turn contains Type 2 (context-free grammar) and that again contains Type 3 grammar (regular grammar).

2.2.3 Government and Binding (GB)

(Eliminated rules of Grammar – since rules were language particular)

Linguists often argue that language structure, especially in resolving structural ambiguity, can be understood through meaning. However, the transformation between *meaning and syntax is not well understood*. Transformational grammars distinguish between surface-level and deep-root-level sentence structures.

Government and Binding (GB) theories rename these as s-level and d-level, *adding phonetic and logical forms* as parallel levels of representation for analysis, as shown in Figure.



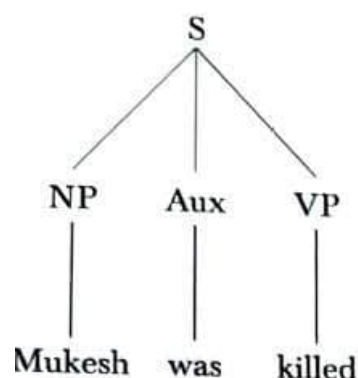
- 'meaning' in a 'sound' form is represented as **logical form** (LF) and **phonetic form** (PF) in above figure.
- The **GB is concerned with LF**, rather than PF.
- The GB imagines *that if we define rules for structural units at the deep level*, it will be possible to generate any language with fewer rules.

Let us take an example to explain d- and s- Structures in GB:

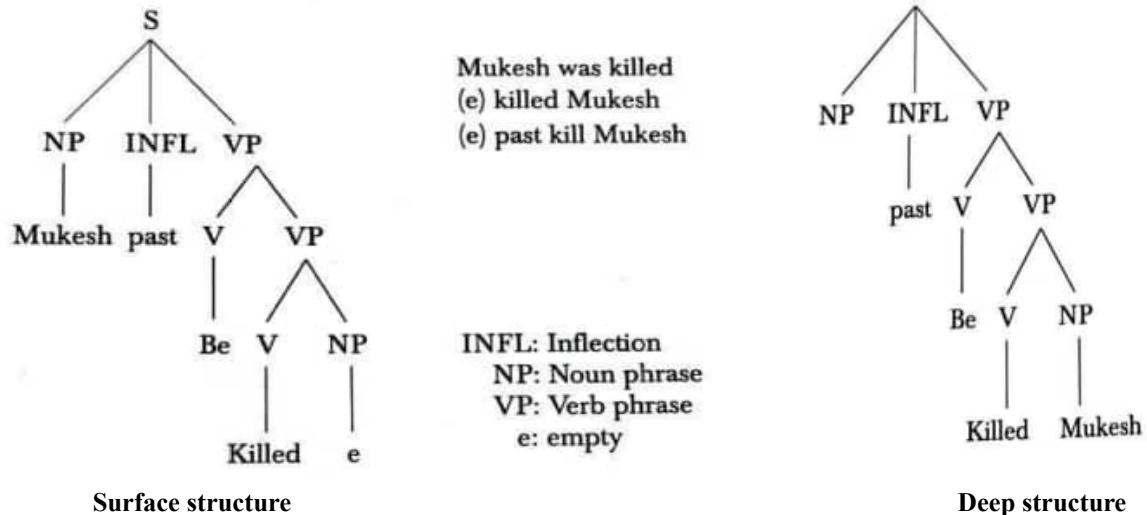
Mukesh was killed

- i) In Transformational grammar, this can be expressed as:

S – NP AUX VP ⑦ as given below



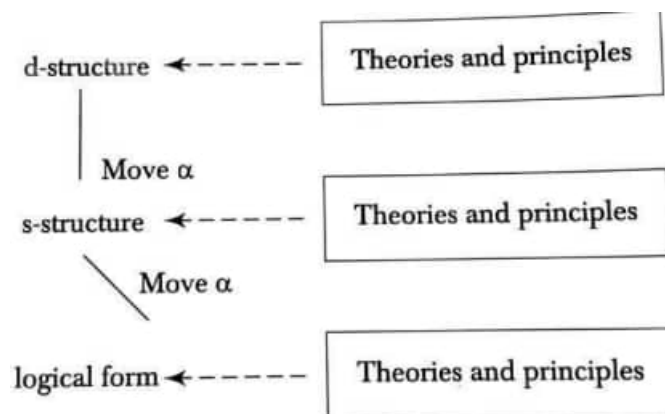
- ii) In GB, s-structure & d-structure are as follows:

**Note:**

- The surface structure is the actual form of the sentence as it appears in speech or writing.
- The deep structure represents the underlying syntactic and semantic structure that is abstract and not directly visible (Represents the core meaning of the sentence). "Someone killed Mukesh" or "A person killed Mukesh."

Components of GB

- Government and binding (GB) comprise a set of theories that map the structures from d-structure to s-structure and to logical form (LF).
- A general transformational rule called 'Move α ' is applied at d-structure level as well as at s-structure level.
- Simplest from GB can be represented as below.



GB consists of 'a series of modules that contain constraints and principles' applied at various levels of its representations and the transformation rule, *Move α* .

The GB considers all three levels of representations (*d*-, *s*-, and *LF*) as syntactic, and *LF* is also related to meaning or semantic-interpretive mechanisms.

GB applies the same Move α transformation to map d-levels to s-levels or s-levels to LF level. LF level helps in quantifier scoping and also in handling various sentence constructions such as *passive* or *interrogative constructions*.

Example:

Consider the sentence: “ *Two countries are visited by most travellers.* ”

Its two possible logical forms are:

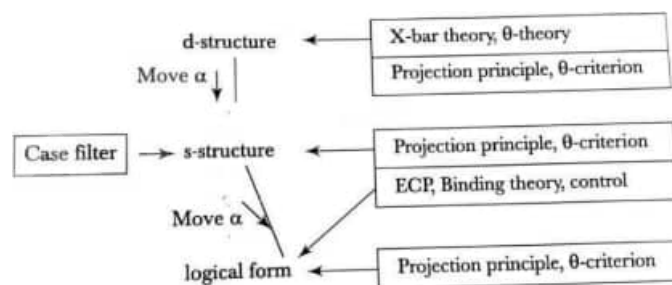
LF1: [_s Two countries are visited by [_{NP} most travellers]]

LF2: Applying Move α

[_{NP} Most travellers_i] [_s two countries are visited by e_i]

- In LF1, the interpretation is that most travellers visit the same two countries (say, India and China).
- In LF2, when we move [most travellers] outside the scope of the sentence, the interpretation can be that most travellers visit two countries, which may be different for different travellers.
- One of the important concepts in GB is that of constraints. It is the part of the grammar which prohibits certain combinations and movements; otherwise Move α can move anything to any possible position.
- Thus, GB, is basically the *formulation of theories or principles* which create constraints to disallow the construction of ill-formed sentences.

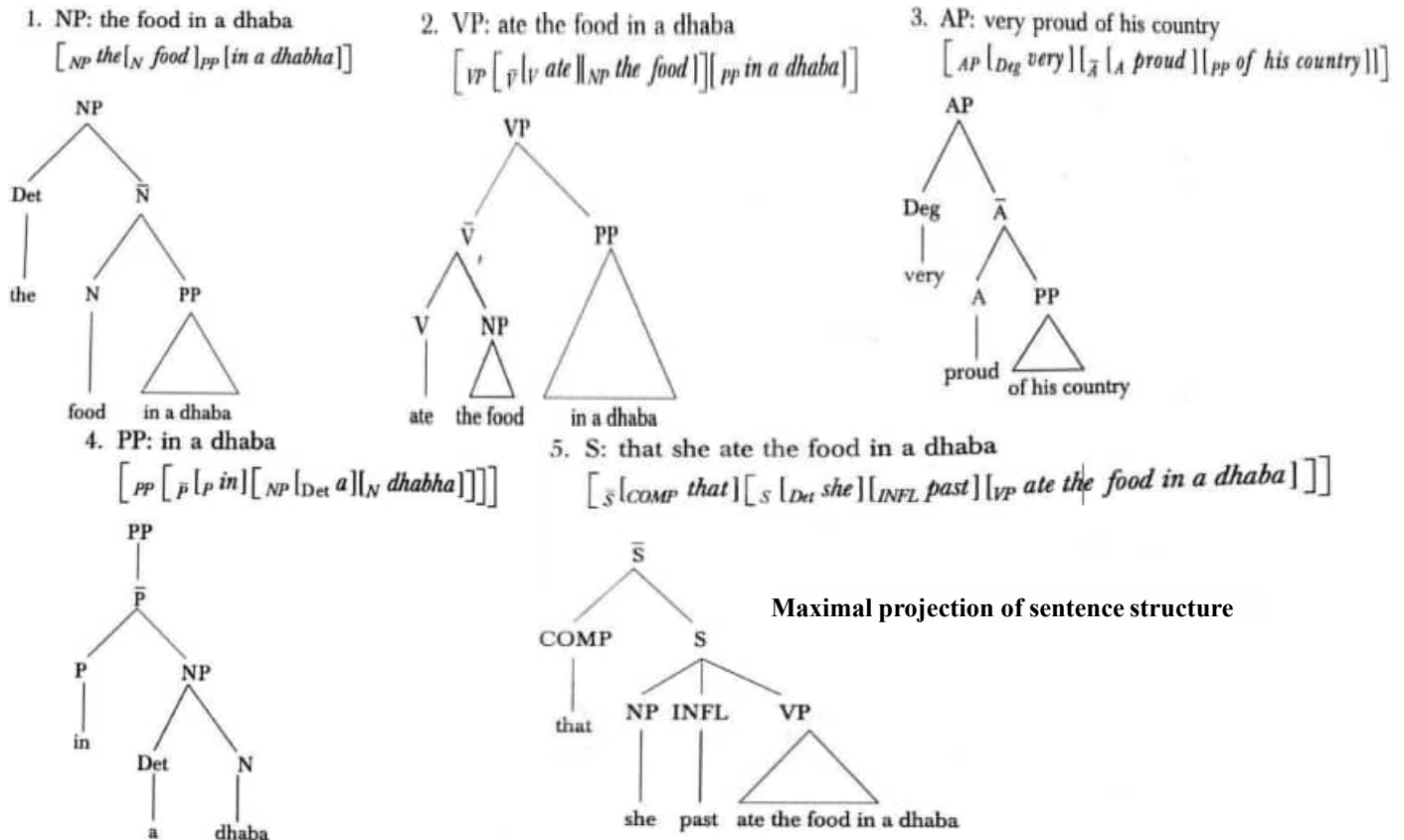
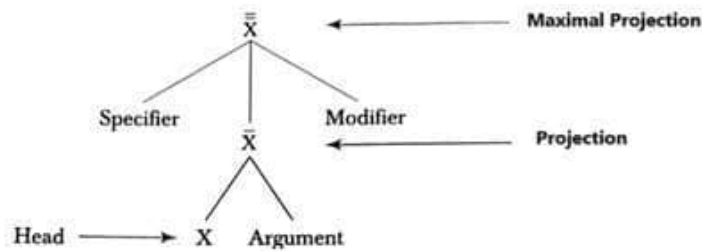
The organization of GB is as given below:



X Theory:

- The **X** Theory (pronounced 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, **X** Theory defines them both as maximal projections of some head.
- Noun phrase (NP), verb phrase (VP), adjective phrase (AP), and prepositional phrase (PP) are maximal projections of noun (N), verb (V), adjective (A), and preposition (P) respectively, and can be represented as head X of their corresponding phrases (where $X = \{N, V, A, P\}$)
- Even the sentence structure can be regarded as the maximal projection of inflection (INFL).
- The GB envisages projections at two levels:
 - The projection of head at semi-phrasal level, denoted by **X**,
 - The Maximal projection at the phrasal level, denoted by **X**.

Figure depicts the general and particular structures with examples



Sub-categorization: It refers to the process of *classifying words or phrases (typically verbs) according to the types of arguments or complements they can take*. It's a form of syntactic categorization that is important for understanding the structure and meaning of sentences.

For example, different verbs in English can have different sub-categorization frames (also called argument structures). A verb like "give" might take three arguments (subject, object, and indirect object), while a verb like "arrive" might only take a subject and no objects.

"He **gave** her a book." ("gave" requires a subject, an indirect object, and a direct object)

"He **arrived**." ("arrived" only requires a subject)

In principle, any **maximal projection can be the argument of a head**, but sub-categorization is used as a filter to permit various heads to select a certain subset of the range of maximal projections.

Projection Principle:

Three syntactic representations:

1. Constituency Parsing (Tree Structure):
 - Sentences are broken into hierarchical phrases or constituents (e.g., noun phrases, verb phrases), represented as a tree structure.
2. Dependency Parsing (Directed Graph):
 - Focuses on the direct relationships between words, where words are connected by directed edges indicating syntactic dependencies.
3. Semantic Role Labelling (SRL):
 - Identifies the semantic roles (e.g., agent, patient) of words in a sentence, focusing on the meaning behind the syntactic structure.

The projection principle, a basic notion in GB, **places a constraint on the three syntactic representations and their mapping from one to the other.**

The principle states that **representations at all syntactic levels** (i.e., d-level, s-level, and LF level) **are projections from the lexicon** (collection or database of words and their associated linguistic information).

Thus, lexical properties of categorial structure (sub-categorization) must be observed at each level.

Suppose 'the object' is not present at d-level, then another NP cannot take this position at s-level.

Example:

- At D-structure, each argument of a verb is assigned a **thematic role** (e.g., **Agent**, **Theme**, **Goal**, etc.).
- In a sentence like "**John gave Mary the book**", the verb "gave" requires three arguments: **Agent** (John), **Recipient** (Mary), and **Theme** (the book).
- If the **object** (Theme) is not present at the deep structure, it cannot be filled at the surface structure (S-structure) by another NP (e.g., a different noun phrase).

Theta Theory (θ -Theory) or The Theory of Thematic Relations

- 'Sub-categorization' only places a restriction on syntactic categories which a head can accept.
- GB puts another restriction on the lexical heads through which it assigns certain roles to its arguments.
- These roles are pre-assigned and cannot be violated at any syntactical level as per the projection principle.
- These role assignments are called theta-roles and are related to 'semantic-selection'.

Theta Role and Theta Criterion

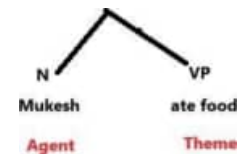
There are certain thematic roles from which a head can select. These are called **θ -roles** and they are mentioned in the lexicon, say for example the verb 'eat' can take arguments with **θ -roles** '(Agent, Theme)'.

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).

Hence in 'Mukesh ate food',

the verb 'eat' assigns the 'Agent' role to 'Mukesh' (outside VP) and

'Theme' (or 'patient') role to 'food'.



Theta-Criterion states that 'each argument bears one and only one ***θ*-role**, and each ***θ*-role** is assigned to one and only one argument'.

C-command and Government

C-Command: It is a syntactic relation that *defines a type of hierarchical relationship between two constituents* (words or phrases) in a sentence. It plays a critical role in the distribution of certain syntactic phenomena, such as **binding**, **agreement**, and **pronoun reference**.

If any word or phrase (say α or β) falls within the scope of and is determined by a maximal projection, we say that it is dominated by the maximal projection.

If there are two structures α and β related in such a way that 'every maximal projection dominating α dominates β ', we say that α C-commands β , and this is the necessary and sufficient condition (iff) for C-command.

Government α governs β iff: α

C-commands β

α is an X (head, e.g., noun, verb, preposition, adjective, and inflection), and every maximal projection dominating β dominates α .

Additional information

C-COMMAND

A c-command is a syntactic relationship in linguistics, particularly in the theory of syntax, where one node (word or phrase) in a tree structure can "command" or "govern" another node in certain ways. In simpler terms, it's a rule that helps determine which parts of a sentence can or cannot affect each other syntactically.

Simple Definition:

C-command occurs when one word or phrase in a sentence has a syntactic connection to another word or phrase, typically by being higher in the syntactic tree (closer to the top).

Example 1:

In the sentence "John saw Mary,"

"John" c-commands "Mary" because "John" is higher up in the tree structure and can potentially affect "Mary" syntactically.

Example 2:

In the sentence "She thinks that I am smart,"

The pronoun "She" c-commands "I" because "She" is higher in the syntactic tree, governing the phrase where "I" occurs.

In essence, c-command helps explain which words in a sentence are connected in ways that allow for things like pronoun interpretation or binding relations (e.g., which noun a pronoun refers to).

GOVERNMENT

-is a special case of C-COMMAND

government refers to the syntactic relationship between a head (typically a verb, noun, or adjective) and its dependent elements (such as objects or complements) within a sentence. It determines how certain words control the form or case of other words in a sentence.

On the other hand, c-command is a syntactic relationship between two constituents in a sentence. A constituent A c-commands another constituent B if the first constituent (A) is higher in the syntactic structure (usually in the tree) and can potentially govern or affect the second constituent (B), provided no intervening nodes.

To put it together in context:

Government: This is a formal rule determining how certain words govern the case or form of other words in a sentence (e.g., verbs can govern the object noun in accusative case in languages like Latin or German).

C-command: This is a structural relationship in which one constituent can influence another, typically affecting operations like binding, scope, and sometimes government.

In short, government often operates within the structures of c-command, but c-command itself is a broader syntactic relationship that is also relevant for other linguistic phenomena, such as binding theory, where one element can bind another if it c-commands it.

Sure! Here are a few examples of government in syntax, showing how one word governs the form or case of another word in a sentence:

1. Verb Government

In many languages, verbs can govern the case of their objects. Here's an example in Latin:

Latin: "Vidēre puellam" (to see the girl)

The verb "vidēre" (to see) governs the accusative case of "puellam" (the girl).

In this case, the verb "vidēre" governs the object "puellam" by requiring it to be in the accusative case.

2. Preposition Government

Prepositions can also govern the case of their objects. Here's an example from German:

German: "Ich gehe in den Park" (I am going to the park)

The preposition "in" governs the accusative case of "den Park" (the park).

The preposition "in" governs the accusative case for the noun "Park" in this sentence.

3. Adjective Government

Adjectives can govern the case, gender, or number of the noun they modify. Here's an example from Russian:

Russian: "Я вижу красивую девочку" (I see a beautiful girl)

The adjective "красивую" (beautiful) governs the accusative case of "девочку" (girl).

In this case, the adjective "красивую" (beautiful) governs the accusative case of "девочку".

4. Noun Government

In some languages, nouns can govern the case of their arguments. In Russian, for example, some nouns govern a particular case:

Russian: "Я горжусь успехом" (I am proud of the success)

The noun "успехом" (success) governs the instrumental case in this sentence.

Here, the noun "успехом" governs the instrumental case of its argument "успех".

Summary:

Government involves syntactic relationships where a head (verb, preposition, adjective, etc.) dictates or determines the form (such as case) of its dependent elements.

In these examples, verbs, prepositions, and adjectives have a "governing" influence on the cases of nouns or objects in the sentence, which is a core part of the syntax in many languages.

Movement, Empty Category, and Co-indexing

Movement & Empty Category:

In GB, *Move α* is described as 'move anything anywhere', though it provides restrictions for valid movements.

In GB, the active to passive transformation is the result of NP movement as shown in sentence. Another well-known movement is the *wh-movement*, where *wh-phrase* is moved as follows.

What did Mukesh eat?
[Mukesh INFL eat what]

As discussed in the projection principle, *lexical categories must exist at all the three levels*. This principle, when applied to some cases of *movement leads to the existence of* an abstract entity called *empty category*.

In GB, there are four types of empty categories:

Two being empty NP positions called *wh-trace* and *NP trace*, and the remaining two being pronouns called *small 'pro'* and *big 'PRO'*.

This division is based on two properties—anaphoric (+a or -a) and pronominal (+p or -p).

Wh-trace	-a, -p
NP-trace	+a, -p
small 'pro'	-a, +p
big 'PRO'	+a, +p

The traces help ensure that the proper binding relationships are maintained between moved elements (such as how pronouns or reflexives bind to their antecedents, even after movement).

Additional Information:

- **+a (Anaphor):** A form that must refer back to something mentioned earlier (i.e., it has an antecedent). For example, "himself" in "John washed himself." The form "himself" is an anaphor because it refers back to "John."
- **-a (Non-Anaphor):** A form that does not require an antecedent to complete its meaning. A regular pronoun like "he" in "He went to the store" is not an anaphor because it doesn't explicitly need to refer back to something within the same sentence or clause.
- **+p (Pronominal):** A form that can function as a pronoun, standing in for a noun or noun phrase. For example, "she" in "She is my friend" is a pronominal because it refers to a specific person (though not necessarily previously mentioned).
- **-p (Non-Pronominal):** A word or form that isn't used as a pronoun. It could be a noun or other word that doesn't serve as a replacement for a noun phrase in a given context. For example, in "John went to the store," "John" is not pronominal—it is a noun phrase.

Co-indexing

It is the indexing of the subject NP and AGR (agreement) at d-structure which are preserved by *Move α* operations at s-structure.

When an NP-movement takes place, a *trace of the movement is created* by having an *indexed empty category* (e) from the position at which the movement began to the corresponding indexed NP.

For defining constraints to movement, the theory identifies two positions in a sentence. Positions assigned θ -roles are called θ -positions, while others are called $\bar{\theta}$ positions.

In a similar way, core grammatical positions (where subject, object, indirect object, etc., are positioned) are called *A-positions* (arguments positions), and the rest are called *A'-positions*.

Binding theory:

Binding Theory is a syntactic theory that explains how pronouns and noun phrases are interpreted and distributed in a sentence. It's concerned with the *relationships* between *pronouns and their antecedents* (myself, herself, himself).

Binding is defined by Sells (1985) as follows:

α binds β iff α C-

commands β , and α

and β are co-indexed

As we noticed in sentence,

[e_i INFL kill Mukesh]
[Mukesh; was killed (by e_i)]
Mukesh was killed.

Empty clause (e_i) and Mukesh (NP_i) are bound. This theory gives a relationship between NPs.

Empty clause (e_i) and Mukesh (NP_i) are bound. This theory gives a relationship between NPs (including pronouns and reflexive pronouns). Now, binding theory can be given as follows:

(a) An anaphor (+a) is bound in its governing category.

(b) A pronominal (+p) is free in its governing category.

(c) An R-expression (-a, -p) is free.

Example

A: Mukesh_i knows himself_i

B: Mukesh_i believes that Amrita knows him_i

C: Mukesh believes that Amrita_j knows Nupur_k (Referring expression)

Similar *rules apply on empty categories also*:

NP-trace: +a, -p: Mukesh, was killed e_i

wh-trace: -a, -p: Who does he_i like e_i

Empty Category Principle (ECP):

The 'proper government' is defined as:

α properly governs β iff:

α governs β and α is lexical (i.e. N, V, A, or P) or α

locally A-binds β

The ECP says 'A trace must be properly governed'.

This principle justifies the creation of empty categories during *NP-trace* and *wh-trace* and also explains the subject/object asymmetries to some extent. As in the following sentences:

(a) What_i do you think that Mukesh ate e_i?

(b) What_i do you think Mukesh ate e_i?

Mukesh is subject, ate is a verb and *what is object that moves to the front*. Mukesh remains in its original position.

Bounding and Control Theory:

Note: There are many other types of constraints on Move α and not possible to explain all of them.

In English, the long-distance movement for complement clause can be explained by bounding theory if NP and S are taken to be bounding nodes. The theory says that the application of Move α may not cross more than one bounding node. The theory of control involves syntax, semantics, and pragmatics.

Case Theory and Case Filter:

In GB, case theory deals with the distribution of NPs and mentions that *each NP must be assigned a case*. In English, we have the nominative, objective, genitive, etc., cases, which are assigned to NPs at particular positions. Indian languages are rich in case-markers, which are carried even during movements.

Example:

He is running ("He" is the subject of the sentence, performing the action. - nominative)

She sees **him**. ("Him" is the object of the verb "sees." - Objective)

The **man's** book. (The **genitive** case expresses **possession** or a relationship between nouns,)

Case filter: An NP is ungrammatical if it has phonetic content or if it is an argument and is not case-marked. Phonetic content here, refers to some physical realization, as opposed to empty categories.

Thus, *case filters restrict the movement of NP at a position which has no case assignment*. It works in a manner similar to that of the θ -criterion.

Summary of GB:

In short, GB presents a model of the language which has *three levels of syntactic representation*.

- It assumes phrase structures to be the maximal projection of some lexical head and in a similar fashion, explains the structure of a sentence or a clause.

- It assigns various types of roles to these structures and allows them a broad kind of movement called Move α .
- It then defines various types of constraints which restrict certain movements and justifies others.

2.2.4 Lexical Functional Grammar (LFG) Model

%Watch this video: https://www.youtube.com/watch?v=EoCLhS_0cmE %

- LFG represents sentences at two syntactic levels - *constituent structure (c-structure)* and *functional structure (f-structure)*.
- Kaplan proposed a concrete form for *the register names and values* which became the functional structures in LFG.
- Bresnan was more concerned with the problem of *explaining some linguistic issues, such as active/passive and dative alternations*, in *transformational approach*. She proposed that such issues can be dealt with by using lexical redundancy rules.
- The unification of these two diverse approaches (with a common concern) led to the development of the LFG theory.

The term 'lexical functional' is composed of two terms:

- The 'functional' part is derived from 'grammatical functions', such as subject and object, or roles played by various arguments in a sentence.
- The 'lexical' part is derived from the fact that the lexical rules can be formulated to help define the given structure of a sentence and some of the long-distance dependencies, which is difficult in transformational grammars.

C-structure and f-structure in LFG

The c-structure is derived from the usual phrase and sentence structure syntax, as in CFG

The grammatical-functional role cannot be derived directly from phrase and sentence structure, functional specifications are annotated on the nodes of c-structure, which when applied on sentences, results in f-structure



Example: She saw stars in the sky

```
[
  SUBJ: [ PERS: 3, NUM: SG ], // "She" is the subject, 3rd person, singular
  PRED: "see", // The verb "saw" represents the predicate "see"
  OBJ: [ NUM: PL, PRED: "star" ], // "stars" is the object, plural, and the predicate is "star"
  LOC: [ PRED: "sky", DEF: + ] // "sky" is the location, with a definite determiner ("the")
]
```

*f-structure**c- structure*

Example:

She saw stars in the sky

CFG rules to handle this sentence are:

S $\rightarrow$ NP VP

VP $\rightarrow$ V {NP} PP* {S'}

Stars Sky

PP $\rightarrow$ P NP

NP $\rightarrow$ Det N {PP}

S' $\rightarrow$ Comp S

Where: S: Sentence

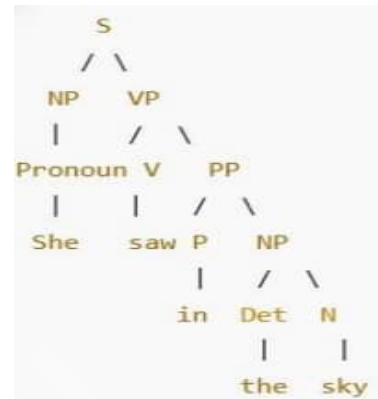
V: Verb P: Preposition

N: Noun

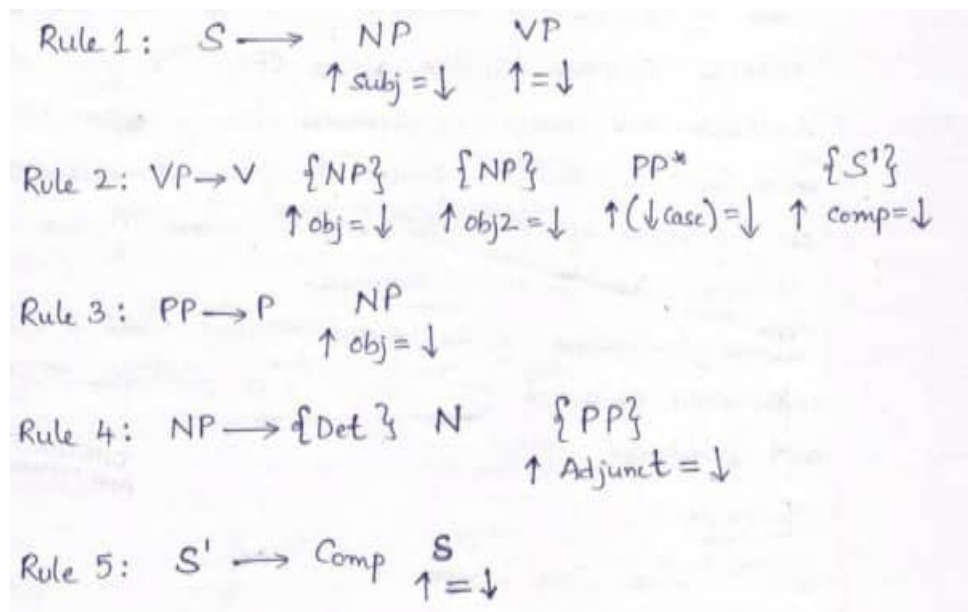
S': clause

Comp: complement { }: optional

* : Phrase can appear any number of times including blank



When annotated with functional specifications, the rules become



- Here, $\uparrow$ (up arrow) refers to the f-structure of the mother node that is on the left-hand side of the rule.
- The $\downarrow$ (down arrow) symbol refers to the f-structure of the node under which it is denoted.

- Hence, in Rule 1, $\uparrow \text{subj} = \downarrow$ indicates that the f-structure of the first NP goes to the f-structure of the subject of the sentence, while $\uparrow = \downarrow$ indicates that the f-structure of the VP node goes directly to the f-structure of the sentence VP.

Consistency In a given f-structure, a particular attribute may have at the most one value. Hence, while unifying two f-structures, if the attribute Num has value SG in one and PL in the other, it will be rejected.

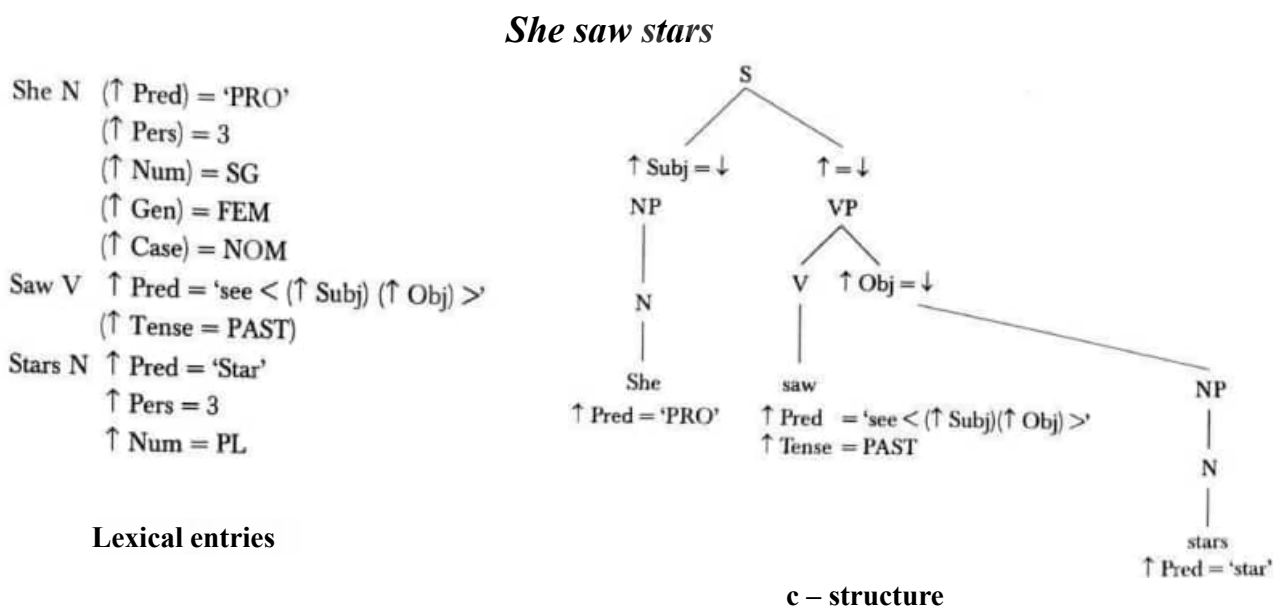
Completeness When an f-structure and all its subsidiary f-structures (as the value of any attribute of f-structure can again contain other f-structures) contain all the functions that their predicates govern, then and only then is the f-structure complete.

For example, since the predicate 'see < ($\uparrow$ Subj) ($\uparrow$ Obj) >' contains an object as its governable function, a sentence like 'She saw' will be incomplete.

Coherence Coherence maps the completeness property in the reverse direction. It requires that all governable functions of an f-structure, and all its subsidiary f-structures, must be governed by their respective predicates. Hence, in the f-structure of a sentence, an object cannot be taken if its verb does not allow that object. Thus, it will reject the sentence, 'I laughed a book.'

Example:

Let us see first the lexical entries of various words in the sentence:



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

It is interesting to note that the final f-structure is obtained through the unification of various f-structures for subject, object, verb, complement, etc. This unification is based on the functional specifications of the verb, which predicts the overall sentence structure.

$$\left[\begin{array}{l} \text{subj} \\ \text{obj} \\ \text{Pred} \end{array} \left[\begin{array}{l} \left[\begin{array}{ll} \text{Pers} & 3 \\ \text{Num} & \text{SG} \\ \text{Gen} & \text{FEM} \\ \text{Case} & \text{NOM} \end{array} \right] \\ \left[\begin{array}{ll} \text{Pers} & 3 \\ \text{Num} & \text{PL} \end{array} \right] \\ \text{'see' < } (\uparrow \text{subj}) (\uparrow \text{obj}) > \end{array} \right]$$

Lexical Rules in LFG

Different theories have different kinds of lexical rules and constraints for handling various sentence-constructs (active, passive, dative, causative, etc.).

In LFG, the *verb is converted to the participial form*, but the sub-categorization is changed directly.

Consider the following example: *oblique*

agent (Obl_{ag}) phrase:

Active: Tara ate the food.

Passive: The food was eaten by Tara

Active: ↑ Pred='eat'(<↑ Subj> (↑ Obj)>,'

Passive: ↑ Pred='eat'(<↑ Obl_{ag}> (↑ Subj)>,'

Here, Obl_{ag} represents oblique agent phrase.

Similar rules can be applied in active and dative constructs for the verbs that accept two objects.

oblique goal (Obl_{go}) phrase:

Active: Tara gave a pen to Monika.

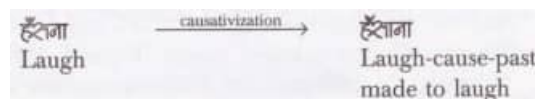
Passive: Tara gave Monika a pen.

Active: ↑ Pred='give'(<↑ Subj> (↑ Obj₂) (↑ Obj)>,'

Passive: ↑ Pred='give'(<↑ Subj> (↑ Obj) (↑ Obl_{go})>,'

Here, Obl_{go} stands for oblique goal phrase.

Similar rules are also applicable to the process of causativization. This can be seen in Hindi, where the *verb form is changed* as follows:



Example

Active: तारा हँसी

Taaraa hansii

Tara laughed

Causative: मोनिका ने तारा को हँसाया

Monika ne Tara ko hansaayaa

Monika Subj Tara Obj laugh-cause-past

Monika made Tara to laugh.

Active: ↑ Pred='Laugh'(<↑ Subj>,'

Causative: ↑ Pred='cause'(<↑ Subj> (↑ Obj) (Comp)>,'

Long Distance Dependencies and Coordination

In GB, when a category moved, it creates an empty category.

Here, a new predicate is formed which causes the action and requires a new subject, while the old subject becomes the object of the new predicate and the old verb becomes the X-complement (complement to infinital VPs).

In LFG, unbounded movement and coordination is handled by the functional identity and by correlation with the corresponding f-structure.

Example: Consider the wh-movement in the following sentence.

Which picture does Tara like-most?

The f-structure can be represented as follows:

Focus	[	Obl _{th}	[	pred	'PRO'	]	]
				Refl	+		
Subj		Pred		'picture <(<Obl _{th} >)'			
Obj		[	pred	'Tara'	]		
Pred				'like <(<↑Subj> (<↑Obj>))'			

2.2.5 Paninian Framework

Paninian grammar (PG) was written by Panini in 500 BC in Sanskrit (the original text being titled *Asthadhyayi*), the framework can be used for other Indian languages and possibly some Asian languages as well.

Unlike *English* (*Subject-Verb-Object* ordered), **Asian languages are SOV (Subject-Object-Verb) ordered** and **inflectionally rich**. The inflections provide important syntactic and semantic cues for language analysis and understanding. The Paninian framework takes advantage of these features.

Note: Inflectional – refers to the changes a word undergoes to express different grammatical categories such as tense, number, gender, case, mood, and aspect without altering the core meaning of the word.

Indian languages have traditionally used oral communication for knowledge propagation. In Hindi, we can change the position of subject and object. For example:

(a) माँ बच्च को खाने द ती है।

Maan Bachche ko khanaa detii hai

Mother child to food give-(s)

Mother gives food to the child.

(b) बच्च को माँ खाने द ती है।

Bachche ko Maan khanaa detii hai

Child to mother food give-(s)

Mother gives food to the child.

The auxiliary verbs follow the main verb. In Hindi, they remain as separate words:

खा रहा है

khaa raha hai

eat-ing

eating

करता रहा है

kartaa rahaa hai

doing been has

has been doing

In Hindi, some verbs (main), e.g., give (द नेा), take (ल नेा), also combine with other verbs (main) to change the aspect and modality of the verbs.

उनस खानेा खाया।
 Usne khanaa khaayaa
 He (Subj) food ate
 He ate food

वह चला
 He moved

उनस खानेा खा िलया।
 Usne khaanaa kha liyaa
 He (Subj) food eat taken
 He ate food (completed the action)

वह चल िदया
 He move given
 He moved (started the action)

The **nouns are followed by post-positions** instead of prepositions. They generally remain as separate words in Hindi,

र खा क निता

Rekha ke pita

Rekha of father

Father of Rekha

उसक निता

Uske pita

Her (His) father

All **nouns are categorized as feminine or masculine**, and the **verb form must have a gender agreement with the subject**

ताला खो गया

Taalaa kho gayaa

Lock lose (past)

The lock was lost

चाभी खो गयी

Chaabhii kho gayeee

key lose (past)

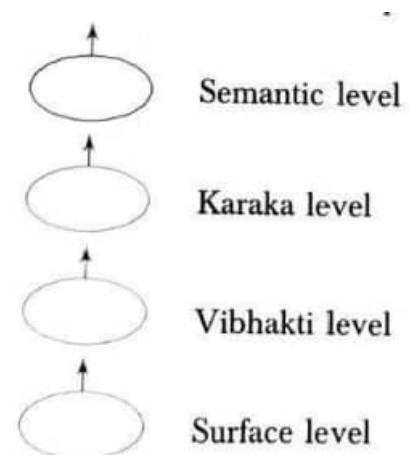
The key was lost.

Layered Representation in PG

The GB theory represents three syntactic levels: deep structure, surface structure, and logical form (LF), where the LF is nearer to semantics. This theory tries to resolve all language issues at syntactic levels only.

Paninian grammar framework is said to be syntactico-semantic, that is, one can go from surface layer to deep semantics by passing through intermediate layers.

- The **surface** and the **semantic levels** are obvious. The other two levels should not be confused with the levels of GB.
- Vibhakti** literally means inflection, but here, it refers to word (noun, verb, or other) groups based either on case endings, or post-positions, or compound verbs, or main and auxiliary verbs, etc
- Karaka (pronounced Kaaraka) literally means Case, and in GB, we have already discussed case theory, θ -theory, and sub-categorization, etc. Paninian Grammar has its own way of defining Karaka relations.



Karaka Theory

- Karaka theory is the central theme of PG framework.

- Karaka relations are assigned based on the roles played by various participants in the main activity.
- Various Karakas, such as Karta (subject), Karma (object), Karana (instrument), Sampradana (beneficiary), Apadan (separation), and Adhikaran (locus).

Example:

माँ बच्ची को आँगन में हाथ से रोटी खखलाती है।

Maan bachchi ko aangan mein haath se rotii khilaatii hei

Mother child-to courtyard-in hand-by bread feed (s).

The mother feeds bread to the child by hand in the courtyard.

- 'maan' (mother) is the *Karta*, Karta has generally 'ne' or 'o' case marker.
- rotii (bread) is the *Karma*. ('Karma' is similar to object and is the locus of the result of the activity)
- haath (hand) is the *Karana*. (noun group through which the goal is achieved), It has the marker “dwara” (by) or “se”
- 'Sampradan' is the beneficiary of the activity, e.g., bachchi (child).
- 'Apaadaan' denotes separation and the marker is attached to the part that serves as a reference point (being stationary). It takes the marker “ko” (to) or “ke liye” (for).
- aangan (courtyard) is the *Adhikaran* (is the locus (support in space or time) of Karta or Karma).

Issues in Paninian Grammar

The two problems challenging linguists are:

- (i) Computational implementation of PG, and
- (ii) Adaptation of PG to Indian, and other similar languages.

However, many issues remain unresolved, specially in cases of shared Karak relations. Another difficulty arises when mapping between the Vibhakti (case markers and post-positions) and the semantic relation (with respect to verb) is not one to one. Two different Vibhakti can represent the same relation, or the same Vibhakti can represent different relations in different contexts.

2.3 Statistical Language Model

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).

2.3.1 n-gram Model (<https://www.youtube.com/watch?v=Vc2C1NZkH0E>)

Applications: Suggestions in messages, spelling correction, Machine translation, Handwritten recognition...

It is a statistical method that predicts *the probability of a word appearing next in a sequence based on the previous "n" words*.

Why n-gram?

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

$$\begin{aligned} P(s) &= P(w_1, w_2, w_3, \dots, w_n) \\ &= P(w_1) P(w_2/w_1) P(w_3/w_1 w_2) P(w_4/w_1 w_2 w_3) \dots \\ &\quad P(w_n/w_1 w_2 \dots w_{n-1}) \\ &= \prod_{i=1}^n P(w_i/h_i) \end{aligned}$$

where h_i is history of word

w_i , defined as $w_1 w_2 \dots w_{i-1}$

So, in order to calculate sentence probability, we need to calculate the probability of a word, given the sequence of words preceding it. *This is not a simple task.*

An 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.

$$P(w_i/h_i) = P(w_i/w_{i-n+1} \dots w_{i-1})$$

Thus, an n-gram model calculates $P(w/h)$ by modelling language as Markov model of order n-1, i.e., 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 estimate, the probability of a sentence can be calculated as:

$$\begin{aligned} P(s) &\approx \prod_{i=1}^n P(w_i/w_{i-1}) \\ \text{and} \quad P(s) &\approx \prod_{i=1}^n P(w_i/w_{i-2} w_{i-1}) \end{aligned}$$

Example: The Arabian knights are fairy tales of the east bi-gram approximation - $P(\text{east}/\text{the})$, tri-gram approximation - $P(\text{east}/\text{of the})$ One pseudo-word <s> is introduced to mark the beginning of the sentence in bi-gram estimation.

Two pseudo-words <s1> and <s2> for tri-gram estimation.

How to estimate these probabilities?

1. Train n-gram model on training corpus.
2. Estimate n-gram parameters using the maximum likelihood estimation (MLE) technique, i.e., using relative frequencies.

- Count a particular n-gram in the training corpus and divide it by the sum of all n-grams that share the same prefix
- 3. The sum of all n-grams that share first n-1 words is equal to the count of the common prefix $W_{i-n+1}, \dots, W_{i-1}$.

$$P(w_i / w_{i-n+1}, \dots, w_{i-1}) = \frac{C(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C(w_{i-n+1}, \dots, w_{i-1})}$$

Example tri-gram:

"Corpus"

The girl bought a chocolate
The boy ate the chocolate
The girl bought a toy
The girl played with the toy

Input: The girl

$$P(\text{bought} | \text{The}, \text{girl}) = \frac{\text{count}(\text{The}, \text{girl}, \text{bought})}{\text{count}(\text{The}, \text{girl})}$$

Tri-gram
Prefix for "bought"

$$P(\text{bought} | \text{The}, \text{girl}) = \frac{2}{3} = 0.67$$

"Vocabulary"

{the, girl, bought, a,
chocolate, boy, ate, toy,
played, with}

$$P(\text{played} | \text{The}, \text{girl}) = \frac{\text{count}(\text{The}, \text{girl}, \text{played})}{\text{count}(\text{The}, \text{girl})}$$

$$P(\text{played} | \text{The}, \text{girl}) = \frac{1}{3} = 0.33$$

Probabilities for our vocabulary for Input: The girl....

the : 0
girl : 0
bought : 0.67
a : 0
chocolate : 0
boy : 0
ate : 0
toy : 0
played : 0.33
with : 0

Predicted word for "The girl **bought**"

Example

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:

$P(\text{the}/<s>) = 0.67$ $P(\text{Arabian}/\text{the}) = 0.4$ $P(\text{knight}/\text{Arabian}) = 1.0$
 $P(\text{are}/\text{these}) = 1.0$ $P(\text{the}/\text{are}) = 0.5$ $P(\text{fairy}/\text{the}) = 0.2$

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.

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

Limitations:

- Multiplying the probabilities might cause a numerical underflow, particularly in long sentences. To avoid this, calculations are made in log space, where a calculation corresponds to adding log of individual probabilities and taking antilog of the sum.
- The n-gram model **faces data sparsity**, assigning zero probability to unseen n-grams in the training data, leading to many zero entries in the bigram matrix. This results from the assumption that a word's probability depends solely on the preceding word(s), which isn't always true.
- Fails to capture long-distance dependencies in natural language sentences.

Solution:

- A number of smoothing techniques have been developed to handle the data sparseness problem.
- *Smoothing in general refers to the task of re-evaluating zero-probability or low-probability n-grams and assigning them non-zero values.*

2.3.2 Add-one Smoothing

- It adds a value of one to each n-gram frequency before normalizing them into probabilities. Thus, the conditional probability becomes:

$$P(w_i | w_{i-n+1}, \dots, w_{i-1}) = \frac{C(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C(w_{i-n+1}, \dots, w_{i-1}) + V}$$

Where, V is the vocabulary size.

- **Yet, not effective, since it assigns the same probability to all missing n-grams, even though some of them could be more intuitively appealing than others.**

Example:

Consider the following toy corpus:

- "I love programming"
- "I love coding"

We want to calculate the probability of the bigram "I love" using **Add-one smoothing**. **Step**

1: Count the occurrences

- **Unigrams:**

- "I" appears 2 times
- "love" appears 2 times
- "programming" appears 1 time
- "coding" appears 1 time

- **Bigrams:**

- "I love" appears 2 times
- "love programming" appears 1 time
- "love coding" appears 1 time

- **Vocabulary size sV:** There are 4 unique words: "I", "love", "programming", "coding".

Step 2: Apply Add-one smoothing

For the bigram "I love":

$$P(\text{"love"} \mid \text{"I"}) = \frac{\text{Count}(\text{"I love"}) + 1}{\text{Count}(\text{"I"}) + V}$$

$$P(\text{"love"} \mid \text{"I"}) = \frac{2 + 1}{2 + 4} = \frac{3}{6} = 0.5$$

So, the probability of "love" following "I" is 0.5 using **Add-one smoothing**.

Step 3: For an unseen bigram

Let's say we want to calculate the probability for the bigram "I coding" (which doesn't appear in the training data):

$$P(\text{"coding"} \mid \text{"I"}) = \frac{\text{Count}(\text{"I coding"}) + 1}{\text{Count}(\text{"I"}) + V}$$

Since "I coding" does not appear in the training data, $\text{Count}(\text{"I coding"}) = 0$, so:

$$P(\text{"coding"} \mid \text{"I"}) = \frac{0 + 1}{2 + 4} = \frac{1}{6} \approx 0.167$$

The bigram "I coding" now has a non-zero probability, thanks to Add-one smoothing.

2.3.3 Good-Turing Smoothing

- Good-Turing smoothing improves probability estimates by adjusting for unseen n-grams based on the frequency distribution of observed n-grams.
- It adjusts the frequency f of an n-gram using the count of n-grams having a frequency of occurrence $f+1$. It converts the frequency of an n-gram from f to f^* using the following expression:

where n is the number of n-grams that occur exactly f times in the training corpus.
 As an example, consider that the number of n-grams that occur 4 times is 25,108 and the number of n-grams that occur 5 times is 20,542. Then, the smoothed count for 5 will be:

$$f^* = (f + 1) \frac{n_{f+1}}{n_f} \quad \frac{20542}{25108} \times 5 = 4.09$$

2.3.4 Caching Technique

The **caching model** is an enhancement to the basic n-gram model that addresses the issue of frequency variation across different segments of text or documents. In traditional n-gram models, the probability of an n-gram is calculated based solely on its occurrence in the entire corpus, which does not take into account the local context or recent patterns. The **caching model** improves this by incorporating the **recently discovered n-grams** into the probability calculations.