Unit -1

finding the structure of words
- words and their components
- Issues and challenges
- Morphological models

finding the structure of Documents
- Introduction
- methods
- complexity of the approaches
- performances of the approaches
- features.

Defination of NLP

Natural language processing is a cross disciplianry field of linguistic, computer science and AI. It is releated to interactions blw digialized computing devices and human languages.

-The NLP deals with designing and programming digital competitional devices to process and analysis large amount of natural language data

Natural language processing (NLP) is a field of AI that enables machine to understand, interpret, and generate human language through compatational techniques.

-Applications of NLP

1. spam dectation
2. machine translation
3. speech recognization
4. chat bot
5. sentement analysis.

phases of NLP:-

1. lexical analysis / morphological analysis
2. syntatic analusis
3. sematic analusis
4. disclosure Integration
5. pragmatic analysis.

1. Lexical Analysis
- Breaking down text into smaller units like paragraphs, sentences and words (tokenization)
- Data cleaning techniques such as lemmation, stopwords removal, and correcting misspelled words.
- Identify morphems, the smallest units of meaning in a word.

2. Syntactic Analysis (Parsing):-
- understaning the structure of a sentintee and its grammatical correctness.
- Assigning parts-of-speech (POS) tags to each word (e.g., nouns, verbs)
- Ensurning logical consistency and reducing ambiguity.

3. Semantic Analysis:-
- Extracting the meaning of text.
- Understanding dictionary definitions and usage in context.
- Tasks like Named Entity Recognition (NER) and word sense Disambiguation (WSD)

4. Discourse Integration
- comprehending relationships between the current sentence and previous sentences or larger context.
- Contextual references and anaphora resolution (e.g. understanding pronouns).

5. Pragmatic Analysis
- Interpreting the inferred meaning beyond literal content.
- Understanding intentions, implications, and underlying assumptions.
- Examples include figurative expressions and contextual interpretations.

Words and their components

words are nothing but a meaningful unit of sentences. by using this kind of words we can construct meaningful sentintence

I | love | reading | Research papers

↓

word or Token.

1. Tokens

Dividing sentence into words. The words are called tokenks

3 types of tokens.

1. word token
2. character token
3. subword token.

2. lexems

Based on canonical forms of words

running ran

run → base word
or
canonical form of word

In NL a word often denotes one linguistic form in a given context and the concept behind the form and set of alternative forms. That can express it such sets are called leme's or lexical items.

lexems can be divided by their behaviour into the lexical categories of verbs. noun, adjectives, conjunctions or other parts of speech.

3. Morphems

The morphems it nothing but smallest meaningful unit of any language

Two types of morphems:-

1. free
2. Bound

free:-

are nothing but this morphems are independent of another morphems.

ex:- appoint

Bound:-

Dependents on other morphoms.

ex:- dis|appoint|ment.

4. Typology:-

Refers to classification or categriation of language based on the structure of or grammatical features.

It is used to identify common patterns or variations across the language under their properties or relationships.

word = $w_1 + w_2 + w_3$

e.g: disagreement = dis + agree + ment

This model is used to analysis by using different unity of words by combining those words one particular word will be formed.

d. Functional based:-

It is also used to analysis the structure of a word from the given statement

5. Morphology Induction:-

which analyzes the pattern.

Issues and challenges

→ 1) Irregularity
2) Ambiguity
3) productivity

Irregularity :-

It means existence of such forms and structures, that or not describe appropriately by a protypical linguistic model.

Some irregularites can be understood by redesigning the model, and improving it rules. other lexical depending irregularites often can't be generalized.

1. Irregular verbs & nouns
2. Exceptional inflection

1) Irregular verbs & nouns :- doesn't follow regular pattern.

2) Exceptional inflection :-
comparative & superlative

big	bigger	biggest
dark	darker	darkest
good	great	best

2. Ambiguity :-
(multiple meanings)
word forms that can be understand multiple ways out of context
There are different types
1. word sense ambiguity (particular word differ meaning)
2. parts of speech " (particular word changing the context)
3. Structural "

we can write a particular sentence in multiple ways.

3. Productivity :-
Ability to generate new words or word forms using productive rules
examples :-
generates new words by taking two words.

Morphological models.
There are 5 types of morphological models
1. Dictionary lookup
2. finite state
3. Unification
4. functional
5. morphological induction

a. Dictionary lookup
It is use to analyze the particular word (or) specific word.
It is used to analyze the structure of a particular word

Process of analyzing the structure of word :-

1. selecting word.
2. convert the selected word based form or canonical
3. Search for a particular word in the dictionary.
4. If it match with the base word or canonical word. it give info about particular word.

A dictionary is a data structure that directly enables the precomputed word analysis.
Dictionaries enumerate the set of association b/w word form & their descriptions.

b. finite state morphology
It is also used to analyze the structure of a word.
But it is uses the finite automata
Ex. formal language theory.

c. Unification
unification is the key operation by which feature structures can be

Finding structure of Document

Language is structured and analyzing this structure helps various NLP tasks like parsing machine translation, and semantic role labeling. Identifying sentence boundaries improves readability in speech recognition systems while topic segmentation determines while a topic starts and ends in a document.

1. Sentence Boundary detection

Identifying where sentences begin and end in a sequence of text.

challenges:-

- Abbreviations (e.g., "Dr". ~~in~~ "
- Quoted sentences may contain multiple sentece boundaries

2. Topic Boundary detection.

Dividing text into meaningful topic-based segments.

Applications:-

used in IR & ext summarization

UNIT - II

~~Core Concepts: Introduction to parts~~ Analytics

Types

15 x 15 + 15 =

Finding the structure of Document

In human language whats and sentences don't appear ~~dappear~~ randomly usually have a structure

---

ex: combination of words form sentence meaningful grammatical units such as statements, commands e.g. command

If the structure of document is extracted it makes further processing of text easy in NLP

The NLP tasks the depends on the document structure or parsing, machine translations, & semantic role and labeling in sentences.

Sentence boundary detection:

is the problem in NLP of deciding where the sentences begin & end

The task is also called a sentence segmentation. deals with segmenting.

a sequence of word token token

The process of segmenting sequence of words into units is nothing but sentence boundary detection.

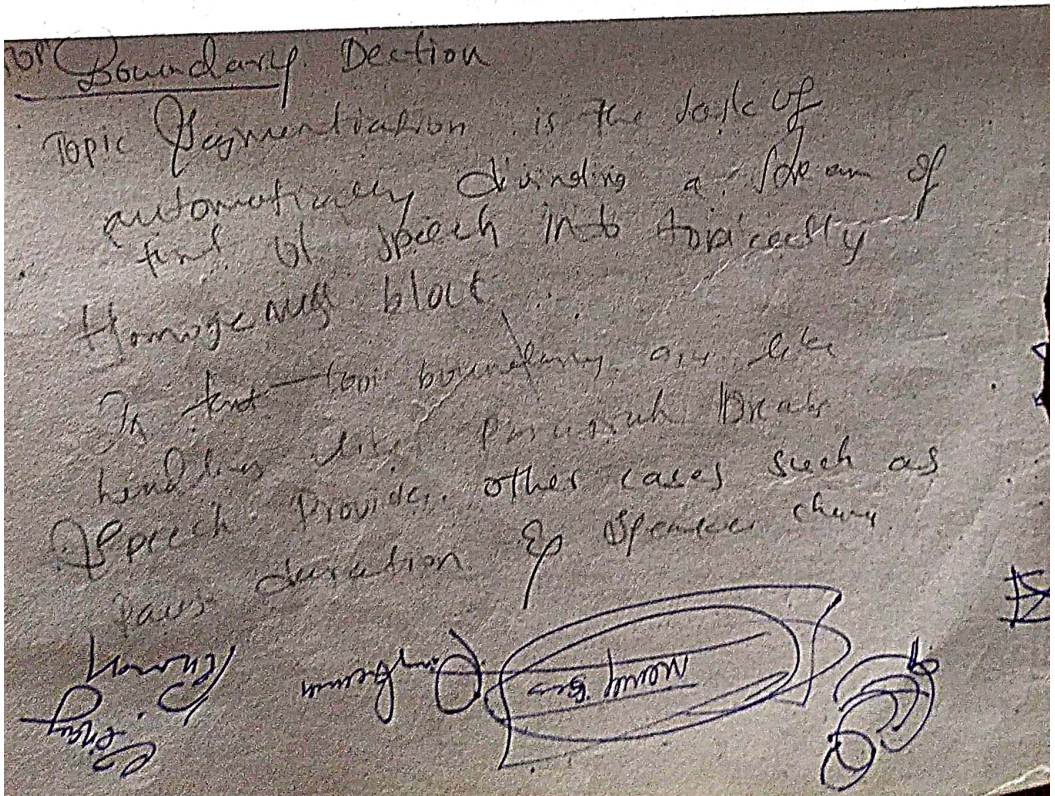
Ex:-

I talked to Dr. Vishn ~~[illegible]~~ and i explain about health.

5) Boundary Detection

Topic Segmentation is the task of automatically dividing a stream of text or speech into thematically Homogeneous blocks.

In text boundary are like handling data punctuation Breaks Speech Provides other cases such as pause duration Ep Speaker change.



methods

2 types

1. Generative sequence classification method.
2. Discriminative local classification method

Generative :-

It refers to the techniques that utilies Generative models to classifie sequences of text by first learning the underlying Probability distribution of data across different classes.

It allowing them to generate to new text samples similar to each class. And then classify new input sequence, by comparing them these generative samples.

→ These methods have models which have not only predicting most likely labels on but also Generating the sequence itself based on learned probabilites.

→ It is a cable of Predicting the mostly likely label of sentence, or boundary of topic.

Discriminative local classification method

It will only consider local features of a particular sentence and then it will predict to particular sentence.

These approach focus on making decision based on local info.

→ we directly learn the decision boundary blw different classes in the I/p space rather modeling the entire data distribution.

Complexity of the approaches :-

The complexity can be measured by rating training & prediction and also measured by their performance of the system

Predicting means they will predicting new sentence based on training examples

performance of the approaches :-

we are going to calculate the Performcence

$$\text{Error percentage} = \frac{\text{No. of errors}}{\text{No. of examples}}$$

$$\text{precision} = \frac{\text{Total no. of correct positive prediction}}{\text{Total no. of positive prediction}}$$

$$\text{Recall} = \frac{\text{Total no. of all correct positive predict}}{\text{Total no. of positive instance}}$$

$$\text{F1 score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Unit – II

Syntax I:

- parsing Natural Language
- Treebanks:
  - A Data-Driven Approach to Syntax
- Representation of Syntax structure
- Parsing Algorithms.

Parsing Natural language

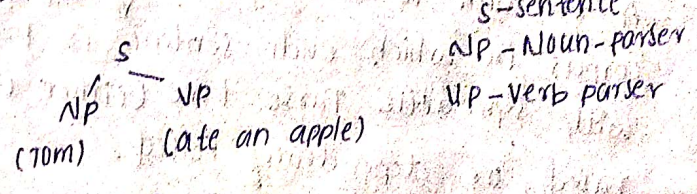
Parsing refers to analyzing a sentence to determine its grammatical structure based on a given formal grammar

It is essential for applications like text-to-speech summerization, question answering and more

(or)

Parsing means analyzing a sentence to understand its structure - like breaking it into smaller parts (subject, verb, object, etc) so a computer can understand it.

ex:- "Tom ate an apple"



→ Output of parsing is a:

Parse tree or dependency tree

Types of parsing structures :-

1. Constituency Tree (phrase structure)
   Breaks the sentence into nested phrases (like Noun phrases, verb phrases)
2. Dependency tree
   shows word-to word relationships, like
   - "ate" → main verb
   - "Tom" → subject of "ate"
   - "apple" → object of "ate"

What does a parser do

1. Takes a sentence
2. Uses grammar rules
3. Builds a tree (either dependency or constituency)
4. Helps applications like translation, voice commands etc.

Challenges in NL parsing:-

1. Ambiguity 2. Incomplete Grammar
3. Robustness 4. Performance.

Treebanks : A Data-Driven Approach to syntax

- A treebank is a linguistically annotated corpus in which each sentence is provided with syntactic parse tree (either consituency based or dependency based.

→ Corpus means collection of something

Construction of a Treebank:-

- Built by linguists manually or through semi-automatic tools.
- Each sentence is:
  - pos tagged (part-of-speech tagging)
  - Annotated with syntatic structure.
- Requires annotation guidelines to maintain consistency across all sentence.

- Parsing involves recovering syntactic information that is not explicitly present in a sentence.

- A parser needs knowledge (syntatic rules) beyond just the sentence to generate proper syntatic analysis.

- Traditionally, this knowledge is represented as a grammar (CFG), but natural language syntax is too complex to fully specify with explicit CFG rules.
- Beside rules, we must know which possible syntatic analysis is most plausible (probably) for a given sentence. This is the second Knowledge Problem.

- Treebank Provides a data-driven soln to both these problems.
- A treebank is a corpus of sentence where each sentence has a complete syntatic analysis provided by human experts
- These annotations represent the most probable analysis for each sentence.
- The treebank does not explicitly provide grammar rules; instead the syntatic analyses serve as direct knowledge for parsing.

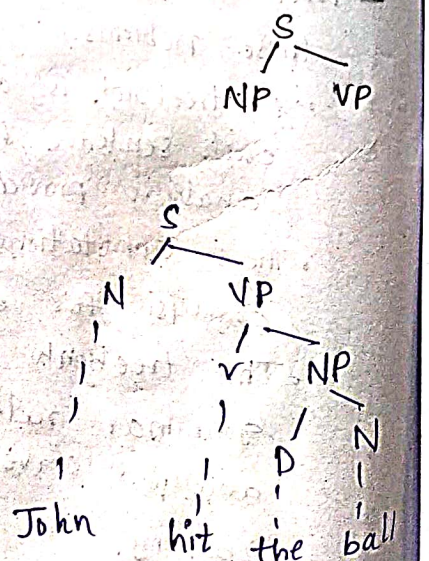
• Treebanks enable parsers to:
  - avoid explicitly encoding grammar rules
  - use supervised machine learning on the annotated data to learn how to choose the best parse for new sentences.

- Two main forms of syntax representation in tree banks.
  - phrases structure trees (constituency-based)
  - Dependency graphs.

- Treebanks ensure consistent, high-quality annotation by following detailed guidelines during the human annotation process.

Ex:- John hit the ball

S → NP VP
VP → V NP
NP → DN
V → hit
D → the
N → John | ball



Representation of Syntactic structure.

It can be represented in two major ways:

- Dependency Graphs
- phrase structure trees (or) Constituency Trees.

Syntax analysis using Dependency Graphs

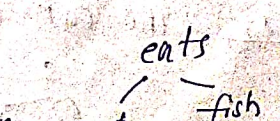
- A dependency graph represents the syntactic structure by directly linking head words with thier dependent using directed arcs.

Key:-

- words in the sentence are the only nodes in the graph
- Arcs show head → dependent relations.
- One word acts as the root (index 0), representing the main action or verb
- It's a form of tree structure, where each word (expect the root) depends on exactly one head.
- It is asymmetric and directional
- The aim is to find the head of each word.

Ex:- "The cat eats fish"

word	Head	Relation
The	cat	det
cat	eats	nsubj
eats	0	root
fish	eats	obj



usage:-

- Preferred for free word order languages like Czech, Turkish.
- Used in prague Dependency Treebank
- common in modern NLP application.

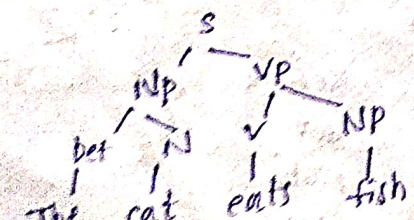
Syntax Analysis using Phrase structure Trees

Phrase structure Trees(or Constituency Trees) breaks a sentence into nested phrases like Noun phrase (NP) and Verb phrase (VP) based on context-free grammar (CFG)

key:-

- Derived from generative grammar.
- shows hierarchical grouping of words into constituents.
- Helps handle long-distance dependencies like questions, gaps, and wh-movement.
- Each sentence is analyzed based on subject predicate structure.

Ex:- "The cat eats fish"



NP - "the cat" - Noun phrase

VP - "eats fish" - verb phrase

- Dependency Graphs are simple and effective for passing languages with flexible word order. They model 'word-to-word relations directly.
- phrase structure Trees are useful for analyzing nested grammatical structures and are better suited for languages with more fixed word order
- Both methods help address the knowledge acquisition problem in syntax by provided structured syntatic annotations.
- Treebanks use these representations to create supervised data for training parsers in NLP.

ie Dependency Graph	phrase structure Tree (constituency tree)
shows direct binary relations between words	shows hierarchical grouping of words into phrases.
words (head and dependent)	phrases (NP, VP, etc)
Directed graph, with arcs as dependencies	Tree structure with nested constituents.
Explicit(main verb/root symbol is at index 0)	implicit (topmost node is S - sentence)
Head-dependent relations	constituent phrases.

Free word order languages (Czech, French)	Fixed word order languages (English, French)
single words connected via arcs	Groups of words forming phrases.
less complex, more compact	More detailed and hierarchical.
Machine learning parsing models, information extraction.	Understanding sentence structure, grammar teaching.
~~Arcs~~ from head to dependent (graph form)	Tree with nodes and branches (top-down hierarchy)
Relation labels (e.g nsubj, obj, det)	phrase labels (e.g., NP, VP, S)

Parsing Algorithms :-

- Parsing algorithms helps in analyzing the syntatic structure of NL sentences
- Parsing algorithms help in understanding by applying the grammatical rules, allowing systems to generate grammatically correct sentence.

4 types of parsers:-

1. Recursive Decent Parser
2. Shift-reduce parser
3. Chart parser
4. Regex parser.

1. Recursive Decent Parser

- It is a type of parsing techniques
- It is a top down parser and reads the input from left to right.
- It works by recursively descending through the input string, checking each token against a set of production rules defined by the grammar of the language.
- The process continues until continues until the entire input string has been successfully parsed or until an error is encountered.

'3+4 * (5-2)'

Grammar :-

expression → term(('+'|'-') term)*

term → factor(('*'|'/') factor)*

factor → NUMBER | ('expression')'

expression(3+4 * (5-2))

term()

factor() → parse the number 3

expression() → parse '+' operator

term()

factor() → parse the number '4'

expression()

term() → parse '*' operator

factor() → parse '(' and '5'

expression() → parser the '-' operator

term()

factor() → parse the number '2' and ')'

Successfully parsed.

Key :-

- Simple and intuitive
- Works well for unambigous grammars
- Cannot handle left-recursion.
- Commonly used in hand written parsers.

2. Shift Reduce parser

- Shift reduce parser is used to reduce a string to the start symbol of a given grammar
- Two parts

1. stack :-
   shift reduce parser uses a stack to hold / store the grammar

2. Input tape :
   Shift reduce parser uses input tape to hold the string.

- This parser performs two actions ;
  shift and reduce that is why it is known as shift reduce parser.
- At the shift action, the current symbol in the input string is pushed to the stack.
- At the reduction, the symbols will be replaced by the non-terminals.

ex :- production rules

T → T+T

T → T-T

T → (T)

T → (O) C

string :-

$C_1 - (C_2 + C_3)$

stack	Input	action (shift / reduce)
$	$C_1 - (C_2 + C_3)$ $	shift
$ C_1	$- (C_2 + C_3)$ $	reduction T → C
$ T -	$(C_2 + C_3)$ $	shift

$T-(	(2+(3)$	shift
$T-((2	+(3)$	shift
$T-(T	+(3)$	reduce T→c
$T-(T+	c3)$	shift
$T-(T+c3	)$	shift
$T-(T+T	)$	reduce
$T-(T	)$	reduce
$T-(T)	$	shift
$T-T	$	reduce
$T	$	reduce
$T	$	Accept

Key:-

- uses stack & buffer
- Good for bottom-up parsing
- used in compiler design
- can handle ambiguous grammars using operator precedence.

* chart parser

→ A chat parser is a type of top-down parsing technique.

It is used to analyze the syntatic structure of the given sentences based on a given grammar.

The chat parser is incrementally builds parse trees by combining smaller constituents into larger ones according to the rules of the grammar.

Grammer rules

S → NP, VP
NP → deter, N
VP → NP, PP
VP → V, NP
VP → N
PP → P, NP

Ex:- The cat chased the mouse
↓ ↓ ↓ ↓ ↓
Det Noun verb Det Noun

4. Regex parser

It also called as regular expression parser

It is used to identify patterns or sequences of characters. extract data based on predefined regular expressions.

The main use of regex parser, when a particular sentence is given and to find out some words in it

In some cases the regex parser is validating the data

Assignment :-

1. Explain the problems by using context free grammar for syntatic anaalysis of a NL.
2. Write a notes on tree bank approach of syntax analysis.
3. Explain about syntax analysis using dependecince graph
4. Explain the working of shift reduce parser.
5. Explain different types of parsing techniques briefly.

Unit-III

Syntax II :-

- models for Ambiguity Resolution in parsing
- Multilingual Issues

Semantic parsing I :-

- Introduction
- semantic Interpretation.
- System paradigms
- word sense

What is Ambiguity in parsing

Ambiguity in parsing occurs when a sentence can be analyzed (parsed) in more than one way due to unclear grammar or structure.

Ex:- "I saw the man with the telescope"

- Who have the telescope? you or the man?

There sentence has multiple valid parse trees.

Models for ambiguity resolution

3 models for ambiguity resolution

1. Probability or Probabilistic context free Grammar
2. Generative model
3. Discriminative model

1. Probablistic content free Grammar

- PCFG is a statistical model for parsing NL
- It is an extension of CFG by assiging the probabilites to production rules
- commonly used for ambiguity resolution.

Why use it?

To resolve ambiguity by choosing the parse tree with the highest probability.

steps:-

1. Define a set of production rules which specify how sentence are formed from constituent part
2. Train the PCFG model with the text. During training, assign production rules with probability

probability = (frequency of the rule) / (total no. of times of non-terminal.

3. Try different parse trees to combine words
4. choose the best parse tree from it.

ex:- CFG

S → NP VP
VP → V NP | V NP PP
PP → P NP
NP → Det N | NP PP

Now add probabilites (PCFG):

S → NP VP [1.0]
VP → V NP [0.3] | V NP PP [0.5]
PP → P NP [1.0]
NP → Det N [0.7] | NP PP [0.3]

sentence :-

"I saw the man with the telescope"

This has two parse trees:

1. "with the telescope" modifies the man (NP → NP PP)
2. "with the telescope" modifies the verb (VP → V NP PP)

PCFG calculates the total probability of each tree and chooses the most probable one.

2. Generative model

It generates all possible parse trees for a given sentence and selects the most likely interpretation based on a probablisitic framework.

- They model how sentences are generated
- assign a joint probability to both the sentence and its parse tree.
- work by assuming a hidden structure that generates observed data.

steps:-

1. Define a set of production rules
2. Generate the possible parse trees for the given sentence.
3. Assign the probabilites score for each parse tree of sentence
4. Choose one parse tree which is have more probability

ex:- "The cat sat on the mat"

- Generates a parse tree : S → NP VP
- generates words from terminals
  NP → Det N . Det → "The", N → "cat", etc
- Assign probabilites to each step

The goal is to compute

P(sentence, parse tree) = P(parse tree) × P(sentence | parse tree)

Then choose the tree with the highest probability.

3. Discriminative models.

This model is used to predict the most like parse tree for the given sentence without explicitly generating all possible trees.

- They directly learn to distinguish blw correct and incorrect parses.
- Focus on P(parse/sentence) instead of P(sentence, parse)

steps:-

1. Define a set of features to capture relevant information about the sentence. These features include lexical information, syntatic features and contextual features.
2. Train the classifier with example sentences paired with correct parse trees
3. It predicts the most likely parse tree for the given new unseen sentence.

Ex:- "Flying planes can be dangerous"

Two parses:

1. "flying" as verb: planes that can flying are dangerous.
2. "flying" as a noun: the act of flying planes is dangerous.

A discriminative parser will

- Extract features like POS tags, dependency paths, content
- score each parse tree
- pick the tree with highest score.

Summary :-

- PCFG
  - Adds probabilites to CFG rules; resolves ambiguity using the most likely parse.
- Generative models
  Model how a sentence and its parse are generated together.
- Discriminative models
  Learn to directly select the best parse using features and data.

Multilingual Issuses

1. ~~Token~~ What does multilingual parsing mean?

Multilingual parsing involves developing systems that can analyze and understand the syntax of multiple languages, not just englist

But this is challenging because different languages:

- follow different grammatical rules
- Have different word orders
- Use rich morphology.

1. Tokenization

Tokenization is the process of breaking a sentence into smaller units, typically words or subwords

Example:-

"I love playing Cricket"

→ ["I", "love", "playing", "cricket", "."]

Why it's challenging in multilingual NLP

- English:
  uses spaces clearly b/w words - relatively easy.
- Languages like Chinese, Japanes, Thai:
  No space between words.
- Ambiguity in compound words:
  Tokenizer must know where to split.
- Case sensitivity can be an issue in tokenization.

How to handle:-

- language-specific tokenizers
- Subword tokenization
- Neural tokenizers.

2. Word Segmentation

Closely related to tokenization, word segmentation refers specifically to detecting word boundaries in languages that don't use space.

Issues:-

- No spaces in many Asian languages → hard to segment.
- Ambiguous segmentation:-
  Multiple valid segmentations possible.
- In languages like Turkish & Finnish a single word can encode complex meaning through affixes.
- Languages like German forms compound words by concatenating multiple small words

soln:-

- Statistical models
- Dictionary-based approaches
- Neural models.

3. Morphology

Morphology studies the structure of words how they are formed by root, prefixes suffixes etc.

→ Breaking a word into morphems.

Issues:-

1. When we try to understand sentencies in some languages, because some words can change the meaning based on sentence

- By using we can identify the context of the grammar.

- Rich morphology in many languages
  - A single root can have dozens of forms.
- Data sparsity
  - Many word forms = low frequency = hard for models to learn.
- Ambiguity in affixes
  - suffixes can mean different things in different contexts.

soln
- Morphological analyzers
- stemming and lemmatization
- subword tokenization
- character-level models.

Semantic parsing - I

Introduction :-

What is semantic parsing ?

The process of understanding the meaning and interpretation of words, signs and sentence structure is called semantic parsing.

(or)

Semantic parsing is the process of converting natural language into machine-understandable meaning (like logical form or code)

- Using semantic parsing, the computers can understand Natural language the way humans do.
- It is the toughest phase and not fully solved.
- semantic → study of meaning
- parsing → identify and relate pieces of information.
- Purpose :-
  To make computers understand what a sentence means so they can act on it.

Ex :- "Sanchon serves vegetarin food"
→ output :- serve(sanchon, Vegetrianfood)
(This is called a meaning representation)

Two types:-

Type	meaning	use case
Deep parsing	detailed, domain specific meanings	e.g:- Travel apps, chatbots.
shallow parsing	General meaning (less detailed)	e.g: word sense, sentiment apps.

Semantic Interpretation

Semantic Interpretation is the process of exactracting the meaning of a sentence from its syntactic structure (grammar tree or parsed sentence).

It converts "how words are arranged" into what they actually mean.

Why it is important?

Because:

- Computers need to understand not just the words, but what is being said.
- without sematic interpretation, even a grammatically correct sentence may remain meaningless to machines.

It involves identifying predicates and arguments, resolving ambiguites and pronouns and producing a formal representing like logic.

Ex:- "John gave Mary a gift"

→ predicte: gava
→ Arguments: John (subject), Mary (indirect object), gift (direct object)
→ Representation: give (John, mary, gift)

~~A~~ ~~Semantic~~

- Katz and Fodor in their 1963 paper "The structure of a semantic Theory" put forward what properties a semantic theory should process.

A sematic theory should be able to:

1. Explain sentences having meanings.
2. Resolve the ambiguities of word in content
3. Identify meaningless but syntatically well-formed sentences.

Building blocks of sematic interpretation

1. structural Ambiguity
2. word sense
3. entity & event resolution
4. Predicate - Argument structure
5. Meaning representation.

1. Structural ambiguity

Structural ambiguity happens when a sentence can be parsed in multiple ways due to its structure - even though the words are the same.

Same sentence, different meanings

Ex:- "I saw the man with a telescope"

This can mean:-

1. I used a telescope to see the man.
2. The man I saw had a telescope.

Ambiguity is in how the parse "with a telescope" is attached (to verb or noun).

2. Word sense (Disambiguation)

Word sense refers to the correct meaning of a word in context, especially when the word has multiple meanings.

Word sense Disambiguation (WSD)

The task of selecting the correct meaning of a word based on the surrounding words.

Ex:-

- "He went to the bank to deposit money"
  → bank = financial institution.
- "He sat by the bank of the river".
  → bank = riverbank

3. Entity and Event Resolution

Entity Resolution:

Identifying and linking different references to the same real-world entity.

Ex:- "Barack Obama was born in Hawaii. He became president in 2008"

→ "He" = Barack Obama (resolved by coreference)

Event Resolution:-

Identifying and tracking events and their (who did, what, when, and where)

Ex:-

"The company launched a new product. The launch was a success".

→ Both sentences refer to the same event: the product launch.

4. Predicate-Argument Structure

This structure defines the main action (predicate) in a sentence and the entities involved (arguments)

"Who did what to whome?"

Format:- predicate (Arg1, Arg2, ...)

Ex:- "Sita gave a book to Ram"

→ predicate: gave

→ Arguments:

- Arg1 (subj) = Sita
- Arg2 (obj) = book
- Arg3 (indirect obj) = Ram

→ Logical form: Give (Sita, Book, Ram)

5. Meaning Representation

It is a formal way to express the meaning of a sentence - often used by machines to reason, query or respond.

Ex:- "The dog chased the cat"

Meaning representation: chase (Dog, cat)

Type	Ex
first order logic	Loves (John, Mary)
frames / scripts	{ action: "give", giver: "sita" receiver: "Ram" }
Lambda calculus	used for functional compositional meanings.

System Paradigms

System paradigms refer to the different ways semantic parsing systems are designed and functions depending on:

1. How they are built (architecture)
2. Where they are used (scope)
3. How deeply they understand (coverage)

Think of paradigms as different "styles" or "models" used to create semantic parsers.

The historic approaches which are more prevalent and successfully generally fall into 3 categories.

a. Based on Architecture

This classification is based on how the semantic parsing system is built.

4 major architecture-based paradigms:

1. Knowledge-Based systems.
   - It rely on manually written rules and Ontologies.
   - They do not learn from data; everything is pre-programmed.

   Ex:- ELIZA (early chatbot)

   If sentence contains "book a flight from x to y"
   Then BookFlight(x, y)

2. Supervised systems:

   These are trained using labled datasets where each sentence is paired with a correct meaning representation.

   Ex:- Neural sematic parser

   "Book a flight from Delhi to mumbai"

   BookFight(Delhi, Mumbai)

3. Unsupervised learning system.

   It learn from unstructured data without labels, using clustering or distribution similarity

4. Semi-Supervised system :-

combine small labeled data with large unlabeled data to improve performance. These are useful when labeled data is limited".

b. Scope

Scope refers to how general or spetic the system in terms of the domains or topics it can handle.

Two types of scope paradigms

scope Type	Description
Domain-Dependent	works one specific domain
Domain-Independent	works across many domains (general purpose)

Ex:-

1. Dependent
   "Book a flight from Delhi to mumbai"
   BookFlight (from = "Delhi", to = mumbai)
2. Independent
   - "Book a flight to Delhi"
   - "set an alarm for 6 AM"

c. Coverage

Coverage refers how deeply the system understands or analyzes the sentences.

Two types of coverage

Types	Description
shallow parsing	extracts basic info (like subjects, objects, verbs)
Deep parsing	Builds full semantic meaning representations

Ex:-

- shallow :-
  Input : "John ate an apple"
  - subject : John
  - verb : ate
  - Object : apple
- Deep :
  Eat (John, Apple)

Word sense :-

A word sense is a specific meaning of a word when used in a particular content

One word can have multiple senes (meanings depending on how & where it is used

Types of word ambiguity

Type	Description	Example
Homonymy	same word, unrelated meaning	Bat (animal) vs Bat (used in cricket)
Polysemy	same word, related meaning	Bank (finance) vs Bank (store money)
categorial Ambiguity	word used as different parts of speech	Book (noun) vs Book (verb)

Word sense Disambiguation (WSD)?

WSD is the process of identifying the correct sense of a word based on the context in which it appears.

- It's a subtask of semantic interpretation in NLP.

Approaches to word sense Disambiguation

There are 4 main approaches to performing WSD:

1. Rule-Based Approach
   - Uses manually written rules, dictionaries and thesauri.
   - Matches words with definations or contexts

   Ex:-

   If "bank" appears with "money", assign there = financial institution

   Algorithms or Techniques
   - Lesk Algo
   - Structural semantic Interconnections (SSI)

2. Supervised learning Approach
   - Trained on labeled data where the correct sense is already known.
   - Use: algo like decision trees, neural networks or transformers.
   - Advantages:-
     * It achieves high accuracy.

3. Unsupervised learning
   - unlabeled data. relies on context similarity
   - Groups words into clusters of similar usage.
   - No human-annoated data required.

   Ex:-

   "Eat", "consume", "devour" may appear in similar contents → grouped together.

   Adv:-

   No need for extensive manual annotation, making it scalable & adaptable to various languages.

   Limitations:-
   - May require sophisticated algorithm to achieve high accuracy.
   - performance can be lower.

4. Semi-Supervised
   - Starts with a small set of labeled data + large unlabeled data.
   - Uses bootstrapping or self-training to improve performance.

Difference b/w System Paradigms & word sense

System paradigms refer to the different architectual approaches used to build semantic parsing systems, such as rule-based or domain-specific designs.

In contrast,

Word sense Disambiguation is the process of resolving ambiguity in individual word meanings based on context, such as distinguish b/w "bank" as a financial institution or a riverbank.

Unit – IV

Semantic Parsing – II

- predicate - Argument structure
- Meaning Representation systems.

Predicate - Argument Structure

- Predicate-argument structure, also known as semantic role labelling, is a method used to identify the roles of different parts of sentence.

(or)

- Predicate - Argument structure describe how a verb (predicate) relates to other elements (arguments) in a sentence. It captures the roles of entities involved in an event or action.

It is a framework to describe the meaning of a sentence by breaking it down into:

- Predicate: The main verb or action in a sentence.
- Arguments: The entities involved in that action (subject, object, etc)

e.g:- It answers the question:
Who did what to whom, where, when and how?

Simple example

"Ravi gave a book to Arjun"

- predicate : gave
- Arguments:
  - Ravi (giver → subject)
  - a book (thing → object)
  - Arjun (reciever → indirect object)

Logical :- Give (Ravi, Book, Arjun)

Why is it important

- Helps machine understand sentence meaning
- Crucial for:
  - Semantic parsing
  - Question answering
  - Information extraction
- This is important for things like translating languages, answering questions, and even helping virtual assistants understand commands better.

- Semantic Roles (Argument Roles)

Each argument has a role in the event.

Role	Meaning	Example
Agent	Does of the action	Ravi ("in Ravi kicks the ball)
Theme	Thing being acted upon	Ball (in above ex)
Goal	End location or reciver	Arjun ("Ravi gave book to Arjun")
Source	starting point	Delhi ("she flew from Delhi")
Instrument	Tool used	knife ("He cut with a knife")
Location	where something happened	park ("He played in the park")

- Predicate-argument structure is part of semantic interpretation.
- It is useful for creating meaning representations that machines can understand and reason about.
- Helps break complex sentence into action + participants format.

(Who (Agent) → Did what (predicate) → To whom/what (Theme/Goal))

Tools used in NLP:-

- FrameNet
  - uses frames (e.g. Buying, Giving) to group roles
- PropBank
  - Labels arguments as Arg0, Arg1, etc.

PropBank and FrameNet are two important tools used in semantic role labeling and predicate-argument structure analysis.

1. FrameNet:

FrameNet is a lexical database that groups verbs and nouns into semantic frames — conceptual structures representing a type of event or scenario.

It is based on the theory of frame semantics, which suggests that the meaning of a word can be understood in terms of the typical situations it describes.

* A frame is a situation (like buying, giving, travelling) with expected participants (called frame elements).

Key elements :-

Frames :- A frame is a type of situation or scenario. Each frame involves certain participants, which are called frame elements.

Frame elements :-
These are the roles played by the different participants in a frame.

Working :-

1. Identify Frames :-
Researchers identify common situations (frames).

2. Assign Frame Elements :-
Each frame has specific roles.

3. Label sentences :
sentences are tagged with these frames and frame elements to show how words are used in context.

Ex :- "Ravi brought a car from Arjun"

Frame : Commerce_buy

Frame Elements :
- Buyer : Ravi
- Goods : a car
- Seller : Arjun

o/p :-
Commerce_buy (Buyer = Ravi, Goods = car, Seller = Arjun)

2. PropBank

- PropBank is a corpus of text where each verb is annotated with its arguments, giving us a clear idea of who is doing what to whom in sentence.
- This helps in understanding the roles of different entites in relation to the verb.

PropBank stands for proposition Bank.

- It helps NLP systems understand the roles of words in sentence.

Key elements :-

predicate : Usually a verb, it represents an action or state.

Arguments :- The paricipants involved in the action or state described by the predicate. Arguments are categorized as core or adjunctive.

Goal of propBank

To provide:-

- A consistent, corpus-based labeling of predicate-argument structures.
- Numbered roles (ARG0 to ARG5) for different verbs.
- A foundation for training semantic Role labeling (SRL) systems.

How PropBank works

1. Each verb is annotated with:
   - Its sense (meanings).
   - Its arguments (entities participating in the event)
2. Roles are numbered as ARG0, ARG1, ARG2, etc., based on the verb's meaning.
3. Modifiers (like time, location) are labeled as ARGM-TMP, ARGM-LOC etc.

- common PropBank Argument Roles

Role	General meaning
ARG 0	Agent / subject / Doer
ARG 1	Theme / patient / thing acted upon
ARG 2	Benficiary / Instrument
ARGM-TMP	Temporal Modifier (Time)
ARGM-LOC	Location
ARGM-MNR	Manner
ARGM-CAU	Cause.

ex:- "Sita opened the door with a key".

predicate : opened

Arguments (based on propBank frame for "open")

- ARG0 = sita → Agent (person doing the action)
- ARG1 = the door → Theme (thing being opened)
- ARG2 = a key → Instrument (tool used).
- ARGM-MNR = with a key (can be modifier too)

Representation:-

open.01
ARG0 : sita
ARG1 : the door
ARG2 : the key

Both are used to help NLP systems understand sentence meaning for applications like question answering, summerization and machine translation.

Meaning Representation Systems (MRS)

- Meaning representation is a deeper level of Semantic interpretation aimed at converting natural language into format that machines can understand and act on.
- This process is similar to how programming languages are compiled into machine code that computers execute.

- Unlike artificial languages, natural language is flexible and relies on context and general world knowledge for understanding, which poses a challenge for machines.
- Researchers have been working for decades to develop methods to interpret and encode this context and knowledge for machines.
- However, current techniques are limited to specific domains and problems and do not scale well to arbitrary domains

A meaning Representation system is a way to formally represent the meaning of a natural language sentence so that a computer can:
- Understand it
- store it
- Reason with it
- Act on it

Think of it as converting language into a mathematical or logical format.

Goals:-
1. unambiguity: No confusion in interpretation
2. Compositionality: Meaning is built from parts
3. Executability: Machines can use it to make decisions
4. Expressiveness:- can express complex ideas & relationships

Key components:-

component	Description
Entites	people, objects, place mentioned in the sentence.
Events/predicates	Actions or states involving entites (e.g. "eat")
Arguments	participants in the event (subject, object etc).
Relations	Connections b/w entities (e.g. ownership, location)

Popular Meaning Representation systems:

1. First order predicate logic (FOL)
- Based on FOL
- Represents facts and relationships using predicates, constants, variables, quantifiers.

Ex:- "Every student studies"
→ ∀x(student(x) → studies(x))

sentence: "Ravi loves Priya"
→ Loves(RAVI, Priya).

2. Frame-Based
- uses a slot-filler structure to represent meaning
- Good for representing events and roles in context.

3. Semantic Networks
- Meaning is stored as a graph of nodes and links
- Nodes = concept Identities
- Edges = relations.

Resources:-

1. ATIS (Air Travel Information system)

• Domain : Flight booking / airline travel queries.

: Task :

Convert natural language into a database query.

Ex:-

"Show me flights from Boston to Denver tomorrow"

o/p :-

Flight (from = Boston, to = Denver, date = tomorrow)

2. Communicator

• Domain :- Multi-modal travel planning & ticketing

• Use case :- Dialogue system research.

- Developed by DARPA for evaluating spoken dialogue systems.

Focus:-

• Robustness of dialogue understanding

• Content tracking, dialogue flow, meaning representation.

3. Geo Query :-

- Us. geography database.

- Task :-

Convert user questions into Prolog-style logical forms or SQL.

- Example Query

- Input :- "what is the capital of Texas"?

- o/p :- answer (capital (texas))

4. RoboCup :

It is an international competition where teams of robots play soccer, and its organized by the artificial intelligence community. The goal is to advance AI and robotics research through this challenging and fun domain.

Ex:-

Input :- 'Move the defenders back and focus on the wings'.

o/p :- Action logic or planning instructions.

These datasets help evaluate the accuracy, generalization, and compositionality of semantic parsers.

FrameNet vs PropBank

Feature	FrameNet	PropBank
Purpose	Models situations (frames) and their participants.	Labels verb arguments with general semantic roles.
Annotation Type	frame-based annotation	predicate-argument annotation.
Focus	Meaning from the frame or scenario	meaning from the verb and its argument
Role labels	Descriptive labels (e.g Buyer, seller, Item)	Generic numbered roles (ARG0, ARG1, ARG2)
verb sense coverage	More thematic & lexical richness, often deeper	Focused on broad coverage and consistency.
Ex:-	"Ravi brought a car from Arjun"	"Ravi brought a car from Arjun"
	Frame: commerce_buy Buyer: Ravi Goods: car seller: Arjun	verb: buy.01 ARG0: Ravi (buyer) ARG1: car (item) ARG2:
Domain adapility	Rich for linguistic theory	easy to use in machine learning models
used In	FrameNet-based, theoretical linguistics	semantic Role
used In	FrameNet-based parsers	Larger corpus (PropBank + Penn Treebank)
Data size	smaller (few thousand frames), high manual effort	

key differences

- FrameNet :- Scenrio-based, uses descriptive roles
- PropBank = verb-based, uses numbered roles (ARG0, ARG1)

Unit-V

Language Modeling:

- Introduction
- N-Gram Models
- Language model Evaluation
- Baysian parameter estimation
- ~~Language Model Adaptation~~,
- Language model Adaption
- Language models
  - class based
  - variable length
  - Bayesian topic based
- multilingual & cross lingual language modeling.

Language modeling

Introduction:-

language models are crucial for various applications in human language technology. This help predict the likelihood of word sequences based on training data.

A language model is a statistical tool used to assign a probability to a sequence of words. It helps in predicting the next word, evaluating sentence fluency, and guiding decisions in NLP systems such as machine translation, speech recognition and chatbots.

Why DO we need:-

language is ambiguous and unstructured, so NLP systems need a way to:

- predict the next word in a system.
- scores sentences based on how likely they are
- Help in choosing correct words in translation, correction or suggestion.

eg:- sentence:- "The cat sat on the ___ "

- mat → 0.6
- roof → 0.2
- car → 0.1

most likely next word: mat.

Applications

- speech recognization
- machine translation
- Chatbots
- spelling correction
- Auto complete.

common language model:

- N-Gram model
- Neural language models
- class based models
- variable-length models
- Bayesian Topic model
- Multilingual model.

Before discussing this models, lets know what is the traditional or old method used to predict the word. That is chain rule.

Chain rule :-

The chain rule in probability is used to compute the joint probability of a sequence of words by breaking it into conditional probabilites like:

$$P(w_1, w_2, w_3) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2)$$

(or)

$$P(w_1, w_2 \ldots w_n) = P(w_1) \times P(w_2|w_1) \times \ldots : P(w_n|w_1 \ldots w_{n-1})$$

But computing this full history is impractical. Bcoz it allows us to compute the probability of an entire sentence word by word.

limitations :- computational complexity, data sparsity, need for large storage.

→ To over come this some model like markov assumption are used. and N-Gram.

Markov Assumption

The probability of a word depends only on a limited no. of previous words, not the entire history.

example :-

Instead of computing

$P(w_4 | w_1, w_2, w_3)$

we assume

$P(w_4 | w_3)$ → first order markov assumption.

-Bigram model

$$P(w_1, w_1 \ldots w_n) = P(w_n | w_{n-1})$$

-trigram model

$$P(w_1, w_2 \ldots w_n) \approx P(w_n | w_{n-2}, w_{n-1})$$

It limits the words dependency to the last k words simplify computation in language models

N-Gram model

An N-Gram model is a statistical language model that predicts the next word in a sentence using the previous (N-1) words.

-It applies the markov assumption

A word depends only on a deplimited no. of previous words.

-An N-Gram is a sequence of N words

An N-Gram model uses these sequences to estimate the probability of word sequence.

It estimates

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

using $P(w_1, w_2 \ldots w_n) = P(w_n | w_{n-1}, \ldots w_{n+k+1}) =$ N Gram approximation

Depending on the length on n, we distinguish blw unigrams.

Unigrams (n=1) : The probability of each word is independent of any previous word.

$$P(w_1, w_2 \dots w_n) = P(w_n)$$

Bigram (n=2) = The probability of each word depends on previous word

$$P(w_1, w_2 \dots w_n) = P(w_n \mid w_{n-1})$$

Trigram (n=3) : uses two previous words

$$P(w_1, w_2 \dots w_n) = P(w_n \mid w_{n-2}, w_{n-1})$$

Ex:- "I want to eat pizza"

- Bigram (n=2) $P(I) \times P(want \mid I) \times P(eat \mid to) \cdot P(pizza \mid eat)$
- Trigram (n=3)

$P(I) \cdot P(want \mid I) \cdot P(to \mid I, want) \cdot P(eat \mid want, to)$

$P(pizza \mid to, eat)$

- The model uses frequency counts from a corpus to estimate each probability

Advantages :- Simple & fast, efficient for short context, Good Baseline, Real-time friendly

Limitations :-
- Data sparsity :- Many word combinations never appear in training data.
- Fixed context size
- large storage for Higher N
- Needs ~~something~~ smoothing requires techniques like laplace, backoff etc

An n-gram model is also known as (n-1)-th order markov model bcoz it only considers the n-1 previous words to predict the next one.

The markov assumption is the idea that a words depends only on a few previous words. The N-Gram model is practical application of this assumption using the last (N-1) words to predict the next word.

LANGUAGE MODEL EVALUATION

It is the process of measuring how well a language model performs, especially in predicting word sequence correctly.

To judge the performance of the language model i.e how sucessfull it is at estimating the word probabilites we use language model evaluation.

1. Perplexity

Perplexity tells how many word choices the model is "confused between" at each step.

It is a measure of how well a language model predicts a sequence of words.

→ Lower perplexity = better performance

formula :-

$$\text{perplexity} = 2^{-1/N \sum_{i=1}^{N} \log_2 P(w_i \mid \text{context})}$$

- If the model has no predictive power at all, perplexity is equal to vocabulary size.
- A model achieving perfect prediction, by contrast, has a perplexity of one.

2. Coverage Rate

- Coverage rate refers to the percentages of words in the test set that were seen during training.

high coverage = better vocabulary match between training and test data

If the model knows all the words, it has high coverage.

- It tells the percentage of n-grams present in the test set.
- A special case is the out-of-vocabulary rate (OOV rate), which is the percentage of unique words in the test set that the model does not know.

Formula: $\text{coverage rate} = \frac{\text{No. of known (seen) words}}{\text{Total words in test set}} \times 100\%$

e.g.:- If the test set has 1000 words, and 920 of them were seen during training

$\text{coverage Rate} = \frac{920}{1000} \times 100 = 92\%$

3. Cross validation:-

A method where data split into k-parts to test the model multiple times for more reliable evaluation.

4. Split validation (Train-Test split)

Split validation means dividing data into 2 fixed parts

- Training set :- to train the model
- Testing : to evaluate the model

In the ratio's 80/20 → 80% training, 20% testing

or 70/30 or 60/40

Parameter Estimation

I what are parameters?

In NLP models, parameters are values that the model uses to make decisions.

- For example, in a language model, parameters might include probabilites that represent how likely one word is to follow another word.

parameter estimation:-

It is process of determining the best values for the parameters of a model so that it can accurately perform tasks like

- understanding, generating, tranlating text

parameter estimation is the process of calculating the probability values (parameters) that a language model uses to predict words.

In simple terms

It means learning how likely each word or word sequence is, based on the training data.

Importance :-

- The accuracy and effectiveness of an NLP model depend on how well its parameter are set.
- Good parameter estimation ensures that the model can understand & generate human-like text accurately.

There are 2 techniques

1. maximum likelihood estimation
2. Bayesian estimation.

Maximum likelihood estimation

MLE is about finding the parameter values that make the observed data most likely.

- In the context of NLP, this means finding the probabilities of words or sequences of words that best explain the text data you have.

How it works:-

1. Gather a large amount of text data
2. Count how often different words or sequences of words appear in the data.
3. Use these counts to estimate the probabilities of words or sequences. For example, the probability of a word w given a previous word w_{prev} is estimated as:

$$P(w \mid w_{prev}) = \frac{\text{Count}(w_{prev}, w)}{\text{Count}(w_{prev})}$$

Here, $\text{Count}(w_{prev}, w)$ is the no. of times w follows w_{prev}, and $\text{Count}(w_{prev})$ is the no. of times w_{prev} appears.

Ex:- "I like pizza. I like coding"

Count("I like") = 2
Count("like pizza") = 1
Count("like coding") = 1
Count("I") = 2
Count("like") = 2

Now

$P(\text{like} \mid \text{I}) = \frac{2}{2} = 1.0$ $P(\text{pizza} \mid \text{like}) = \frac{1}{2} = 0.5$

$P(\text{coding} \mid \text{like}) = \frac{1}{2} = 0.5$

These probabilities are estimated from word counts - this is parameter estimation.

Bayesian Estimation :-

Bayesian parameter estimation uses Baye's theorem to calculate the probability of a parameter based on both prior knowledge and observed data.

It combines

- prior belif (before seeing data)
- likelihood (from the data)
- To produce a posterior probability (updated belif after seeing data).

How it works:-

1. Start with a prior distribution that represents your initial beliefs about the parameters
2. Gather your text data.
3. Use Bayes Theorem to update the prior distribution with the observed data to get the posterior distribution. This posterior distribution represents your updated belifs after seeing the data.

Bayes theorem :-

$$P(\theta \mid D) = \frac{P(D \mid \theta) \cdot P(\theta)}{P(D)}$$

where, θ - the parameter to estimate (like a probability)
D - observed data
$P(\theta)$ - prior probability
$P(D|\theta)$ - likelihood
$P(\theta|D)$ - posterior (final estimated value).

ex:- we are estimating $P(\text{pizza} | \text{like})$

This is the conditional probability:
- what is the probability of the word "pizza" appearing after the word "like"?

Step 1:- word counts from traing data.
lets assume your corpus has only these 2 bigrams
- like pizza → occured 1 time
- like coding → occured 1 time.

that gives us:-

Bigram	count
like pizza	1
like coding	1
like dogs	0
like rain	0

so, total times "like" was followed by something
$= 1 (\text{pizza}) + 1 (\text{coding}) = 2$.

Step 2:- Vocabulary size (V)
Assume the vocabulary contains 4 possible words that could follow "like"
{pizza, coding, dogs, rain}
so $\boxed{V = 4}$

Step 3:- Apply Laplace smoothing (Bayesian estimation)
we use the Dirichlet prior ($\alpha = 1$), which adds 1 pseudo-count to every word.

Step 4:- use the formula
Bayesian smoothed probability

$$P(w|\text{like}) = \frac{\text{count}(\text{like } w) + \alpha}{\text{count}(\text{like}) + \alpha \cdot V}$$

plug in values:-
for pizza: $P(\text{pizza}|\text{like}) = \frac{1+1}{2+1 \cdot 4} = \frac{2}{6} \approx 0.33$

for coding $P(\text{coding}|\text{like}) = \frac{1+1}{2+4} = \frac{2}{6} \approx 0.33$

for dogs $P(\text{dogs}|\text{like}) = \frac{0+1}{2+4} = \frac{1}{6} \approx 0.166$

for rain $P(\text{rain}|\text{like}) = \frac{0+1}{2+4} = \frac{1}{6} \approx 0.166$

each word is assigned a probability using both real data (actual counts) and prior belief ($\alpha = 1$). This prevents assigning zero probability to unseen words & makes the model more robust.

Language model Adaption

It is the process of modifying a trained language model so that it works better for a new domain, topic, language or user.

In simple terms:-
we "adapt" a general model (trained on large data) to make it work well on specific or smaller datasets.

This is important when the data used to train the model is different from the data when the model will actually be used.

Why adaption is Needed:-

- General models are trained on broad corpora (e.g news, wikipedia)
- But in real world use, we need to perform well on:-
  - specific domains (medical, legal, tech)
  - specif users (custom vocabulary or style)
  - specific languages (dialects, regional use).

Techniques:

1. fine Tuning :-
   You take a general language model and train it finely using data from a specific domain.
   - Continue training the model on small, target-domain data
   - common in neural language model
   e.g:- a genral GPT-like model fine-tuned on medical records.

2. Model Interpolation
   - Also called mixture models
   - combines two models
     - A general model
     - A domain-specific model.

   Helps use both broad and focused knowledge.

3. Topic-Dependent model
   These models are trained or adapted for specific topics or domains (e.g sports, medice etc)

to improve accuracy & relevance.

e.g:- In a news app, you might have
- A model adapted to sports language

If user is reading sports news, the system activates the sports-specific language model.

4. Self adaptive models.
   - Also called as trigger models
   - This technique adapts the model dynamically based on the words that have already been seen in the text.

   e.g:- In a financial document, seeing the word "stock" increases the likehood of subsequently seeing "market" "price" or "exchang".

5. Unsupervised Adaption
   Adaption done without labeled data, using raw unannotated text forom the target domain

   e.g = suppose you want to adapt your model for legal documents, but you don't have labeled examples.

Advantages :-

1. Improves Accuracy in specific Domain
2. Reduce errors on specialized terms
3. Better User Experience
4. makes use of target Data
5. Dynamic learning

Language models.

It is a system that assigns probabilities to sequences of words and predicts the next word based on the previous words.

Class Based

A class-Based language model groups into semantic or syntatic classes (like nouns, cities, verbs) and predicts words based on their class, instead of prediciting individual words directly.

Advantages :-

- Reduces data sparsity
- Generalizes better across similar words
- Improves accuracy in low-resource settings

limitations :-

- Requires a good word-to-class mapping
- might losse individual word-level nuances.

Discuss with example class based variable length model.

class based :-

class based language models are way of simplifying language processing by grouping words that have similar meanings or functions into categories, called "classes".

- Instead of treating each word separately the model works with separately, the model works with these groups of words to make predictions about language.

Ex:- let's say we have a sentence:

"The favourite pet is a ___".

class prediction: The model might predict the next word belongs to the "animals" class because "pet" commonly appear in this context.

word :- word selection :-

within the "Animals" class, it might choose the word "dog", resulting in sentence

"The favourite pet is dog".

How it works :-

- words are grouped into classes based on their roles or meaning
- calculates the class probability
- calculates the word probability within the class.

Applications :-
- low resource language modeling
- speech recognition - class based machine translati

Variable length language model
It uses content of different lengths dynamically, instead of fixed-size n-grams.

It is also called
- variable-order n-gram model
- cache-based or Trie based model.

How it works :-
- Instead of fixed history, it uses a much useful history as available
- Adapts to longer or shorter content dynamically

Ex:- "she ate a cake with a fork"
A fixed trigram :-
P(cake | a, ate), P(with | cake, a)
variable-length model :-
P(fork | a cake with a) → longer context gives better prediction.
* improves content sensitivity & prediction accuracy.

Applications :-
- Neural language models
- speech recognition with large vocabularies
- content-sensitive predective text.

variable length model

A variable length language model uses different amounts of content (not a fixed N-1 window like N-grams) to predict the next word.

Advantages:-

- captures long-term dependencies better than fixed N-grams.
- More flexible in real-world data
- Reduces zero probabilites from unseen long sequences

limitations:-

- can be computationally heavy
- Requires more memory to store variable-length histories.

Bayesian Topic Based model

A Bayesian topic based model is a probabilistic language model uses 'Bayes Theorem' to discover hidden topics in a collection of documents, and then uses these topics to model and generate language more accurately.

most common model: LATent Dirichlet Allocation (LDA)

LDA is popular unsupervised Bayesian model for

- Topic modeling
- Document classification
- Language modeling.

key concepts :-

- Topic : A distribution over words
  (e.g. Topic 1 : {game, team, score} Topic 2 : {doctor, patient})
- Document : A mixture of topics
  (e.g. A sport article 80% Topic 1, 20% Topic 2)
- Bayesian Inference :-
  Used to compute the posterior probability of topics given the words in document.

eg :- we have 3 documents

1. The doctor examined the patient
2. The team won the game
3. The hospital hired a new doctor

LDA might find :-

- Topic A (medical) : doctor, patient, hospital, medicine
- Topic B (sports) : team, game, win, score.

Then,

- DOC 1 = 90% Topic A, 10% Topic B
- DOC 2 = 95% Topic B, 5% Topic A
- DOC 3 = 70% Topic A, 30% Topic B.

Advantages :-

- captures hidden topics
- Improves predictions
- works with unlabled data
- Generalizes across ~~data~~ documents

limitations :-

- Needs long document
- Requires hyperparameter tuning
- Computationally expesive.

Multilingual and cross-lingual language modeling

Multilingual language modeling involves training models on multiple languages so they can understand, generate, or translate across different languages.

Cross-lingual language modeling focuses on transfering knowledge from one language (usually a high-resource language like English) to help tasks in another (often low-resource) language.

Why Its important :-

- many languages lack enough data
- cross and multilingual LM allow models to
  - share knowledge across languages
  - perform translation, sentiment analysis or Q&A in multiple languages.
  - understand context even in code-mixed or bilingual data.

Feature	multilingual	cross lingual
Def	Trained on multiple languages at once	Trained in one language used to perform task in other.
Goal	Understand / generate text in many languages	transfer knowledge from one language to another
e.g Task	One model doing translation or Q&A in English, Hindi, Tamil	Train on English Q&A apply to Hindi Q&A
Training Data	Requires multi-language datasets	May use only one language's data during training.
Use case	Multilingual chatbots translation systems	low-resource language support using high resource training.
Model examples	mBERT, MT5, XLM-R	XLM, LASER, cross-lingual BERT.

example :-

multilingual :-

mBERT trained on English + Hindi + Tamil → can understand all

Cross lingual :-

Train a model on English news → use it to classify news in Hindi without training in Hindi.

You train a QA system on English data. Then you ask it a question in Hindi:

"भारत की राजधानी क्या है?"

A cross-lingual model can still answer "नई दिल्ली", by transferring knowledge from English training.

Advantages :-

- low resource support
- transfer learning
- multilingual NLP tasks
- Handles code-mixing.

Limitations :-

- Bias toward high resource languages
- vocabulary mismatch
- Expensive training.