



NLP - Module 4 Gaurav Panjabi

What is semantic analysis, and why is it considered difficult?

- Semantic analysis is the process of understanding the meaning of text.
- Semantic analysis, also known as semantic parsing or semantic understanding, is a crucial component of natural language processing (NLP) that focuses on extracting the meaning and intent from human language.
- It goes beyond the syntactic structure (grammar and arrangement of words) of a sentence to understand the semantics, or the meaning, of the words and how they relate to each other.
- It involves identifying the relationships between words and phrases, as well as the overall meaning of the text.
- Semantic analysis is difficult because language is complex and ambiguous.
- Semantic analysis is also challenging because of the richness of human language.
- Words can have multiple meanings, and the meaning of a sentence can depend on the context in which it is used.
- For example, the sentence "The bank is on the river" could mean that a financial institution is located on the bank of a river, or that a cliff is on the edge of a river.

Semantic analysis is challenging for several reasons: **(MARKS KE ACCORDING NUMBER OF POINTS LIKHNA AND AGAR CHULL H TOH SUB LIKH KE AAO TOMPR LOG)**

1. **Named Entity Recognition (NER):** Identifying named entities, such as names of people, organizations, locations, and other specific entities, is crucial for understanding the context of a text. However, names can be ambiguous (e.g., "Apple" could refer to a fruit or a technology company), and entities might be expressed in various forms (e.g., abbreviations, acronyms). NER is challenging

because it requires knowledge about specific domains and the ability to disambiguate entities based on the context in which they appear.

2. **Ambiguity:** Natural languages are inherently ambiguous. Words and phrases can have multiple meanings depending on the context in which they are used. Disambiguating these meanings accurately is a complex task.
3. **Context Dependency:** The meaning of a word or phrase often depends on the surrounding words and the broader context of the conversation. Capturing this context and understanding its impact on meaning is difficult.
4. **Sarcasm and Irony:** People often use sarcasm, irony, or other forms of figurative speech, which can be difficult to detect without understanding the speaker's tone and intention.
5. **World Knowledge:** Understanding text often requires background knowledge about the world. For example, understanding a news article might require knowledge about current events and historical context.
6. **Word Sense Disambiguation:** Determining which meaning of a word is being used in a particular context is a fundamental challenge. Words can have different meanings, and selecting the correct one is crucial for accurate semantic analysis.
7. **Discourse:** The meaning of a sentence can also depend on the discourse that has come before it. For example, the sentence "He's a great guy" has a different meaning if it is said in the context of a conversation about someone's friend than if it is said in the context of a conversation about someone's romantic partner.
8. **Common sense:** In order to fully understand the meaning of a text, it is often necessary to have some common sense knowledge. For example, in order to understand the sentence "The cat sat on the mat," it is necessary to know that cats are animals that can sit, and that mats are objects that cats can sit on.
9. **Temporal References:** Text often contains references to time, such as dates, durations, and events. Understanding the temporal relationships between different events and their implications on the overall meaning of a text can be complex. Temporal expressions might be explicit ("yesterday," "next week") or implicit ("after the meeting," "before the event"), and correctly interpreting them is essential for understanding the chronological order of events and the causal relationships between them.
10. **Cultural Variations:** Different cultures have unique language structures, idiomatic expressions, and social norms. Understanding these variations is vital

for interpreting text accurately, especially in a global context where communication often happens across diverse cultural boundaries. Cultural nuances impact how words and phrases are used, making semantic analysis more challenging when dealing with multicultural or international communication.

Explain various approaches to perform semantic analysis.

1. Rule-Based Systems:

- **Description:** Rule-based systems use predefined rules to match patterns in text and extract semantic information.
- **Pros:** They are interpretable, and domain experts can easily define rules.
- **Cons:** Limited scalability and adaptability, as rules need to be manually crafted and updated.

2. Statistical Approaches:

- **Description:** Statistical models, such as Hidden Markov Models (HMMs) or Conditional Random Fields (CRFs), can be trained on labeled data to learn patterns and relationships between words in a sentence.
- **Pros:** Learning from data allows for adaptability to different domains.
- **Cons:** May require large amounts of annotated data, and the models may not capture complex semantic relationships well.

3. Machine Learning and Deep Learning:

- **Description:** Supervised machine learning models, including traditional classifiers (e.g., Support Vector Machines) and deep learning architectures (e.g., neural networks), can be trained for semantic analysis tasks.
- **Pros:** Can handle complex relationships and patterns, and can generalize well to new data.
- **Cons:** Require substantial labeled data for training, and deep learning models may be computationally intensive.

4. Semantic Role Labeling (SRL):

- **Description:** SRL involves identifying the roles played by different words in a sentence, such as the agent, patient, or theme.
- **Pros:** Provides a structured representation of the semantics in a sentence.

- **Cons:** May not capture all nuances of meaning, and performance can be sensitive to sentence structure.

5. Dependency Parsing:

- **Description:** Dependency parsing involves analyzing the grammatical relationships between words in a sentence, which can be used to infer semantic relationships.
- **Pros:** Captures syntactic and semantic dependencies between words.
- **Cons:** Dependency parsing can be challenging for languages with flexible word order.

6. Semantic Graphs:

- **Description:** Representing sentences as semantic graphs, where nodes are entities or concepts, and edges represent relationships between them.
- **Pros:** Provides a structured representation of relationships in a sentence.
- **Cons:** Creating accurate semantic graphs may require sophisticated algorithms.

Discuss different semantic relationships between words. OR Explain, with suitable examples, the following relationships between word meanings: Homonymy, Polysemy, Synonymy, Antonymy, Hypernymy, Hyponymy, and Meronymy. <IMP>


1. Homonymy:

- Homonyms are words that share the same spelling or pronunciation but have different meanings.
- For instance, "bat" can refer to a flying mammal or a piece of sports equipment used in baseball.
- Homonymy can lead to confusion in language comprehension, especially in written texts where the context might not be immediately clear.
- Example:
 - bat (animal)
 - bat (sports equipment)

- key (to open a door)
- key (musical term)

2. Polysemy:

- Polysemy occurs when a single word has multiple related meanings.
- These meanings are usually linked through metaphorical extension or historical usage.
- For example, "head" can mean the physical part of the body, the leader of an organization, or the top of a bottle.
- Context is essential in understanding which specific meaning of a polysemous word is intended in a particular instance.
- Example:
 - bank (financial institution)
 - bank (side of a river)
 - bank (row of objects)

 The difference between homonymy and polysemy is a matter of **relatedness**.
(FOR OUR UNDERSTANDING)

- **Homonyms** are words that have the same spelling or pronunciation, but different meanings, and **are not related**.
- **Polysemous** words are words that have multiple meanings, but **are related in some way**.

For example, the words "bat" and "bat" are homonyms. They have the same spelling and pronunciation, but they have different meanings. The word "bat" can refer to a flying mammal, while the word "bat" can refer to a piece of sports equipment. The two words are not related in any way.

On the other hand, the word "bank" is polysemous. It has multiple meanings, but they are all related to the basic concept of a financial institution. The word "bank" can refer to a physical building where people can deposit and withdraw money, or it can refer to the financial institution itself. The word "bank" can also be used figuratively to refer to a large reserve of something, such as a "bank of knowledge."

3. Synonymy:

- Synonyms are words with similar meanings that can be used interchangeably in some contexts.
- Synonyms have closely related meanings. For example, "happy" and "joyful" both express positive emotions.
- However, synonyms are not always perfectly interchangeable; they can have subtle differences in connotation, formality, or usage.
- Example:
 - big - large
 - happy - joyful
 - sad - unhappy

4. Antonymy:

- Antonyms are words with opposite meanings.
- There are two main types of antonyms:
 - **Gradable Antonyms:** These represent the two ends of a scale and can have intermediate degrees. For example, "hot" and "cold."
 - **Complementary Antonyms:** These are pairs where one word's presence implies the absence of the other. For example, "alive" and "dead." Something cannot be both alive and dead at the same time.
- Example:
 - big - small
 - happy - sad
 - hot - cold

5. Hypernymy

- **Hypernymy:** Hypernyms are words that represent broader categories or general concepts.
- For example, "**animal**" is a **hypernym** of "*dog*" and "*cat*."
- Understanding these relationships is crucial for organizing vocabulary hierarchies and categorizing words in lexical databases.
- Example:

- animal (hypernym of dog and cat)
- vehicle (hypernym of car and truck)
- food (hypernym of apple and banana)

6. Hyponymy:

- **Hyponymy:** Hyponyms are specific instances within a broader category.
- Example, "**dog**" and "**cat**" are **hyponyms** of "*animal*."
- Understanding these relationships is crucial for organizing vocabulary hierarchies and categorizing words in lexical databases.
- Example:
 - dog (hyponym of animal)
 - car (hyponym of vehicle)
 - apple (hyponym of food)

7. Meronymy:

- Meronyms are words that refer to parts of a whole.
- There are two types:
 - **Partitive Meronyms:** These indicate a part-whole relationship, such as "wheel" being a part of a "car."
 - **Substance Meronyms:** These indicate the substance a whole object is made of, like "water" being a substance meronym of "ocean."
- Example:
 - wheel (meronym of car)
 - arm (meronym of human)
 - leaf (meronym of tree)

What is WordNet? What is its structure? <IMP.>

- WordNet is a comprehensive lexical database of the English language developed by researchers at Princeton University.
- It organizes words into sets of synonyms, known as synsets, grouping together words that share similar meanings and semantic relationships.

- Each synset represents a distinct concept and includes words belonging to various parts of speech, such as nouns, verbs, adjectives, and adverbs.
- WordNet not only offers definitions for these synsets but also illustrates their usage through example sentences.
- One of its key strengths lies in capturing the intricate web of semantic relationships, including hypernymy (generalization), hyponymy (specification), meronymy (part-whole relationships), and antonymy (opposite meanings).
- With its hierarchical structure and rich semantic data, WordNet serves as a vital resource for natural language processing applications, aiding tasks like word sense disambiguation, information retrieval, and machine translation by providing a structured framework for understanding the meanings and connections between words in the English language.

Structure of WordNet:

1. Synsets:

- **Synonyms:** Words in the same synset are considered synonyms, meaning they have similar meanings. For example, the synset for "car" might include words like "automobile," "vehicle," and "motorcar."

2. Hypernyms and Hyponyms:

- **Hypernyms:** These are broader terms. For example, "animal" is a hypernym of "dog."
- **Hyponyms:** These are more specific terms. In the example above, "dog" is a hyponym of "animal."

3. Meronyms and Holonyms:

- **Meronyms:** These are parts or components of something. For example, "wheel" is a meronym of "car."
- **Holonyms:** These are whole items that contain parts. Using the previous example, "car" is a holonym of "wheel."

4. Antonyms:

- WordNet provides information about antonyms, words with opposite meanings. For instance, the antonym of "hot" is "cold."

5. Attributes:

- WordNet captures adjectival relationships, including attributes. For example, the adjective "tall" might have attributes like "height" or "stature."

6. Verb Hierarchies:

- WordNet also includes information about verb relationships, including entailment (e.g., "snore" entails "sleep"), troponyms (verbs that specify a manner of doing something, e.g., "run" and "jog"), and more.

7. Similarity Measures:

- WordNet provides measures of semantic similarity between words or concepts. These measures help in tasks such as word sense disambiguation and information retrieval.

8. Usage Examples:

- WordNet includes sample sentences illustrating how words are used in context.

Use Cases:

- **Word Sense Disambiguation (WSD):** WordNet is used to determine the correct sense of a word in context.
- **Information Retrieval:** WordNet's structure helps improve search accuracy by understanding synonyms and related terms.
- **Text Mining:** WordNet aids in identifying relationships between words in large text corpora, enabling researchers to extract meaningful information.
- **Sentiment Analysis:** Understanding synonyms and antonyms can improve the accuracy of sentiment analysis algorithms by capturing subtle differences in meaning.
- **Machine Translation:** WordNet helps in selecting the most appropriate translation of a word based on its context and meaning.

How is "sense" defined in WordNet? What is its significance for WSD ? <IMP>

- In WordNet, a "sense" refers to a specific meaning of a word.
- Words often have multiple senses, each representing a distinct meaning or usage. For example, the word "bank" can refer to a financial institution or the side of a river. Each of these meanings constitutes a separate sense in WordNet.

- The concept of senses is of paramount importance in Word Sense Disambiguation (WSD).
- WSD is the task of determining the correct sense of a word in a particular context.
- It is a crucial challenge in natural language processing because many words have multiple meanings, and understanding the intended sense is essential for accurate language understanding and interpretation.
- WordNet provides a structured and organized way to enumerate and define these senses.
- Each word sense, or synset, is associated with a specific meaning, and the relationships between these synsets (such as hypernymy, hyponymy, and meronymy) provide context for understanding the nuances between different word senses.
- For WSD, computational algorithms often rely on WordNet's sense inventory. By mapping the words in a context to WordNet senses, these algorithms can disambiguate which specific sense of a word is being used in a particular sentence or phrase.
- This disambiguation process helps improve the accuracy of various natural language processing applications, such as machine translation, information retrieval, and question answering, where understanding the precise meaning of words in context is crucial for generating meaningful and relevant responses.
- WordNet's sense distinctions and relationships play a vital role in enhancing the semantic accuracy of these applications.
- For example, consider the sentence "The man went to the bank." Using WordNet, a WSD system can identify the two possible senses of the word "bank" in this sentence: (1) financial institution and (2) side of a river. The WSD system can then use other information, such as the context of the sentence, to determine which sense of the word "bank" is intended.
- WordNet is not the only resource that can be used for WSD. However, it is one of the most widely used and well-known resources.
- WordNet has been shown to be effective for WSD in a variety of tasks, such as machine translation, question answering, and text summarization.

5. What do you mean by Word Sense Disambiguation (WSD)? Discuss the dictionary-based approach for WSD. <V.V.V.V.IMP>

- Word Sense Disambiguation (WSD) is a natural language processing task that aims to determine the correct meaning or sense of a word in a particular context.
- Many words in the English language have multiple meanings, or senses, and the intended sense of a word often depends on the surrounding words in a sentence.
- WSD is crucial for various NLP applications, such as machine translation, information retrieval, question answering, and sentiment analysis, where understanding the correct sense of words is essential for accurate interpretation of the text.
- Consider the word "bank," which can have different meanings:
 1. **Financial institution:** "I deposited my money in the bank."
 2. **Side of a river:** "The children played by the river bank."
- In the sentence "I went to the bank to deposit my savings," without the context, it's unclear whether "bank" refers to a financial institution or the side of a river. WSD algorithms analyze the context of the word in a given sentence to determine the correct sense of the word.
- Various techniques are used for WSD, including rule-based methods, supervised machine learning algorithms (using annotated training data), and unsupervised methods (which rely on algorithms to cluster word senses based on the context in large corpora of text).

Dictionary-based approach for WSD. <V.IMP>

- **Dictionary-based WSD** is a type of WSD that uses dictionaries to identify the correct sense of a word.
- Dictionary-based WSD systems typically work by first identifying all of the possible senses of a word in a given context. This is done by looking up the word in a dictionary.
- Once the possible senses of the word have been identified, the system then uses a variety of factors, such as the context of the word and the context of the sentence, to choose the most likely sense.
- One common dictionary-based WSD approach is the **Lesk algorithm**. The Lesk algorithm works by comparing the definitions of the possible senses of a word to the context of the word. The algorithm then chooses the sense with the most overlap in definitions.

- For example, consider the sentence **"I went to the bank."** The Lesk algorithm would first identify all of the possible senses of the word "bank" in this sentence, which are (1) financial institution and (2) side of a river.
- The algorithm would then compare the definitions of these two senses to the context of the word. The definition of the first sense, "financial institution," is "a business that provides financial services, such as accepting deposits and making loans." The definition of the second sense, "side of a river," is "the land that forms the edge of a river."
- The Lesk algorithm would then choose the first sense, "financial institution," because it has more overlap in definitions with the context of the word.

Dictionary-based WSD systems have a number of advantages. They are relatively simple to implement and are able to handle a wide range of words. However, dictionary-based WSD systems can also be inaccurate, especially for words with multiple senses that are semantically similar.

knowledge- based WSD. <V.IMP>

- Knowledge-based Word Sense Disambiguation (WSD) relies on external knowledge sources to determine the correct sense of a word in a given context.
- The goal is to use structured knowledge representations, such as ontologies, semantic networks, or other knowledge bases, to capture the relationships between words and their meanings.
- Here are some key aspects of knowledge-based WSD:

1. Ontologies and Semantic Networks:

- Ontologies are formal representations of knowledge that include concepts, relationships, and properties. Semantic networks represent semantic relations between entities.
- Knowledge-based WSD may involve using ontologies and semantic networks to model the meanings of words and their connections in a more detailed and nuanced way than a simple dictionary.

2. Semantic Relations:

- Knowledge-based approaches often leverage semantic relations between words. For example, if two words are related in a semantic network, they may be more likely to share similar senses.

- Understanding hypernyms, hyponyms, meronyms, and other semantic relationships can aid in disambiguating word senses.

3. Contextual Information:

- Knowledge-based WSD considers the context in which a word appears. It takes into account the surrounding words and their meanings to infer the most appropriate sense for the ambiguous word.
- Contextual information can be used to disambiguate between different senses of a word based on the overall meaning of the sentence or document.

4. Machine Learning with Knowledge Features:

- Some knowledge-based WSD methods may integrate machine learning techniques. In this case, features derived from external knowledge sources are used alongside other features to train models for disambiguation.
- The features could include information about the relationships between words, the structure of the knowledge base, or the prevalence of different senses in a given context.

5. Evaluation and Challenges:

- Knowledge-based WSD systems are evaluated based on their ability to correctly identify the sense of words in various contexts, often using standard evaluation datasets.
- Challenges include the dynamic nature of language and the need for continuously updated knowledge bases to capture evolving meanings and relationships.

Explain Yarowsky bootstrapping approach of semi supervised learning. / Explain the Semi-supervised method (Yarowsky) for Word Sense Disambiguation in detail. <IMP>

- The Yarowsky algorithm is a semi-supervised learning method used for word sense disambiguation (WSD).
- Word sense disambiguation is the task of determining the correct sense of a word in a particular context, especially when a word has multiple meanings. The Yarowsky algorithm is one of the classic approaches in this field.

- The Yarowsky algorithm was proposed by David Yarowsky in 1995.
- It leverages a small amount of labeled data (instances where the senses of words are known) and a large amount of unlabeled data to disambiguate word senses.
- The basic idea behind the Yarowsky algorithm is to use the labeled data to create a set of seed words that are highly indicative of specific senses of the ambiguous word. Then, these seed words are used to classify the unlabeled data.

Step-by-step explanation of the Yarowsky algorithm:

1. **Gather Labeled Data:** Collect a small set of labeled examples where the senses of the ambiguous word are known.
2. **Seed Word Selection:** Identify words in the labeled data that are highly indicative of specific senses of the ambiguous word. These words act as seed words. For instance, if the ambiguous word is "bank," the word "river" might be a seed word for the sense of "bank" related to a river.
3. **Contextual Evidence:** Use these seed words to identify the context in which they appear in the unlabeled data. For example, if the seed word "river" appears in a sentence with the word "money," it might indicate the sense of "bank" related to a financial institution.
4. **Label Unlabeled Data:** Label the instances in the unlabeled data where the seed words appear in indicative contexts. These labeled instances are then used to classify other instances with similar contexts.
5. **Iterative Process:** The process is often iterative, where newly labeled data is used to refine the classification of other instances. This iterative process helps improve the accuracy of sense disambiguation.

Advantages:

1. **Semi-Supervised Learning:** Uses a small labeled dataset to disambiguate senses in large unlabeled data.
2. **Domain Adaptability:** Can be adapted to different languages and contexts.
3. **Iterative Improvement:** Continuously refines its model for better accuracy.
4. **Effective for Polysemous Words:** Works well for words with multiple meanings and distinct contexts.

Disadvantages:

1. **Dependency on Seed Words:** Accuracy depends heavily on selecting appropriate seed words.
2. **Limited to Context:** Relies only on contextual information, ignoring deeper semantic and syntactic nuances.
3. **Data Intensity:** Requires substantial unlabeled data, limiting its application in data-scarce situations.
4. **Sense Overlap:** Struggles when word senses overlap in contexts, leading to ambiguity.

Explain Unsupervised Word Sense Disambiguation (Hyperlex). ← HYPERLEX IS THE MAIN QUESTION

- Hyperlex is one of the most famous unsupervised approach for Word Sense Disambiguation.
- HyperLex is capable of automatically determining the uses of a word in a text base without recourse to a dictionary.
- Despite of being unsupervised it has been found to be comparable to state of - the art supervised approaches.

Algorithm Steps:

- **Step 1: Tokenize and Clean Corpus:**
 - Suppose we have the following sentence from a corpus: "The bank approved my loan, and I deposited the money in my savings account."
 - The sentence is tokenized into words, and common stop words (e.g., "the," "and," "in") are removed: "bank approved loan, deposited money savings account."
- **Step 2: Define Collocations:**
 - In this step, we create pairs of words that occur within three words before and three words after each other in the sentence:
 - ('bank', 'approved')
 - ('approved', 'loan')
 - ('loan', 'deposited')
 - ('deposited', 'money')

- ('money', 'savings')
- ('savings', 'account')
- **Step 3: Rank Collocations and Create a Graph:**
 - We rank these collocations based on their frequency of occurrence in the corpus.
 - Let's assume that the collocation ('bank', 'approved') has the highest frequency.
 - Now, we create a graph where words are nodes, and collocations define edges. The weight of each edge corresponds to the frequency of the collocation:
 - Graph:
 - Nodes: ['bank', 'approved', 'loan', 'deposited', 'money', 'savings', 'account']
 - Edges:
 - ('bank', 'approved') with a high weight
 - Other collocations with lower weights
- **Step 4: Choose a Focus Word and Remove Popular Nodes:**
 - Let's choose "bank" as the focus word for disambiguation. We remove nodes (words) in the direct neighborhood of "bank" (within three words before and after)
 - those that are two hops away, starting with the most popular nodes. In this case, we'll remove 'approved', 'loan', and 'deposited' due to their high frequency.
 - The remaining nodes around "bank" include 'money', 'savings', and 'account.'
- **Step 5: Identify Associated Words:**
 - The words that remain in the subgraph centered around "bank" after removing popular nodes are considered those most associated with the senses of the focus word.
 - In this context, 'money,' 'savings,' and 'account' are the words associated with the "financial institution" sense of "bank."

- So, based on the context and co-occurrence patterns in the example sentence, we can disambiguate "bank" as referring to a "financial institution" sense rather than a "river bank" sense.

- **Advantages:**

- **Unsupervised:** One of the main advantages is that it doesn't require manually labeled sense data, making it applicable to a wide range of words and domains without the need for extensive training data.
- **Simple and Transparent**
- **Contextual Analysis:** By analyzing the context in which a word appears and identifying associated words, it can capture subtle sense distinctions based on co-occurrence patterns.
- **Customization:** The algorithm is flexible and can be customized for different focus words and context windows, making it adaptable to specific WSD tasks.

- **Disadvantages:**

- **Limited Precision:** It relies solely on co-occurrence patterns and doesn't incorporate deeper linguistic or semantic knowledge.
- **Context Window Sensitivity:** The effectiveness of the algorithm can depend on the choice of the context window (e.g., three words before and after). If the window is too narrow or too wide, it may miss important context or include irrelevant information.
- **Lack of Sense Discrimination:** It may struggle with distinguishing fine-grained senses of a word, particularly when context alone is insufficient to differentiate them.

- **Difficulty with Homonyms:** For words with multiple senses that have little overlap in their co-occurrence patterns,

- **Disadvantages:**

- **Difficulty with Homonyms:** For words with multiple senses that have little overlap in their co-occurrence patterns, the algorithm may face challenges in distinguishing between homonyms (words with different meanings that are spelled and pronounced the same).
- **Scalability:** The algorithm's performance may be affected when applied to large corpora due to increased computational requirements.