MODULE 2
SEMANTIC ANALYSIS

What is Semantic Analysis

Semantic analysis refers to a process of understanding natural language (text) by extracting insightful information such as context, emotions, and sentiments from unstructured data.

It gives computers and systems the ability to understand, interpret, and derive meanings from sentences, paragraphs, reports, registers, files, or any document

Semantic analysis is the process of finding the meaning from text.

Therefore, the goal of semantic analysis is to draw exact meaning or dictionary meaning from the text.

The semantic analysis process begins by studying and analyzing the dictionary definitions and meanings of individual words also referred to as lexical semantics.

Difference Between Syntax and Semantics

- **Syntax:**
- It refers to the rules and regulations for writing any statement in a programming language.
- It does not have to do anything with the meaning of the statement.
- A statement is syntactically valid if it follows all the rules.
- It is related to the grammar and structure of the language.
- Example:
  x = false

- **Semantics:**
- It refers to the meaning associated with the statement in a programming language.
- It is all about the meaning of the statement which interprets the program easily.
- Errors are handled at runtime.
- Example:
  x * y = z

Advantages of semantic analysis

1.Gaining customer insights : Semantic analysis helps in processing customer queries and understanding their meaning, thereby allowing an organization to understand the customer's inclination. Moreover, analyzing customer reviews, feedback, or satisfaction surveys helps understand the overall customer experience by factoring in language tone, emotions, and even sentiments.

2.Boosting company performance: Automated semantic analysis allows customer service teams to focus on complex customer inquiries that require human intervention and understanding. Also, machines can analyze the messages received on social media platforms, chatbots, and emails. This improves the overall productivity of the employees as the tech frees them from mundane tasks and allows them to concentrate on critical inquiries or operations.

Advantages of semantic analysis

3. Fine-tuning SEO strategy: Semantic analysis helps fine-tune the **search engine optimization (SEO) strategy** by allowing companies to analyze and decode users' searches. For example, understanding users' **Google searches**. The approach helps deliver optimized and suitable content to the users, thereby boosting traffic and improving result relevance.

NOTES: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-semantic-analysis/#:~:text=Semantic%20analysis%20analyzes%20the%20grammatical,language%20processing%20(NLP)%20systems.

How Does Semantic Analysis Work?

The semantic analysis method begins with a language-independent step of analyzing the set of words in the text to understand their meanings.This step is termed '**lexical semantics**' and refers to fetching the dictionary definition for the words in the text.

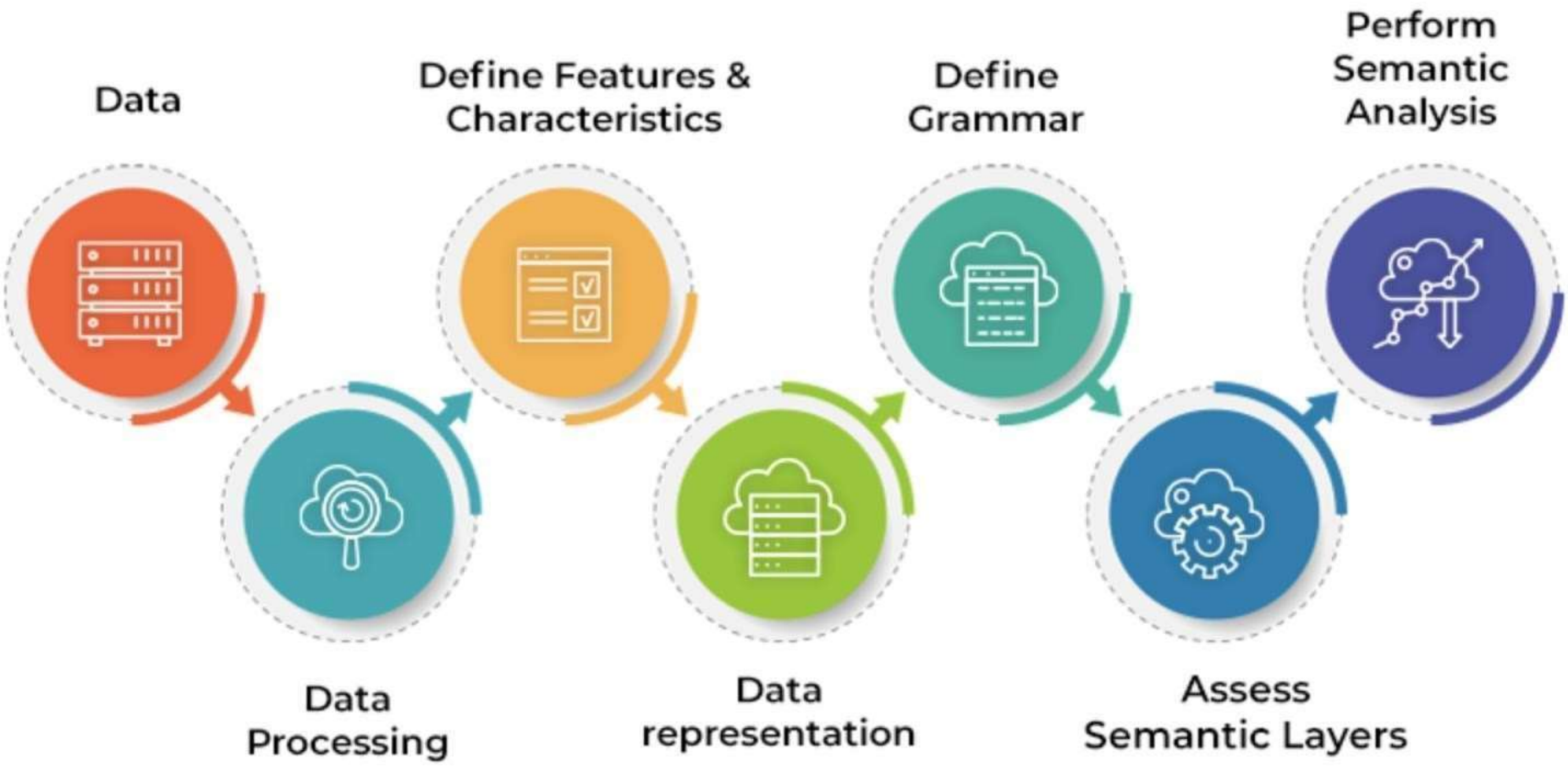
Each element is designated a grammatical role, and the whole structure is processed to cut down on any confusion caused by ambiguous words having multiple meanings.

Upon parsing, the analysis then proceeds to the interpretation step, which is critical for artificial intelligence algorithms.

For example, 'Blackberry is known for its sweet taste' may directly refer to the fruit, but 'I got a blackberry' may refer to a fruit or a Blackberry product. As such, context is vital in semantic analysis and requires additional information to assign a correct meaning to the whole sentence or language.

Technically, semantic analysis involves:

1. Data processing.
2. Defining features, parameters, and characteristics of processed data
3. Data representation
4. Defining grammar for data analysis
5. Assessing semantic layers of processed data
6. Performing semantic analysis based on the linguistic formalism


Data
Data Processing
Define Features & Characteristics
Data representation
Define Grammar
Assess Semantic Layers
Perform Semantic Analysis

Representing Meaning

Representing meaning in NLP involves converting natural language (words, sentences, or text) into a structured form that a machine can process and understand. The goal is to capture the semantics of language—what words and sentences mean—and enable computers to reason about and manipulate this meaning. This is crucial for tasks like:

- *Machine Translation:* To accurately translate sentences, systems need to understand and preserve the meaning of the source language in the target language.
- *Question-Answering:* Systems need to represent meaning to map user questions to appropriate answers based on the context and relationships between entities.
- *Information Retrieval:* Search engines retrieve relevant documents by understanding the meaning of the user query, not just matching keywords.

Challenges in Representing Meaning in NLP

Representing meaning in natural language processing (NLP) is a complex task due to the inherent richness, ambiguity, and variability of human language. Here are some of the key challenges:

a) Ambiguity

•Lexical Ambiguity: Words often have multiple meanings depending on the context. For example, the word "bank" can mean a financial institution or the side of a river.
Example: "**bat**" (Flying mammal or cricket bat?)

•Syntactic Ambiguity: Sentences can have multiple valid interpretations due to their structure.
Example: "**The chicken is ready to eat**." (Is the chicken itself eating, or is it being served as food?)

•Semantic Ambiguity: Ambiguity can also arise at the semantic level, where entire sentences or phrases have multiple interpretations.
Example: "**John gave a bath to his dog wearing a blue shirt**." (This sentence is semantically ambiguous because it's unclear who is wearing the blue shirt—John or the dog.)

In summary:

•Lexical ambiguity deals with word meaning.
•Syntactic ambiguity deals with sentence structure.
•Semantic ambiguity deals with sentence meaning and the possible different interpretations of a whole phrase or sentence.

b) **Context Dependency**

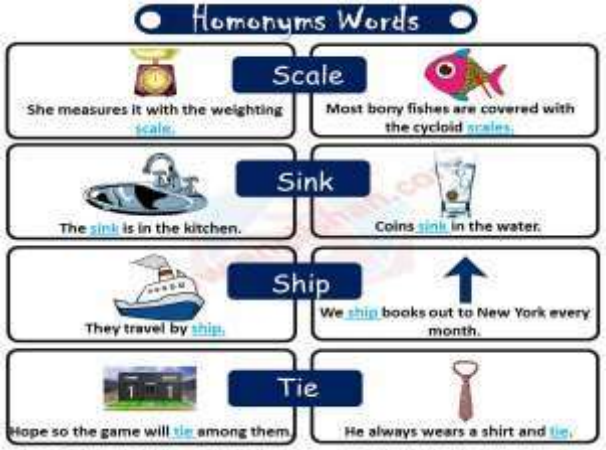
• **Word Meaning Varies by Context:** Words and phrases often change meaning depending on the surrounding words, sentence structure, or broader context.
Example: The word "light" can refer to brightness (light bulb) or weight (light bag).

•**Pragmatics and World Knowledge:** Meaning is often influenced by real-world knowledge and the context in which language is used. Machines struggle with understanding implicit meaning that humans infer naturally.
Example: "Can you pass the salt?" (The literal meaning is a question about capability, but the pragmatic meaning is a request to hand over the salt.)

c) **Polysemy and Homonymy**

• Polysemy: Words can have multiple related meanings.
Example: The word "run" can mean to run a race, run a business, or run software.

• Homonymy: Words can have multiple unrelated meanings.
Example: "Bark" can refer to a dog's sound or the outer covering of a tree.



d) Idiomatic Expressions and Metaphors

• Idioms: Fixed expressions where the meaning is not derived from the literal interpretation of the individual words.
Example: “Break the Ice" It doesn't mean physically breaking the ice.

• Metaphors: Phrases used to imply one concept by referencing another.
Example: "He has a heart of stone." (This doesn’t mean his heart is literally made of stone, but that he is unfeeling.)

e) Compositionality

•**Complex Sentences:** Meaning in complex sentences is derived from the individual meanings of words and the rules governing their combination (compositional semantics). However, this is not always straightforward.

Example: "The man saw the boy with the telescope." (Did the man see the boy using a telescope, or did the boy have the telescope?)

•**Non-Compositional Phrases:** Sometimes, the meaning of a phrase cannot be determined from the meanings of individual words.
Example: Phrases like "spill the beans" cannot be understood by analyzing the individual words alone

f) Cultural and Temporal Variability

•Cultural Differences: The meaning of certain expressions or words may vary between cultures.
Example: The phrase "knock on wood" is a sign of luck in some cultures but might be meaningless or interpreted differently in others.

• Temporal Changes: Language evolves over time, and meanings of words or phrases change.
Example: The word “Cool" historically referred to temperature, but in modern usage, it refers to trendy, or socially appealing.

g) Word Sense Disambiguation (WSD)

•Multiple Senses of a Word: Determining the correct meaning of a word in a given context is a key challenge in NLP.
Example: The word "apple" can refer to the fruit or the technology company depending on context. WSD requires understanding which meaning is intended.

h) **Handling Synonyms and Antonyms**

• Synonyms: Different words can have the same or very similar meanings.
Example: "Big" and "large" mean almost the same thing, but they may have slightly different connotations.
•Antonyms: Words that mean the opposite need to be correctly identified in context.
Example: "Happy" and "sad" are Antony

i) **Reference and Coreference Resolution**

• Coreference: Identifying when two expressions refer to the same entity.
Example: "John took his car to the shop. He was worried it wouldn’t start." (Resolving "he" as John and "it" as the car.)
•Anaphora and Cataphora: Referring expressions that point backward (anaphora) or forward (cataphora) in text can complicate meaning representation.
Example: "If you see Mary, tell her I miss her." (Understanding "her" refers to Mary.)

j) **Granularity of Meaning Representation**

•Fine-Grained vs. Coarse-Grained Representation: Deciding the level of detail in representing meaning can be challenging. Some tasks require a deep understanding of nuanced meanings, while others may only need a general sense.

Example of Fine-Grained Representation: In a sentiment analysis task that distinguishes between nuanced emotional tones, a fine-grained representation might differentiate between "happy," "elated," and "content." This level of detail helps the model understand subtle emotional differences in user feedback, allowing for more tailored responses or analyses.

Example of Coarse-Grained Representation: In a document classification task where the goal is to categorize articles into broad topics like "Sports," "Politics," or "Health," a coarse-grained approach suffices. Here, distinguishing between different sentiments or specific emotions is less critical than correctly identifying the overall category of the document.

11. **Scalability and Efficiency**

•Complexity of Representation: Building detailed and accurate meaning representations can be computationally expensive, particularly for large texts and corpora

• Real-Time Processing: Systems need to process meaning representations efficiently in real-time applications like chatbots or search engines

Several approaches used to represent meaning in NLP (Representing Linguistically Relevant Concepts)

i) Logical Form (Predicate Logic)

Predicate Logic is a formal system for representing meaning using predicates and arguments. It represents the relationships between entities and allows for reasoning based on these relationships.

- Predicates: Represent actions or properties.
- Arguments: Represent the entities involved in those actions.

Example: • Sentence: "The cat chased the mouse."
• In Predicate Logic: Chases(Cat, Mouse)

Predicate logic can handle complex logical structures with quantifiers like "all" (∀) and "some" (∃), which allows for more sophisticated reasoning.

ii) Semantic Networks

Semantic Networks represent knowledge in a graph structure where:
• Nodes are concepts or entities.
• Edges are relationships between these entities.
These networks are used to represent hierarchical knowledge, such as taxonomies or conceptual relationships like is-a, part-of, and causes.
Example: For the sentence: "A dog is an animal," a semantic network might look like: • Dog → is-a → Animal
This structure helps with reasoning tasks and knowledge extraction in NLP systems

iii) Frames and Scripts

Frames are data structures used to represent stereotypical situations. Each frame contains a set of slots that hold information about the participants, actions, and objects in a scenario. Scripts extend frames to represent sequences of actions in a specific situation.
Example: For the sentence: "John booked a flight," the frame for this action might include:
• Agent: John
• Action: Booking
• Object: Flight
Frames are useful for understanding events and contexts, and are commonly used in dialog systems or information extraction

iv) Distributional Semantics

Distributional Semantics is based on the idea that the meaning of a word can be understood from the context in which it occurs. Words that appear in similar contexts tend to have similar meanings. This approach is often implemented using word embeddings (like Word2Vec, GloVe, and BERT) that represent words as vectors in a high-dimensional space. Similar words are close to each other in this space.

Example: In a distributional semantic model, words like cat and dog will be closer to each other than cat and car, since cat and dog often occur in similar contexts.

Semantic roles and their functions

Semantic roles (also known as thematic roles or theta roles) describe the relationship between a verb and the entities involved in the action or event that the verb expresses. In essence, they explain "who did what to whom," capturing the function of each participant in a sentence. Understanding semantic roles is critical for interpreting sentence meaning because they go beyond just syntactic structure, focusing on the roles participants play in the action described by the verb.

Key Semantic Roles

1. **Agent** • The Agent is the entity that performs or initiates the action. It is usually the subject of a sentence.
• Example: In "The dog chased the cat," the dog is the Agent because it is performing the action of chasing.

2. **Patient** (Theme) • The Patient (also called the Theme) is the entity that undergoes the action or is affected by it.
• Example: In "The dog chased the cat," the cat is the Patient because it is the entity being chased.
In "John painted the house," the house is the Patient because it undergoes the action of being painted.

3. **Experiencer** • The Experiencer is the entity that feels or perceives something but does not perform an action in the traditional sense.
•Example: In "Mary heard the music," Mary is the Experiencer because she perceives the sound but does not actively create it.
In "John fears spiders," John is the Experiencer since he experiences fear

4. **Instrument** • The Instrument is the entity used by the Agent to perform an action.
•Example: In "She cut the bread with a knife," the knife is the Instrument used by the Agent (she) to cut the bread.

5. **Beneficiary (Benefactive)** • The Beneficiary (or Benefactive) is the entity that benefits from or is the recipient of the action.
•Example: In "John baked a cake for Mary," Mary is the Beneficiary because she is the one who benefits from John's action of baking a cake.

6. **Goal** • The Goal is the endpoint or destination of an action.
•Example: In "They sent the package to New York," New York is the Goal, as it is the destination of the package.
In "The cat ran to the door," the door is the Goal because it is where the cat is heading.

7. **Source** • The Source is the entity from which something originates or is moved away.
•Example: In "The letter was sent from Paris," Paris is the Source because it is the origin of the letter.
In "She took the book from the shelf," the shelf is the Source from which the book was taken.

8. **Location** • The Location refers to the place where an action occurs.
• Example: In "The party was held at the park," the park is the Location because it is where the event took place.
In "The book is on the table," the table is the Location of the book.

9. **Recipient** • The Recipient is the entity that receives something as a result of the action.
•Example: In "John gave the book to Mary," Mary is the Recipient because she receives the book.

10. **Cause** • The Cause is the reason or force that brings about an action or event.
• Example: In "The flood destroyed the village," the flood is the Cause of the destruction.

Examples of Semantic Analysis

Uber's social listening

Uber uses semantic analysis to analyze users' satisfaction or dissatisfaction levels via social listening. This implies that whenever Uber releases an update or introduces new features via a new app version, the mobility service provider keeps track of social networks to understand user reviews and feelings on the latest app release.

Thus, as and when a new change is introduced on the **Uber app**, the semantic analysis algorithms start listening to social network feeds to understand whether users are happy about the update or if it needs further refinement.

2. Google's semantic algorithm – Hummingbird

Google incorporated 'semantic analysis' into its framework by developing its tool to understand and improve user searches. The Hummingbird algorithm was formed in 2013 and helps analyze user intentions as and when they use the google search engine. As a result of **Hummingbird**, results are shortlisted based on the 'semantic' relevance of the keywords. Moreover, it also plays a crucial role in offering **SEO benefits** to the company.

Examples of Semantic Analysis

3. Cdiscount's semantic analysis of customer reviews

Cdiscount, an online retailer of goods and services, uses semantic analysis to analyze and understand online customer reviews. When a user purchases an item on the ecommerce site, they can potentially give post-purchase feedback for their activity. This allows **Cdiscount** to focus on improving by studying consumer reviews and detecting their satisfaction or dissatisfaction with the company's products.

Cdiscount started using semantic analysis in 2019. It has immensely helped the company handle user queries associated with product deliveries and product returns.

Examples of Semantic Analysis

4. Uber's customer support platform to improve maps

Maps are essential to Uber's cab services of destination search, routing, and prediction of the estimated arrival time (ETA). All these services perform well when the app renders high-quality **maps**. Along with services, it also improves the overall experience of the riders and drivers.

All factors considered, Uber uses semantic analysis to analyze and address customer support tickets submitted by riders on the **Uber platform**. The analysis can segregate tickets based on their content, such as map data-related issues, and deliver them to the respective teams to handle. The platform allows Uber to streamline and optimize the map data triggering the ticket.

5. IBM's Watson conversation service

IBM's Watson provides a conversation service that uses semantic analysis (**natural language understanding**) and deep learning to derive meaning from unstructured data. It analyzes text to reveal the type of sentiment, emotion, data category, and the relation between words based on the semantic role of the keywords used in the text. According to IBM, semantic analysis has saved 50% of the company's time on the information gathering process.

First-Order logic

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- FOL is sufficiently expressive to represent the natural language statements in a concise way.
- First-order logic is also known as **Predicate logic or First-order predicate logic**. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.
- First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
  - **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
  - **Relations: It can be unary relation such as:** red, round, is adjacent, **or n-any relation such as:** the sister of, brother of, has color, comes between
  - **Function:** Father of, best friend, third inning of, end of, ...

- In the topic of Propositional logic, we have seen that how to represent statements using propositional logic.

  But unfortunately, in propositional logic, we can only represent the facts, which are either true or false.

- PL is not sufficient to represent the complex sentences or natural language statements.

  The propositional logic has very limited expressive power.

- Consider the following sentence, which we cannot represent using PL logic.
  - "Some humans are intelligent"
  - "All likes cricket."
- To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- FOL is sufficiently expressive to represent the natural language statements in a concise way.
- First-order logic is also known as Predicate logic or First-order predicate logic.
- First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.

- First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
  - Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
  - Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between
  - Function: Father of, best friend, third inning of, end of, ...
- As a natural language, first-order logic also has two main parts:
  - Syntax
  - Semantics

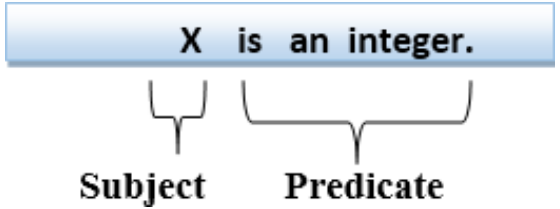
- Following are the basic elements of FOL syntax:
  - Constant 1, 2, A, John, Mumbai, cat,....
  - Variables x, y, z, a, b,....
  - Predicates Brother, Father, >,....
  - Function sqrt, LeftLegOf, ...
  - Connectives $\wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$
  - Equality ==
  - Quantifier $\forall, \exists$

- Atomic sentences:
  - Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
  - We can represent atomic sentences as Predicate (term1, term2, ..., term n).
  - Example: Ravi and Ajay are brothers: => Brothers(Ravi, Ajay).

    Chinky is a cat: => cat (Chinky).

- Complex Sentences:
  - Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
  - Subject: Subject is the main part of the statement.
  - Predicate: A predicate can be defined as a relation, which binds two atoms together in a statement.

- Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.



- A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:
  - Universal Quantifier, (for all, everyone, everything)
  - Existential quantifier, (for some, at least one).

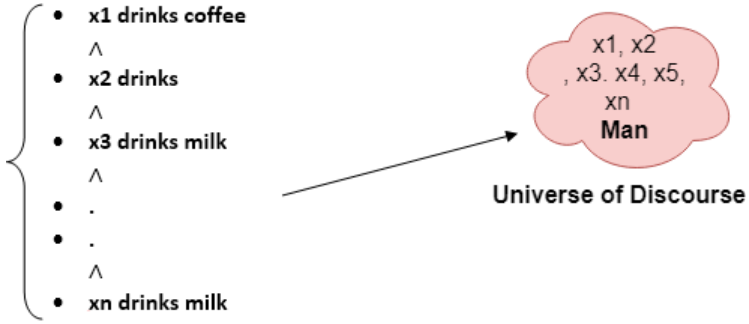
- Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.
- The Universal quantifier is represented by a symbol A, which resembles an inverted A.
- Note: In universal quantifier we use implication "→".
- If x is a variable, then Ax is read as:

  For all x

  For each x

  For every x.

- Example: All man drink coffee.
- Let a variable x which refers to a cat so all x can be represented in UOD as below:



So in shorthand notation, we can write it as :

$\forall x \, man(x) \rightarrow drink(x, coffee)$.

It will be read as: There are all x where x is a man who drink coffee.

- Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something.
- It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.
- Note: In Existential quantifier we always use AND or Conjunction symbol (∧).
- If x is a variable, then existential quantifier will be ∃x or ∃(x). And it will be read as:

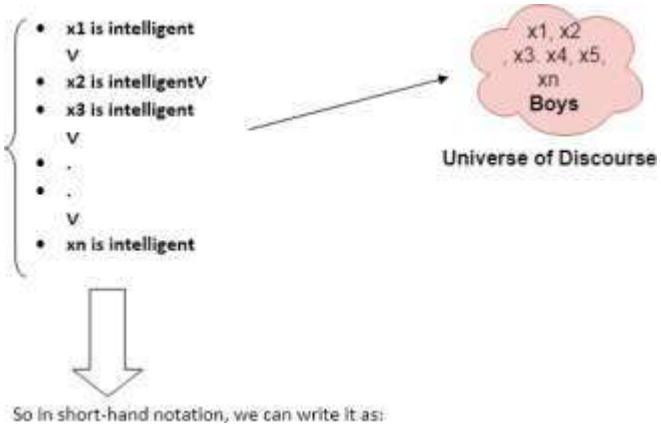
  There exists a 'x.'

  For some 'x.'

  For at least one 'x.'

- Example:

  Some boys are intelligent.



  $\exists x$: boys(x) $\wedge$ intelligent(x)

- It will be read as: There are some x where x is a boy who is intelligent.

- Points to remember:
  - The main connective for universal quantifier A is implication →.
  - The main connective for existential quantifier ∃ is and ∧.
- Properties of Quantifiers:
  - In universal quantifier, AxAy is similar to AyAx.
  - In Existential quantifier, ∃x∃y is similar to ∃y∃x.
  - ∃xAy is not similar to Ay∃x.

- 1. All birds fly.

  In this question the predicate is "fly(bird)."

  And since there are all birds who fly so it will be represented as follows.

  $$\forall x \; bird(x) \rightarrow fly(x).$$

- 2. Every man respects his parent.

  In this question, the predicate is "respect(x, y)," where x=man, and y= parent.

  Since there is every man so will use A, and it will be represented as follows:

  $$\forall x \; man(x) \rightarrow respects(x, parent).$$

- 3. Some boys play cricket.
  In this question, the predicate is "play(x, y)," where x= boys, and y= game. Since there are some boys so we will use ∃, and it will be represented as:

  $$\exists x \; boys(x) \rightarrow play(x, cricket).$$

- 4. Not all students like both Mathematics and Science.
  In this question, the predicate is "like(x, y)," where x= student, and y= subject.

  Since there are not all students, so we will use Awith negation, so following representation for this:

  $$\neg A\,(x)\,[\,student(x) \rightarrow like(x, Mathematics) \wedge like(x, Science)].$$

No	English Sentence	FOL Translation
1	All humans are mortal.	$\forall x\,(Human(x) \rightarrow Mortal(x))$
2	Some cats are black.	$\exists x\,(Cat(x) \wedge Black(x))$
3	Every student loves some subject.	$\forall x\,(Student(x) \rightarrow \exists y\,(Subject(y) \wedge Loves(x, y)))$
4	If it rains, the ground gets wet.	$Rain \rightarrow Wet(Ground)$
5	There exists a person who can solve every problem.	$\exists x\,(Person(x) \wedge \forall y\,(Problem(y) \rightarrow CanSolve(x, y)))$
6	Not all birds can fly.	$\neg \forall x\,(Bird(x) \rightarrow CanFly(x))$
7	Everyone who loves nature is happy.	$\forall x\,(Loves(x, Nature) \rightarrow Happy(x))$
8	Some dogs are friendly.	$\exists x\,(Dog(x) \wedge Friendly(x))$
9	All cars have wheels.	$\forall x\,(Car(x) \rightarrow HasWheels(x))$
10	If a person is rich, they are happy.	$\forall x\,(Person(x) \wedge Rich(x) \rightarrow Happy(x))$

No	English Sentence	FOL Translation
11	Some people are not honest.	∃x (Person(x) ∧ ¬Honest(x))
12	All men are mortal.	∀x (Man(x) → Mortal(x))
13	No animal can speak.	¬∃x (Animal(x) ∧ CanSpeak(x))
14	There is a book that every student reads.	∃x (Book(x) ∧ ∀y (Student(y) → Reads(y, x)))
15	All scientists are intelligent.	∀x (Scientist(x) → Intelligent(x))
16	Some teachers are strict.	∃x (Teacher(x) ∧ Strict(x))
17	If a machine is intelligent, it can learn.	∀x(Machine(x) ∧Intelligent(x) → CanLearn(x))
18	There is no one who knows everything.	¬∃x (Knows(x, Everything))
19	Some people are taller than others.	∃x ∃y (Person(x) ∧ Person(y) ∧ TallerThan(x, y))
20	If a student studies hard, they pass the exam.	∀x (Student(x) ∧StudiesHard(x) → PassesExam(x))

Syntax Driven Semantic Analysis

Syntax refers to the rules that govern the structure of sentences in a language. It is concerned with how words are combined to form meaningful phrases, clauses, and sentences. In NLP and linguistics, understanding the syntax of a language is crucial because the structure of a sentence often contributes to its meaning. In this context, syntax helps organize how words relate to each other and ensures that the sentence conveys the intended message.

Below are key aspects where syntax plays a vital role in determining meaning:

Word Order and Sentence Structure,
Grammatical Relations (Subject, Object, etc.), Ambiguity Resolution,
Scope of Modifiers and Quantifiers, Dependency Parsing,
Sentence Complexity and Subordination, Role of Punctuation and Intonation,
Tree Structures and Parse Trees, Meaning in Different Languages

a. Word Order and Sentence Structure: The way words are ordered in a sentence is critical to understanding meaning. Different languages have different word order conventions (e.g., Subject-Verb-Object or SVO in English).

- Example 1: "The cat chased the mouse."
  "The mouse chased the cat."

Both sentences contain the same words, but the order of the subject and object changes the meaning entirely. In the first sentence, the cat is the chaser, while in the second, the mouse is the one doing the chasing.

b. Grammatical Relations (Subject, Object, etc.): Syntax determines grammatical relations such as subject, object, and indirect object, which are key in understanding the meaning of an action or event. These relations indicate who is performing the action and who is affected by it.

- Example: "John gave Mary a book."
  - The subject (John) performs the action of giving.
  - The object (book) is what is given.
  - The indirect object (Mary) is the recipient of the book.

Rearranging the grammatical relations would completely change the meaning:

- "Mary gave John a book."
- Now, Mary is the one giving the book to John.

c. Ambiguity Resolution: Syntax helps in resolving ambiguity in sentences by providing a structured interpretation. The placement of words, phrases, and clauses can eliminate confusion or introduce multiple interpretations.

- Example: "I saw the man with the telescope."

This sentence can mean: 1. I used a telescope to see the man.
2. The man I saw had a telescope.

Proper syntactic parsing can clarify whether the prepositional phrase "with the telescope" modifies "the man" or "I."

d. Scope of Modifiers and Quantifiers:

Modifiers: (e.g., adjectives, adverbs) and quantifiers (e.g., "all," "some") are interpreted based on their position in a sentence. Syntax governs how these elements affect the meaning of the words they modify.

- Example: "All students did not attend the lecture."
- This sentence could mean:

1. No students attended the lecture (negation applies to the entire group).
2. Some students did not attend the lecture (negation applies to a subset).

The placement of "not" relative to "all" creates ambiguity, and a clear syntactic structure helps determine the intended meaning.

e. Dependency Parsing: In computational linguistics, dependency parsing analyzes the syntactic structure of a sentence by determining which words depend on others. This method helps in understanding the relationships between words and how these relationships contribute to meaning.

- Example: "The boy kicked the ball."
- Dependency parsing would show that "boy" is the subject of the verb "kicked" and "ball" is the object, making the syntactic roles explicit.

f. Sentence Complexity and Subordination: In complex sentences, syntax plays a role in determining how subordinate clauses relate to the main clause. This is essential for understanding cause-effect relationships, conditions, and other relationships between actions.

- Example: "If it rains, the game will be canceled."
  - The conditional structure here ("If it rains") modifies the meaning of the main clause ("the game will be canceled"). The syntax indicates that the cancellation depends on the condition of rain.

g. Role of Punctuation and Intonation: In written text, punctuation marks (commas, periods, semicolons) contribute to the syntactic structure by indicating pauses, separations, or connections between parts of a sentence. In spoken language, intonation can serve a similar role by grouping words into syntactic units and influencing meaning.

• Example: "Let's eat, Grandma!" vs. "Let's eat Grandma!"

▪ The comma changes the meaning dramatically. In the first sentence, Grandma is being addressed, whereas in the second, it suggests Grandma is on the menu!

h. Tree Structures and Parse Trees: In NLP, sentences are often represented as parse trees or syntax trees that depict their hierarchical structure. These trees show how phrases and words are grouped together and how they relate to one another.

• Example: A simple parse tree for "The cat sat on the mat" would show that "the cat" is the noun phrase (NP), "sat" is the verb (V), and "on the mat" is a prepositional phrase (PP) modifying the verb.

i. Meaning in Different Languages In some languages, syntax plays a more prominent role in determining meaning due to strict word order rules. In contrast, other languages, like Latin or Russian, rely more on morphology (word endings) than syntax to convey grammatical relationships.

• Example: In English (a word-order-dependent language), the sentence "John loves Mary" cannot be rearranged without changing meaning. In contrast, in a language like Latin, word order is more flexible due to case markings, and "John loves Mary" could be represented with different word orders without altering meaning.

Semantic Analysis

Syntax-Driven Semantic Analysis

Definition: Syntax-driven semantic analysis — assigning meaning representations based soley on static knowledge from the lexicon and the grammar. [JM]

This provides a representation that is "both context independent and inference free." [JM], presumably referring to semantic context.

Definition: principle of compositionality — the meaning of a sentence is the sum of the meanings of its constituent elements.
NB this includes the relationships among these elements (ordering, grouping, etc.) I.e., the meaning of a sentence is "partially based on its syntactic structure."

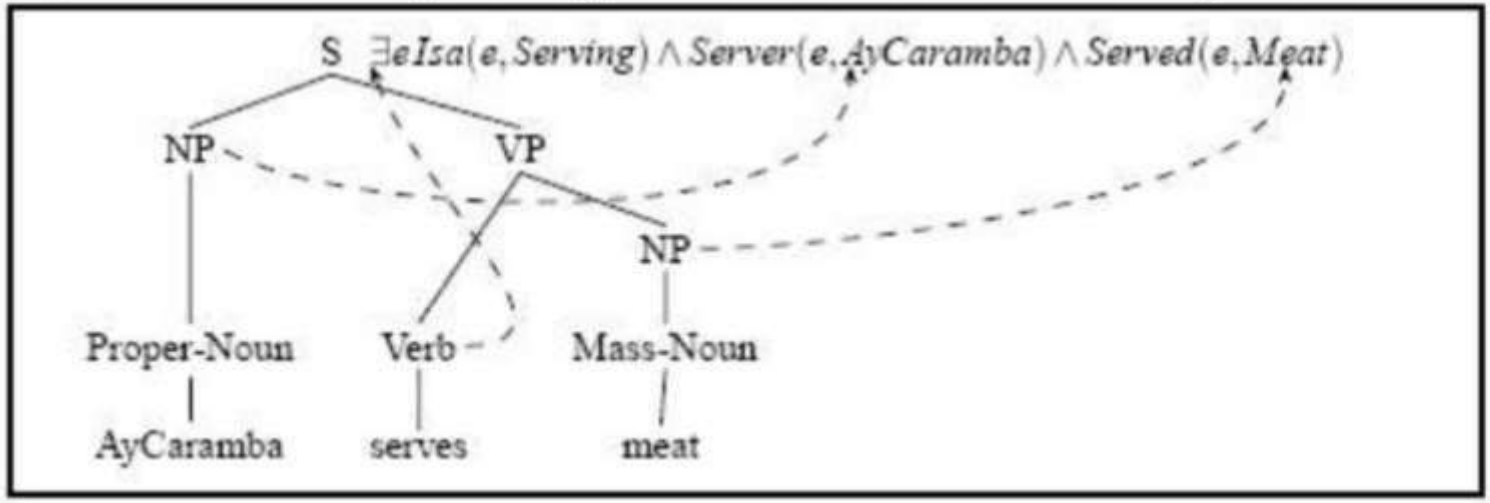
Pipeline Representation

```
Parser => Syntactic Representation => Semantic Analyzer =>
Meaning Representation
```

Example: Rudimentary analysis of:

15.1 AyCaramba serves meat.

[JM Fig. 15.2] shows a simplified parse tree, **annotated** with **meaning representation** (lacking **feature attachments**.)



In English, this says

- There exists an element `e` which satisfies the predicate of being an event of `Serving` [a serving event]
  ∃ `e` such that `e` Is a `Serving` instance
- And `e`, together with `AyCaramba` also satisfies the predicate `Server`, i.e., `AyCaramba` is the server in the event
  ∧ `Server(e,AyCaramba)`
- And `e`, together with `Meat` satisfies the predicate `Served`, i.e., meat is served during the event.
  ∧ `Served(e,Meat)`

	PRODUCTION	SEMANTIC RULES
1)	$L \to E\ \mathbf{n}$	$L.val = E.val$
2)	$E \to E_1\ +\ T$	$E.val = E_1.val + T.val$
3)	$E \to T$	$E.val = T.val$
4)	$T \to T_1\ *\ F$	$T.val = T_1.val \times F.val$
5)	$T \to F$	$T.val = F.val$
6)	$F \to (\ E\ )$	$F.val = E.val$
7)	$F \to \mathbf{digit}$	$F.val = \mathbf{digit}.\text{lexval}$

Figure 5.1: Syntax-directed definition of a simple desk calculator

For the SDD(SYNTAX-DIRECTED DEFINITIONS) of Fig. 5.1, give annotated parse trees for the following expressions:

a. $(3 + 4) * (5 + 6)n.$
b. $1 * 2 * 3 * (4 + 5)n.$

Syntax

The lexicon of a first order language contains the following:

- Connectives and Parentheses: $\neg$, $\rightarrow$, $\leftrightarrow$, $\wedge$, $\vee$, (and);
- Quantifiers: $\forall$ (universal) and $\exists$ (existential);
- Variables: $x, y, z, \ldots$ ranging over particulars (individual objects);
- Constants: $a, b, c, \ldots$ representing a specific element;
- Functions: $f, g, h, \ldots$, with arguments listed as $f(x_1, \ldots x_n)$;
- Relations: $R, S, \ldots$ with an associated arity.

From Natural Language to First order logic (or vv.). Consider the following three sentences:

- "Each animal is an organism"
- "All animals are organisms"
- "If it is an animal then it is an organism"

This can be formalised as:

$$\forall x(Animal(x) \rightarrow Organism(x)) \tag{3.1.1}$$

Observe the colour coding in the natural language sentences: 'each' and 'all' are different ways to say "$\forall$" and the 'is a' and 'are' in the first two sentences match the "$\rightarrow$", and the combination of the "$\forall$" and "$\rightarrow$" in this particular construction can be put into natural language as 'if ... then'.

with the 'exist' matching the ∃, and

– "There are books that are heavy"

(well, at least one of them is) as:

$$\exists x(Book(x) \wedge heavy(x)) \tag{3.1.3}$$

where the 'there are' is another way to talk about "∃" and the 'that' hides (linguistically) the "∧". A formulation like "There is at least one book, and it is heavy" is a natural language rendering that is closer to the structure of the axiom.

A sentence—or, more precisely, its precise meaning—such as "Each student must be registered for a degree programme" requires a bit more consideration. There are at least two ways to say the same thing in the way the sentence is formulated (we leave the arguments for and against each option for another time):

1. $\forall x, y(registered for(x, y) \rightarrow Student(x) \wedge DegreeProgramme(y))$

"if there is a registered for relation, then the first object is a student and the second one a degree programme"

$\forall x(Student(x) \rightarrow \exists y registered for(x, y))$

"Each student is registered for at least one y", where the y is a degree programme (taken from the first axiom)

2. $\forall x(Student(x) \rightarrow \exists y(registered for(x, y) \wedge DegreeProgramme(y)))$

"'Each student is registered for at least one degree programme'

Machine learning algorithm-based automated semantic analysis

One can train machines to make near-accurate predictions by providing text samples as input to semantically-enhanced **ML algorithms**. Such estimations are based on **previous observations** or **data patterns**.

Machine learning-based semantic analysis involves sub-tasks such as relationship extraction and word sense disambiguation.

1. Word sense disambiguation

In semantic analysis, word sense disambiguation refers to an automated process of determining the sense or meaning of the word in a given context. As natural language consists of words with several meanings (polysemic), the objective here is to recognize the correct meaning based on its use.

For example, ‘Raspberry Pi’ can refer to a fruit, a single-board computer, or even a company (UK-based foundation). Hence, it is critical to identify which meaning suits the word depending on its usage.

2. Relationship extraction

Relationship extraction is a procedure used to determine the semantic relationship between words in a text. In semantic analysis, relationships include various entities, such as an individual's name, place, company, designation, etc. Moreover, semantic categories such as, 'is the chairman of,' 'main branch located a'', 'stays at,' and others connect the above entities.

Let's consider a phrase as an example. 'Elon Musk is one of the co-founders of Tesla, which is based in Austin, Texas.'

This phrase illustrates two different relationships.

Elon Musk is the co-founder of Tesla
[Person] [Company]

Tesla is based in Austin, Texas
[Company] [Place]

Semantic analysis techniques

The semantic analysis uses two distinct techniques to obtain information from text or corpus of data. The first technique refers to text classification, while the second relates to text extractor.

1. Semantic classification

Semantic classification implies text classification wherein predefined categories are assigned to the text for faster task completion. Following are the various types of text classification covered under semantic analysis:

- **Topic classification**: This classifies text into preset categories on the basis of the content type. For example, customer support teams in a company may intend to classify the tickets raised by customers at the help desk into separate categories so that the concerned teams can address them. In this scenario, ML-based semantic analysis tools may recognize tickets based on their content and classify them under a 'payment concern' or 'delayed delivery' category.
- **Sentiment analysis**: Today, sentiment analysis is used by several social media platforms such as Twitter, Facebook, Instagram, and others to detect positive, negative, or neutral emotions hidden in text (posts, stories). These sentiments, in a way, denote urgency and may raise 'call to action' alarms for respective platforms. Sentiment analysis helps brands identify dissatisfied customers or users in real-time and gets a hint on what customers feel about the brand as a whole.
- **Intent classification**: Intent classification refers to the classification of text based on customers' intentions in the context of what they intend to do next. You can use it to tag customers as 'interested' or 'not Interested' to effectively reach out to those customers who may intend to buy a product or show an

2. Semantic extraction

Semantic extraction refers to extracting or pulling out specific data from the text. Extraction types include:

- **Keyword extraction**: This technique helps identify relevant terms and expressions in the text and gives deep insights when combined with the above classification techniques.
  For example, one can analyze keywords in multiple tweets that have been labeled as positive or negative and then detect or extract words from those tweets that have been mentioned the maximum number of times. One can later use the extracted terms for automatic tweet classification based on the word type used in the tweets.

- **Entity extraction**: As discussed in the earlier example, this technique is used to identify and extract entities in text, such as names of individuals, organizations, places, and others. This method is typically helpful for customer support teams who intend to extract relevant information from customer support tickets automatically, including customer name, phone number, query category, shipping details, etc.

Word Sense Disambiguation

Word sense disambiguation, in natural language processing (NLP), may be defined as the ability to determine which meaning of word is activated by the use of word in a particular context.

Lexical ambiguity, syntactic or semantic, is one of the very first problem that any NLP system faces.

Part-of-speech (POS) taggers with high level of accuracy can solve Word's syntactic ambiguity.

On the other hand, the problem of resolving semantic ambiguity is called WSD (word sense disambiguation). Resolving semantic ambiguity is harder than resolving syntactic ambiguity.

Word Sense Disambiguation

Window

Window

Window

Please open the window to
let some fresh air in.

For example, consider the two examples of the distinct sense that exist for the word ***"bass"*** –

- I can hear bass sound.
- He likes to eat grilled bass.

The occurrence of the word **bass** clearly denotes the distinct meaning. In first sentence, it means **frequency** and in second, it means **fish**. Hence, if it would be disambiguated by WSD then the correct meaning to the above sentences can be assigned as follows –

- I can hear bass/frequency sound.
- He likes to eat grilled bass/fish.

Evaluation of WSD

The evaluation of WSD requires the following two inputs –

A Dictionary

The very first input for evaluation of WSD is dictionary, which is used to specify the senses to be disambiguated.

Test Corpus

Another input required by WSD is the high-annotated test corpus that has the target or correct-senses. The test corpora can be of two types &minsu;

- **Lexical sample** – This kind of corpora is used in the system, where it is required to disambiguate a small sample of words.
- **All-words** – This kind of corpora is used in the system, where it is expected to disambiguate all the words in a piece of running text.

Word Sense Disambiguation Applications

WSD has many applications in various text processing and NLP fields.

- WSD can be used alongside Lexicography. Much of the modern Lexicography is corpus-based. WSD, used in Lexicography can provide significant textual indicators.
- WSD can also be used in Text Mining and Information Extraction tasks. As the major purpose of WSD is to accurately understand the meaning of a word in particular usage or sentence, it can be used for the correct labeling of words.
- For example, from a security point of view, a text system should be able to understand the difference between a **coal "mine"** and a **land "mine"**.
- While the former serves industrial purposes, the latter is a security threat. So a text mining application must be able to determine the difference between the two.
- Similarly, WSD can be used for Information Retrieval purposes. Information Retrieval systems work through text data primarily based on textual information. Knowing the relevance of using a word in any sentence will surely help.

Challenges in Word Sense Disambiguation

WSD faces a lot of challenges and problems.

- The most common problem is the difference between various dictionaries or text corpus. Different dictionaries have different meanings for words, which makes the sense of the words to be perceived as different. A lot of text information is out there and often it is not possible to process everything properly.
- Different applications need different algorithms and that is often a challenge for WSD.
- A problem also arises is that words cannot be divided into discrete meanings. Words often have related meanings and this causes a lot of problems.

Lesk Algorithm

Lesk Algorithm is a classical Word Sense Disambiguation algorithm introduced by Michael E. Lesk in 1986.

The Lesk algorithm is based on the idea that words in a given region of the text will have a similar meaning. In the Simplified Lesk Algorithm, the correct meaning of each word context is found by getting the sense which overlaps the most among the given context and its dictionary meaning.

We can use NLTK to implement Lesk in Python.

Let us start by importing the libraries.

```
from nltk.wsd import lesk
from nltk.tokenize import word_tokenize
```

Let us now proceed with some examples.

```
a1= lesk(word_tokenize('This device is used to jam the signal'),'jam')
print(a1,a1.definition())
a2 = lesk(word_tokenize('I am stuck in a traffic jam'),'jam')
print(a2,a2.definition())
```

Output:

```
Synset('jamming.n.01') deliberate radiation or reflection of electromagnetic energy for
Synset('jam.v.05') get stuck and immobilized
```

So, here we see that the first meaning was correctly perceived, and the 2nd was also correct.

```
# testing with some data
b1= lesk(word_tokenize('Apply spices to the chicken to season it'),'season')
print(b1,b1.definition())
```

Output:

```
Synset('season.v.01') lend flavor to
```

In this case, also, the output is correct. The word "*season*" is used here from the cooking sense of view.

Let us try another usage of the same word.

```
b2= lesk(word_tokenize('India receives a lot of rain in the rainy season'),'season')
print(b2,b2.definition())
```

Output:

```
Synset('season.n.01') a period of the year marked by special events or activities in
```

```
c1= lesk(word_tokenize('Water current'),'current')
print(c1,c1.definition())
```

Output:

```
Synset('stream.n.02') dominant course (suggestive of running water) of successive eve
```

Correct output again.

Let us try with different use of the word "*current*".

```
# testing with some data
c1= lesk(word_tokenize('The current time is 2 AM'),'current')
print(c1,c1.definition())
```

Output:

```
Synset('current.a.01') occurring in or belonging to the present time
```

Robust Analysis

Techniques for handling ambiguity and ensuring robust meaning representation

1. Disambiguation Techniques

A. Word Sense Disambiguation (WSD)

Example: The word "bank" can refer to a financial institution or the side of a river. WSD helps determine the correct meaning based on context, like in "I deposited money in the bank" vs. "The river bank was eroded."

B. Contextual Embeddings

Techniques like Word2Vec, GloVe, and especially contextual models like BERT and ELMo create word embeddings that vary based on context.

Example: In the sentences "He went to the bank to fish" and "He went to the bank to deposit money," the embeddings for "bank" would differ, helping algorithms disambiguate its meaning

2. Syntactic Parsing Techniques

A. Dependency Parsing

Dependency parsing analyzes the grammatical structure of a sentence by establishing relationships between words.
This helps clarify syntactic ambiguities by identifying which words are dependent on others, thus revealing the intended meaning.

Example: In "I saw the man with a telescope," a dependency parser can provide multiple interpretations based on different attachments to "saw" or "man."

B. Constituency Parsing

This approach breaks down sentences into sub-phrases or constituents based on grammar rules.

Example: For "The old man sells the boat," constituency parsing can clarify that "old man" refers to the subject and "boats" is the object, despite its surface appearance of being ungrammatical.

3. Semantic Role Labeling (SRL)

SRL identifies the roles that words play in the context of a sentence, providing insights into the meaning of sentences.

Example: In "Alice gave Bob a book," SRL helps clarify that "Alice" is the agent, "Bob" is the rec
ipient, and "a book" is the instrument, thus maintaining robust meaning representation.

4. Probabilistic Models

These models use statistical techniques to infer the most likely interpretations of ambiguous language.
•**Techniques**:

•**Hidden Markov Models (HMMs)**: Used for tasks like part-of-speech tagging where states represent different parts of speech.
•**Bayesian Models**: These consider prior probabilities to resolve ambiguities based on the likelihood of certain meanings given the context.

Example: In a probabilistic model, if "bark" is more frequently used as a sound made by a dog in a particular corpus, it will bias the interpretation toward that sense in similar contexts.

5. Contextual Information Utilization

This involves using external context or previous discourse to clarify ambiguous expressions.

Example: In a conversation where someone asks, "Can you bring that over here?" the reference to "that" can be clarified based on the previous context, such as the objects being discussed.

6. Multi-modal Approaches

Incorporating multiple sources of information, such as text, images, and audio, to understand meaning more robustly.

Example: In an application recognizing actions in videos, the combination of visual input (what is happening) and textual descriptions can help clarify ambiguous statements like "He is holding a bat," determining whether it refers to sports equipment or a baseball bat.

Lexemes and Their Senses

A lexeme is the dictionary entry of a word, capturing its core meaning.

Key Characteristics of Lexemes:

1. **Abstract Representation**: A lexeme serves as an abstract representation of a word, allowing for various inflections and derivations while maintaining its core meaning.
2. **Includes Different Forms**: Lexemes include all the grammatical forms of a word. For example, the lexeme "run" includes "run," "runs," "running," and "ran."
3. **Different from Morphological Forms**: While a lexeme represents the general meaning of a word, its various inflected or derived forms (called **morphs**) represent specific instances of that meaning. For example:

- Lexeme: "walk"
- Morphs: "walk," "walks," "walking," "walked"

Examples of Lexemes

- **Lexeme**: "cat"
- **Morphs**: "cat" (singular), "cats" (plural)
- **Lexeme**: "happy"
- **Morphs**: "happy," "happier," "happiest," "happiness"
- **Lexeme**: "write"
- **Morphs**: "write," "writes," "writing," "wrote," "written"

Importance in Natural Language Processing (NLP)

In NLP, understanding lexemes is essential for tasks such as:

- **Tokenization**: Identifying lexemes as the basic units of text for further processing.
- **Stemming and Lemmatization**: Reducing words to their base forms or lexemes to analyze their meanings effectively.
- **Semantic Analysis**: Using lexemes to establish relationships between words and their meanings within sentences.