Natural Language Processing (TYAIML)

Unit-5

Symantic Analysis:

Semantic analysis is the process of understanding the meaning and interpretation of words, sentences, and text in a way that a computer can comprehend and act upon it.

It goes beyond syntax (grammar and structure) to focus on what the text actually means — its *intent*, *context*, and *relationships between words*.

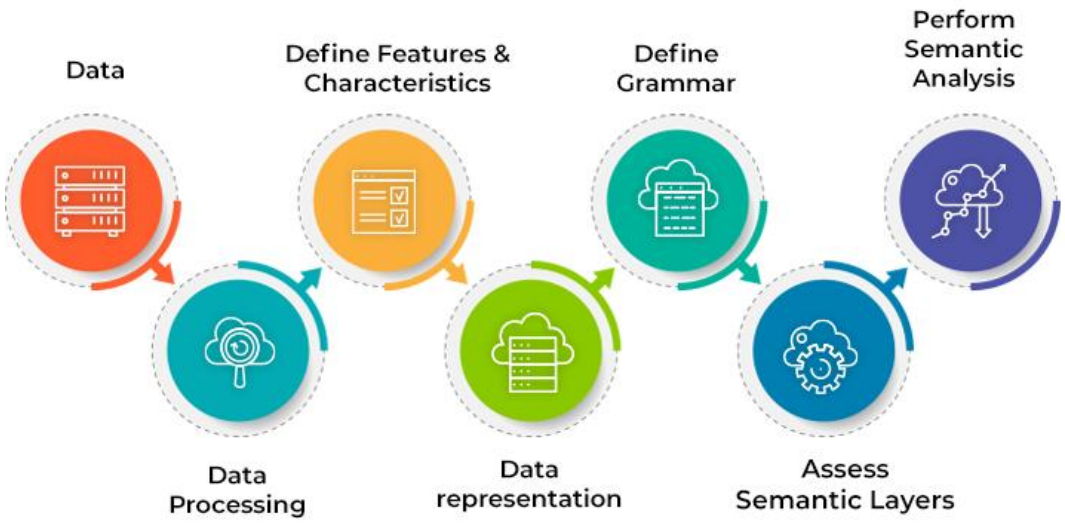
Definition

Semantic analysis in NLP is the process of extracting meaningful information from natural language by analyzing the relationships, context, and intent behind words and phrases.

Objectives

1. **Understand Meaning:**
   Identify what each word or phrase refers to (e.g., "bank" = financial institution vs. riverbank).
2. **Determine Relationships:**
   Recognize how words relate to each other (subject–verb–object, modifiers, etc.).
3. **Interpret Context:**
   Derive overall meaning based on context rather than literal word sense.
4. **Enable Machine Understanding:**
   Help machines "comprehend" human language for tasks like translation, summarization, and dialogue systems.

Workflow or process of semantic analysis:



This figure represents the workflow or process of semantic analysis — a crucial part of Natural Language Processing (NLP) **and** data interpretation. It shows how raw data is transformed into meaningful insights through several structured steps.

Here's a step-by-step explanation of each stage in the figure:

1. Data

- This is the raw input stage**,** where unprocessed information (text, numbers, images, etc.) is collected from various sources.
- Example: Collecting text documents, user reviews, or web content.

2. Data Processing

- In this step, the raw data is cleaned and preprocessed to remove noise or irrelevant information.
- Techniques: tokenization, removing stop words, normalization, stemming, and lemmatization.
- Goal: Prepare the data so it can be effectively analyzed.

3. Define Features & Characteristics

- At this stage, important attributes (features) of the data are identified.
- In NLP, this might include word frequency, sentence length, part of speech tags, etc.
- Purpose: Capture the essential characteristics that represent the meaning of the data.

4. Data Representation

- The processed data is converted into a structured or numerical format suitable for computational analysis.
- Examples:
    - Bag-of-Words or TF-IDF representation
    - Word embeddings (Word2Vec, GloVe, BERT)
- This step enables algorithms to understand and manipulate the data.

5. Define Grammar

- Here, syntactic structures (grammar rules) are established to understand relationships between words and phrases.
- Example: Parsing sentences to identify noun phrases, verb phrases, and dependencies.
- Helps in understanding the grammatical composition of sentences.

6. Assess Semantic Layers

- This stage involves analyzing meanings and relationships between words, phrases, and sentences.

- It determines contextual meaning**,** such as identifying synonyms, antonyms, and entity relations.
- Example: Understanding that “bank” refers to a financial institution, not a riverbank.

7. Perform Semantic Analysis

- The final step integrates all previous stages to extract meaning, context, and intent from the data.
- Applications:
    - Sentiment analysis
    - Machine translation
    - Chatbots and virtual assistants
    - Information retrieval and summarization

Applications of semantic analysis:

The process of understanding the meaning behind words, sentences, and texts in natural language:

1. **Search Engines & Information Retrieval**

- Improving search results by understanding the *intent* behind queries, not just keyword matching.
- Example: Google interpreting “apple” as a company or fruit based on context.

2. Sentiment Analysis

- Determining the *sentiment* (positive, negative, neutral) of text like reviews, social media posts, or customer feedback.
- Helps businesses gauge public opinion or customer satisfaction.

3. Chatbots and Virtual Assistants

- Understanding user queries in a conversational context to provide relevant, meaningful responses.
- Examples: Siri, Alexa, Google Assistant.

4. Machine Translation

- Enhancing translation quality by grasping the *meaning* of sentences rather than literal word-for-word substitution.
- Helps preserve idiomatic expressions and context.

5. Text Summarization

- Automatically generating concise summaries by understanding the main ideas and semantics of documents.
- Useful for news aggregation, legal documents, research papers.

6. Named Entity Recognition (NER)

- Identifying and classifying key elements in text like names of people, places, organizations.
- Used in information extraction, knowledge graphs.

7. Topic Modeling and Classification

- Categorizing documents or articles into topics based on their semantic content.
- Used in organizing large corpora, recommendation systems.

8. Question Answering Systems

- Understanding the semantics of a question to provide accurate answers from a knowledge base or corpus.
- Examples: IBM Watson, advanced FAQ bots.

9. Text-to-Speech and Speech Recognition

- Ensuring that synthesized or recognized speech preserves intended meaning and context.

10. Content Moderation

- Detecting hate speech, offensive content, or misinformation by understanding the meaning rather than just keywords.

Absolutely! Here are simple examples for each semantic analysis application you asked for:

Real life applications

1. Search Engines & Information Retrieval

Example:

User searches for **"apple benefits"**.

- Semantic analysis helps the engine understand whether "apple" refers to the fruit or the tech company based on other query words and user history.
- The engine then returns articles about the health benefits of the fruit, not Apple Inc.

2. Sentiment Analysis

Example:

Review: "The movie was absolutely fantastic, I loved the plot!"

- Semantic analysis classifies this as a positive sentiment**.**

3. Chatbots and Virtual Assistants

Example:

User says, **"Book me a flight to New York next Monday."**

- The chatbot understands the intent to book a flight, identifies the destination (New York), and the date (next Monday), then proceeds with the booking process.

4. Machine Translation

Example:

English: “He kicked the bucket.”

- Semantic analysis helps translate the idiomatic expression correctly into Spanish as “Él murió” (He died), rather than literally translating each word.

5. Text Summarization

Example:

Input article about climate change (long text).

- System outputs: “Climate change is causing rising sea levels and extreme weather events worldwide.”
- The summary captures key points using semantic understanding.

6. Named Entity Recognition (NER)

Example:

Text: “Barack Obama was the 44th President of the United States.”

- NER extracts:
    - Person: Barack Obama
    - Title: President
    - Location: United States

7. Topic Modeling and Classification

Example:

Articles about sports, politics, and technology mixed in a dataset.

- The system categorizes articles automatically into “Sports,” “Politics,” and “Technology” based on semantic content.

8. Question Answering Systems

Example:

User asks: “Who wrote *Pride and Prejudice*?”

- The system returns: “Jane Austen.”

9. Text-to-Speech and Speech Recognition

Example:

User says: “I’m going to read a book.”

- Speech recognition converts the audio into text accurately, then semantic analysis helps TTS pronounce it naturally with correct intonation when reading back.

10. Content Moderation

Example:

User comment: “You’re stupid and worthless.”

- System detects hate speech/offensive language based on semantics and flags or removes the comment.

A<u>pplication of semantic analysis:</u>

Focusing on how different organizations use it to understand meaning, context, and intent from data:

1. Uber’s Social Listening

- **Purpose:** To monitor and understand customer opinions, feedback, and sentiment from social media platforms (like Twitter, Facebook, etc.).
- **Use of Semantic Analysis:**
    - Uber applies semantic analysis to interpret user comments — not just by keywords, but by meaning and tone**.**
    - Example: Detecting whether a post like “Uber driver was late again!” expresses a negative experience.
    - Helps Uber identify service issues, track public sentiment, and improve marketing or customer communication strategies.

2. Google’s Semantic Algorithm – Hummingbird

- **Purpose:** To deliver more accurate and context-aware search results.
- **Use of Semantic Analysis:**
    - Hummingbird focuses on understanding the intent behind search queries rather than exact keyword matches.
    - Example: For the query *“best way to cook healthy food”*, it looks for content related to healthy cooking methods — not just pages containing the words “best,” “cook,” or “food.”
    - It uses semantic relationships and entity recognition from the Knowledge Graph to interpret meaning.

3. Cdiscount’s Semantic Analysis of Customer Reviews

- **Purpose:** To improve product quality and customer experience by analyzing feedback.
- **Use of Semantic Analysis:**
    - Cdiscount, a French e-commerce company, uses semantic techniques to analyze thousands of customer reviews.

- It identifies topics**,** emotions**,** and product features customers talk about — for example, "battery life," "screen quality," or "delivery speed."
- This helps detect customer satisfaction patterns**,** emerging issues, and areas for improvement.

4. Uber's Customer Support Platform to Improve Maps

- **Purpose:** To enhance map accuracy **and** navigation efficiency**.**
- **Use of Semantic Analysis:**
  - Uber processes customer support messages and driver feedback using semantic analysis to extract location-based issues**.**
  - Example: If multiple users mention "pickup point is wrong" or "street name missing," the system semantically identifies map errors **or** incorrect locations**.**
  - These insights automatically update or flag map data for correction.

5. IBM's Watson Conversation Service

- **Purpose:** To enable intelligent, human-like conversations in applications and customer support systems.
- **Use of Semantic Analysis:**
  - IBM Watson applies semantic understanding to interpret user queries**,** analyze context, and generate meaningful responses**.**
  - Example: If a user asks, "How can I change my plan?" Watson identifies *"change"* as an action and *"plan"* as a product feature.
  - It uses semantic intent detection to trigger the correct automated process or response.

6. Conversational Chatbots – Siri, Alexa, Google Assistant

- **Purpose:** To engage users in natural, voice-based interactions and provide relevant responses.
- **Use of Semantic Analysis:**
  - These assistants use semantic analysis and natural language understanding (NLU) to interpret speech**,** understand context**,** and map meaning to actions**.**
  - Example: If you say, "Remind me to call mom when I reach home," the assistant identifies:
    - *Intent:* Set reminder
    - *Entity:* "call mom"
    - *Condition:* "when I reach home"

- This understanding allows them to perform tasks, answer questions, and hold contextual conversations.

Corpus Study in NLP (Natural Language Processing)

A corpus study in NLP refers to the collection, analysis, and use of large sets of real-world texts — called a corpus (plural: *corpora*) — to study how language is actually used.
It is one of the foundational methods in computational linguistics and NLP for developing, testing, and evaluating language models, parsers, and other NLP tools.

Definition:

A corpus is a large, structured, and electronically stored collection of authentic text or speech data used for linguistic analysis and machine learning in NLP.

Purpose of Corpus Study:

Corpus studies help machines and researchers:

1. **Understand how language is used in real contexts.**
2. **Identify patterns** in grammar, vocabulary, or semantics.
3. **Train and evaluate** NLP models (like POS taggers, translators, chatbots).
4. **Build linguistic resources** such as dictionaries, thesauri, and embeddings.

Steps in a Corpus Study:

1. **Corpus Collection:**
   Gather large text datasets (books, articles, tweets, speech transcripts, etc.).
   - Example: British National Corpus (BNC), Brown Corpus, Wikipedia dumps.
2. **Corpus Annotation:**
   Add linguistic labels like:
   - Part-of-speech tags (e.g., noun, verb)
   - Named entities (person, organization, location)
   - Syntactic structure (parse trees)
   - Semantic roles or sentiment labels
3. **Corpus Analysis:**
   Use computational or statistical methods to study patterns — word frequency, collocations, syntactic trends, etc.
4. **Model Development:**
   Train NLP models (like language models, POS taggers, etc.) using corpus data.

Example Use Cases:

- **Machine Translation:** Parallel corpora help align sentences in two languages for translation models.

- **POS Tagging & Parsing:** Annotated corpora (like Penn Treebank) train grammar-based NLP systems.
- **Sentiment Analysis:** Review corpora (e.g., IMDb dataset) help models learn emotional tone.
- **Speech Recognition:** Spoken corpora train systems like Google Voice or Siri.

WordNet in NLP (Natural Language Processing)

WordNet is a large lexical database of the English language that groups words based on their meanings and relationships — rather than just alphabetically like a dictionary.

It is widely used in semantic analysis**,** information retrieval**,** machine translation**,** and many other NLP applications.

Definition:

WordNet is a semantic network of English words that organizes nouns, verbs, adjectives, and adverbs into sets of synonyms called synsets**,** and links them through semantic and lexical relations (like synonymy, antonymy, hypernymy, etc.).

It was developed at Princeton University under the leadership of George A. Miller in the mid-1980s.

Structure of WordNet:

Each entry in WordNet is not a single word, but a Synset — a set of words that share the same meaning.

For example:

Synset: {car, automobile, motorcar}

→ "car," "automobile," and "motorcar" are **synonyms** meaning *a four-wheeled motor vehicle used for transport.*

Main Semantic Relations in WordNet:

Relation	Description	Example
Synonymy	Words with similar meanings	{big, large}
Antonymy	Words with opposite meanings	{hot, cold}
Hypernymy	"Is-a" relationship (general term)	*Animal* is a hypernym of *Dog*
Hyponymy	"Is-a-kind-of" relationship (specific term)	*Dog* is a hyponym of *Animal*
Meronymy	"Part-of" relationship	*Wheel* is a part of *Car*
Holonymy	"Has-a" relationship	*Car* has *wheel*
Troponymy	Specific manner of an action	*Jog* is a troponym of *Run*

Relation	Description	Example
Entailment	One action implies another	*Snore* entails *sleep*

Example

For the word **"car"**, WordNet might provide:

- **Synset:** {car, auto, automobile, machine, motorcar}
- **Definition (Gloss):** "A motor vehicle with four wheels; usually propelled by an internal combustion engine."
- **Hypernym:** {motor vehicle}
- **Hyponyms:** {ambulance, cab, jeep, limousine}

Applications of WordNet in NLP

1. **Word Sense Disambiguation (WSD):**

   Helps determine the correct meaning of a word based on context.
   - Example: "bank" → financial institution or riverbank?
2. **Information Retrieval:**

   Expands search queries using synonyms and related words.
   - Example: Searching "car" also finds "automobile."
3. **Semantic Similarity Measurement:**

   Measures how closely related two words are in meaning.
4. **Text Classification and Summarization:**

   Identifies semantically related terms to improve feature representation.
5. **Machine Translation:**

   Provides multiple word meanings for accurate translation choices.
6. **Ontology and Knowledge Graph Construction:**

   Forms the basis for linking concepts in knowledge systems.

<u>BabelNet in NLP (Natural Language Processing)</u>

BabelNet is a multilingual lexical and semantic network — a massive knowledge graph that connects words, concepts, and entities from hundreds of languages.

It combines information from multiple sources (like WordNet**,** Wikipedia**,** Wikidata, and others) to help computers understand meaning across languages and contexts**.**

Definition:

BabelNet **is a** wide-coverage multilingual semantic network and ontology that integrates lexical knowledge (WordNet) with encyclopedic knowledge (Wikipedia) to represent concepts and named entities in many languages.

It was developed at the Sapienza University of Rome **by** Roberto Navigli and his team.

Core Idea:

BabelNet builds on the concept of **synsets** (from WordNet) but expands them to include:

- **Multilingual word senses**
- **Named entities (people, places, organizations, etc.)**
- **Concepts from encyclopedic sources**

This allows BabelNet to represent: — making it extremely useful for cross-lingual NLP tasks**.**

Structure of BabelNet:

- The main building block is the **BabelSynset**:
    - A set of words (in multiple languages) that share the same meaning or refer to the same concept or entity.
    - Each BabelSynset contains:
        - **Words/phrases** in different languages
        - **Definitions (glosses)**
        - **Semantic relations** (synonyms, hypernyms, meronyms, etc.)
        - **Translations**
        - **Wikipedia/Wikidata links**

Example:

BabelSynset for the concept "car" might include:

- English: *car*, *automobile*
- Spanish: *coche, automóvil*
- French: *voiture*
- Hindi: *कार (kār)*
- Plus links to the **Wikipedia page** for "Car" and related concepts.

Sources Integrated in BabelNet:

BabelNet merges data from:

- **WordNet** → lexical and semantic relationships
- **Wikipedia/Wikidata** → encyclopedic and entity-based knowledge
- **OmegaWiki**, **Wiktionary**, **Open Multilingual WordNet**, and more

As of recent versions, BabelNet covers over 500 languages **and** millions of concepts**.**

Key Features

Feature	Description
Multilingual coverage	Links words and entities across 500+ languages

Feature	Description
Concept + Entity integration	Combines common nouns (like “car”) and named entities (like “Tesla”)
Semantic relations	Captures synonymy, hypernymy, meronymy, etc.
Cross-lingual mapping	Connects equivalent meanings across languages
Knowledge graph	Links concepts to encyclopedic and factual data

Applications in NLP:

Application	How BabelNet Helps
Word Sense Disambiguation (WSD)	Identifies correct meanings of ambiguous words in any language
Machine Translation	Provides aligned concepts across languages for better translation accuracy
Cross-lingual Information Retrieval	Enables searching content in one language and retrieving in another
Question Answering	Connects questions and answers across languages and domains
Semantic Similarity & Text Understanding	Measures meaning relationships between multilingual texts
Named Entity Recognition & Linking (NER/NEL)	Links text mentions to entities (like “Paris” → city or person)

Using BabelNet

Developers and researchers can:

- Access BabelNet through its **API** (requires registration).
- Use the **Python SDK** or RESTful web service for integrating with NLP tools.

BabelNet vs. WordNet:

Feature	WordNet	BabelNet
Language coverage	English only	500+ languages

Feature	WordNet	BabelNet
Data type	Lexical (word meanings)	Lexical + encyclopedic (concepts & entities)
Source	Handcrafted linguistic data	Combined from WordNet, Wikipedia, Wikidata, etc.
Focus	Semantic relationships within one language	Multilingual and cross-lingual semantics
Example	*car → automobile*	*car → coche → voiture → कार*

Sentence Fragment in English Grammar

A sentence fragment is an incomplete sentence — a group of words that looks like a sentence but does not express a complete thought or lacks an essential part**,** such as a subject**,** a verb**,** or an independent clause**.**

It leaves the reader wondering *"Who did what?"* or *"What happened?"*

Definition

A sentence fragment is a group of words that is punctuated as a sentence but does not form a complete idea or cannot stand alone grammatically.

A Complete Sentence Must Have:

1. A **subject** (who or what the sentence is about)
2. A **predicate** (verb that shows what the subject does or is)
3. A **complete thought** (can stand alone)

A sentence fragment is missing one or more of these elements.

Examples of Sentence Fragments:

Fragment	Problem	Correct Sentence
Because it was raining.	Dependent clause – doesn't express a full thought.	Because it was raining, we stayed inside.
Running through the park.	Missing subject and main verb.	She was running through the park.
The book on the table.	Missing verb – no action.	The book on the table belongs to me.
When I finished my homework.	Subordinate clause – incomplete idea.	When I finished my homework, I went to bed.

Fragment	Problem	Correct Sentence
Such a beautiful view!	No subject or verb – just a phrase.	It is such a beautiful view!

Types of Sentence Fragments:

1. **Dependent Clause Fragment**
    - Starts with a subordinating conjunction (because, although, when, if, etc.).
    - *Although he studied hard.*
    - *Although he studied hard, he failed the test.*
2. **Phrase Fragment**
    - Missing a subject or verb (noun phrase, prepositional phrase, etc.).
    - *After the long day.*
    - *After the long day, we went home.*
3. **Missing Subject Fragment**
    - No clear subject performing the action.
    - *Went to the market.*
    - *She went to the market.*
4. **Missing Verb Fragment**
    - Has a subject but no verb.
    - *The boy in the red shirt.*
    - *The boy in the red shirt is my cousin.*

How to Fix a Sentence Fragment:

1. **Add the missing subject or verb.**
2. **Attach the fragment to a nearby complete sentence.**
3. **Remove subordinating words** that make it dependent (like *because*, *when*, *if*).

Here's a clear explanation of each semantic relationship in NLP — all essential for understanding word meaning and relationships in natural language processing 👇

i. Consider the following fragment (incomplete) sentences: –

1. Because of the rain.
2. I ran.
3. Looking forward to seeing you.
4. You ok.
5. Everyone please bring to the party.

Recast the above sentences if required, so that the meaning should be preserved or completed. Assume suitable words whenever required.

Examples:

1. Fragment: When the bell rang.

Complete Sentence: When the bell rang, the students left the classroom.

2. Fragment: After finishing the homework.

Complete Sentence: After finishing the homework, I watched a movie.

3. Fragment: Walking down the street.

Complete Sentence: I saw an old friend while walking down the street.

4. Fragment: Since he was tired.

Complete Sentence: Since he was tired, he went to bed early.

5. Fragment: The book on the table.

Complete Sentence: The book on the table belongs to my sister.

Lexeme: Relations and Senses:

A lexeme is the basic unit of meaning in a language — a word in its abstract form, representing all its possible inflected or derived variants.

In simple terms: A lexeme is the "root meaning" of a word, not tied to a specific grammatical form. Lexemes are not isolated — they are connected to other lexemes through semantic relations, and they may have multiple senses (meanings).

Senses of a Lexeme:

A sense is a specific meaning of a lexeme.

Many lexemes are polysemous, meaning they have more than one sense depending on context.

Sense	Description	Example
Sense 1	Financial institution	"He deposited money in the bank."
Sense 2	River edge	"The kids played on the river bank."

Relations Between Lexemes:

Lexemes are connected to one another through semantic relations. These relationships help machines understand meaning connections in text.

1. Homonymy

Definition:

Homonymy occurs when two words have the same spelling or pronunciation but different, unrelated meanings**.**

- These words are called homonyms**.**
- In NLP, recognizing homonyms helps in Word Sense Disambiguation (WSD) — figuring out which meaning is intended.

Examples:

Word	Meaning 1	Meaning 2
Bank	Financial institution	Edge of a river
Bat	Flying mammal	Sports equipment used in cricket/baseball
Seal	Marine animal	A stamp or symbol of approval

The system must use *context* to choose the correct meaning (e.g., "He went to the bank to deposit money" → *bank* = financial institution).

2. Polysemy

Definition:

Polysemy refers to a single word that has multiple related meanings**.**

Unlike homonyms, the meanings are connected or derived from each other.

Examples:

Word	Related Meanings
Head	Part of the body / Leader of an organization / Top of something
Foot	Part of the body / Base of a mountain / Bottom of a page
Paper	Material to write on / A newspaper / An academic article

Polysemy affects semantic analysis because the system must detect which sense of the word is used depending on the context**.**

3. Synonymy

Definition:

Synonymy means different words that have the same or very similar meanings**.**

Such words are called synonyms**.**

Examples:

Word 1	Word 2	Sentence Example
Big	**Large**	She has a big/large house.
Begin	**Start**	Let's begin/start the meeting.
Purchase	**Buy**	I will purchase/buy a new phone.

Synonymy helps in semantic similarity and information retrieval — for example, a search for "buy car" should also return results for "purchase vehicle."

4. Antonymy

Definition:

Antonymy refers to words with opposite meanings**.**

Such words are called antonyms**.**

Examples:

Word	**Antonym**
Hot	Cold
Happy	Sad
Fast	Slow
True	False

Antonymy is used in sentiment analysis **and** text polarity detection (e.g., identifying that "good" vs. "bad" have opposite sentiments).

5. Hypernymy

Definition:

A hypernym is a general or broad term that represents a category of more specific words.

It answers the question: *"What kind of thing is this?"*

Examples:

Specific Word	**Hypernym**
Dog**,** Cat**,** Elephant	Animal
Rose**,** Tulip	Flower
Car**,** Bus**,** Bike	Vehicle

Hypernyms help in semantic classification and taxonomy creation (e.g., grouping "dog" under "animal" in a knowledge graph).

6. Hyponymy

Definition:

A hyponym is a specific term that falls under a broader category (the hypernym).

It's the *"kind-of"* relationship.

Examples:

Hyponym	Hypernym
Apple, Banana	Fruit
Car, Truck, Bus	Vehicle
Rose, Lily	Flower

In NLP: Hyponyms help systems understand hierarchy and concept specificity — e.g., "A dog is a kind of animal."

7. Meronymy

Definition:

A meronym denotes a part-whole relationship — when one word represents a part of another object.

If *X* is part of *Y*, then *X* is a meronym of *Y*.

Examples:

Part (Meronym)	Whole
Wheel	Car
Leaf	Tree
Chapter	Book
Engine	Airplane

In NLP: Meronymy is useful in ontology building and knowledge representation (e.g., understanding that "wheel" is part of "car").

Summary Table

Concept	Relation Type	Example	Meaning
Homonymy	Same form, unrelated meanings	*Bank* (money / river)	Different meanings, same word
Polysemy	Same form, related meanings	*Head* (body / leader)	Related meanings
Synonymy	Similar meanings	*Big – Large*	Similar meaning words
Antonymy	Opposite meanings	*Hot – Cold*	Opposite meaning words
Hypernymy	General term	*Animal → Dog*	"Is-a" (general)
Hyponymy	Specific term	*Dog → Animal*	"Is-a-kind-of" (specific)

Concept	Relation Type	Example	Meaning
Meronymy	Part-whole	*Wheel – Car*	"Part-of" relationship

Semantic Ambiguity in NLP:

Definition:

Semantic Ambiguity occurs when a sentence or phrase has more than one possible meaning because a word or phrase within it can be interpreted in multiple ways**.**

In Natural Language Processing (NLP)**,** this means the system cannot immediately determine is intended — the interpretation depends on context**.**

Types of Semantic Ambiguity:

There are mainly two types:

1. Lexical Ambiguity

- Happens when a single word has multiple meanings**.**
- The word itself is ambiguous, and context is needed to choose the right meaning.

Examples:

Sentence	Ambiguous Word	Possible Meanings
He went to the bank.	*bank*	(1) A financial institution, (2) The side of a river
She cannot bear children.	*bear*	(1) To tolerate, (2) To give birth to
The crane is flying.	*crane*	(1) A bird, (2) A construction machine

2. Structural (Syntactic) Ambiguity

- Happens when the **structure of the sentence** allows **multiple interpretations**, even if the words themselves are clear.

Examples:

Sentence	Possible Interpretations
I saw the man with the telescope.	(1) *I used a telescope to see the man.* (2) *I saw a man who had a telescope.*
Visiting relatives can be boring.	(1) *When you visit relatives, it can be boring.* (2) *Relatives who visit can be boring.*

Sentence	Possible Interpretations
She watched the man on the hill with a telescope.	(1) *The telescope is on the hill.* (2) *The man is on the hill.* (3) *She used a telescope to watch.*

How NLP Handles Semantic Ambiguity

NLP systems use several methods to resolve ambiguity:

1. **Contextual clues** – Using nearby words and sentence structure.
   e.g., "money," "deposit," "account" → *bank* means financial institution.
2. **Part-of-Speech Tagging** – Identifying grammatical roles (noun, verb, etc.).
   e.g., "bear" (noun: animal) vs. "bear" (verb: tolerate).
3. **Word Sense Disambiguation (WSD)** – Using machine learning or knowledge bases like **WordNet** to determine correct meanings.
4. **Semantic Role Labeling** – Understanding who did what to whom.
5. **Contextual Embeddings (Modern NLP)** – Models like **BERT** or **GPT** use surrounding text to infer the most likely meaning dynamically.

Example in Context

Sentence:

"He put the money in the bank."

If nearby sentences include:

"He opened a savings account last week,"

Then the system interprets *bank* as a financial institution**.**

But if the context says:

"He was fishing near the trees,"

Then *bank* means riverbank**.**

Summary:

Type	Description	Example	Ambiguity
Lexical	One word, multiple meanings	"He went to the bank."	Bank → riverbank / financial bank
Structural	Sentence structure allows multiple interpretations	"I saw the man with the telescope."	Who has the telescope?

Word Sense Disambiguation (WSD):

1. Definition

Word Sense Disambiguation (WSD) **is the** process of identifying which meaning (or sense) of a word is being used in a given context.

Many words in English (and other languages) are polysemous**,** meaning they have multiple meanings depending on how they're used.

Example:

The word **"bank"** can refer to:

- A financial institution – "I deposited money in the bank."
- The side of a river – "The boat went down the river bank."

Here, WSD helps the system decide which sense of "bank" is correct depending on the context**.**

2. Objective:

The goal of WSD is to disambiguate — that is, to select the correct sense of an ambiguous word when it appears in a sentence.

This is crucial for many NLP tasks such as:

- **Machine Translation** (translating the right meaning),
- **Information Retrieval** (retrieving correct documents),
- **Question Answering**, and
- **Text Understanding**.

3. Example Explanation:

Let's consider the example again:

Sentence	Possible Sense of "bank"	Correct Sense (after WSD)
I deposited money in the bank.	(1) Financial institution	Financial institution
The boat went down the river bank.	(2) Edge of a river	Edge of a river

The context words like *money*, *deposited* in the first sentence, and *river*, *boat* in the second, guide the disambiguation process.

4. Common Approaches to WSD:

a. Knowledge-based approach (using WordNet)

- Uses lexical databases like WordNet, which defines all possible senses of a word and their relationships.
- Algorithms such as Lesk's algorithm compare the overlap between the dictionary definitions (glosses) of the ambiguous word and its context words.

Example: Overlap of definition words between “bank (financial)” and context “money, deposit” helps choose the correct sense.

b. Supervised Machine Learning approach

- Treats WSD as a classification problem**.**
- A labeled dataset is used where each word occurrence is tagged with the correct sense.
- Features used include:
    - Neighboring words (context),
    - Part of speech (POS),
    - Collocations (common word pairs).

Example algorithms: Naïve Bayes**,** Decision Trees**,** Maximum Entropy**,** Neural Networks**.**

c. Unsupervised (Clustering) approach

- No labeled data is used.
- The algorithm groups (clusters) instances of a word into clusters based on similarity of context**.**
- Each cluster is assumed to represent a different sense**.**

Example: Instances of “bank” used near words like *money, loan* go to one cluster, and those near *river, water* go to another.

Knowledge-Based Methods:

These methods rely on pre-existing lexical resources like WordNet**,** dictionaries, or thesauri. They don’t need annotated data — instead, they use definitions, semantic relations, and structure of these resources.

(a) Lesk’s Algorithm

- **Idea:** The correct sense of a word is the one whose dictionary definition (gloss) shares the maximum number of words in common with the context in which the word appears.
- **Example:**
    - Sentence: “He went to the *bank* to deposit money.”
    - Possible senses:
        1. *Bank* (financial institution)
        2. *Bank* (river edge)
    - Lesk algorithm compares glosses (dictionary definitions) with the surrounding words (“deposit”, “money”) → more overlap with sense 1, so that is chosen.

- **Limitations:** Works best with rich glosses; performance depends on overlap quality and definition wording.

(b) Dictionary and Thesaurus-Based Methods

- Use structured lexical databases (like WordNet) and semantic relations such as:
  - **Synonymy** (similar meaning)
  - **Hypernymy** (is-a relation)
  - **Hyponymy** (subtype relation)
  - **Meronymy** (part-whole relation)
- The sense is chosen based on semantic similarity between the target word and its context words using these relations.
- **Example:** If "river" appears near "bank," and "bank" has a sense related to rivers, the algorithm selects that sense using thesaurus relations.

(c) Graph-Based Algorithms

- Words and their possible senses are represented as nodes in a graph**.**
- Edges represent semantic similarity or relations between senses (from WordNet or other sources).
- Algorithms like PageRank**,** HITS**,** or random walks identify the most central or connected sense.
- **Example:**

  For the word "bass" (fish vs. instrument), a graph of related words (e.g., "music," "river," "guitar") helps determine the correct sense based on context connections.

2. Machine Learning-Based Methods

These rely on statistical models trained on data, rather than external knowledge bases. They use features extracted from the context (neighboring words, POS tags, etc.).

(a) **Supervised Methods:**

- **Require:** A labeled corpus where each word occurrence is tagged with its correct sense.
- **Models:** Naïve Bayes, Decision Trees, SVM, Neural Networks.
- **Process:** Train a classifier to learn which context features predict each sense.
- **Example:** For the word "plant," a model learns that "factory" predicts the "industrial" sense, and "leaf" predicts the "living organism" sense.
- **Limitation:** Requires large, sense-tagged data — costly and time-consuming to create.

(b) **Semi-Supervised Methods**

- Use a small amount of labeled data and a large amount of unlabeled data**.**
- The labeled data "seeds" the model, and unlabeled data is used to iteratively refine predictions.
- **Techniques:** Bootstrapping, self-training, co-training.
- **Advantage:** Reduces dependence on costly labeled datasets.

(c) Unsupervised Methods

- Do not use labeled data.
- Assume that different senses form natural clusters in the feature space.
- Use clustering algorithms (like K-means, EM algorithm) on context words.
- **Example:** The word "crane" may appear in two types of contexts — with "construction," "machine," or with "bird," "fly." These clusters correspond to two different senses.
- **Limitation:** It identifies *different usages*, but not necessarily *link them to dictionary senses*.

(d) **Semi-Supervised Learning (Concept Explanation)**

- Lies between supervised and unsupervised learning.
- Uses a few labeled examples to guide learning from a large unlabeled corpus**.**
- Common in WSD because sense-labeled data is limited.
- Example: Label a few instances of "bank," and the model generalizes using unlabeled sentences to classify others automatically.

3. Hybrid Methods

- Combine knowledge-based and machine learning approaches to use the strengths of both.
- **Example:**
    - Use WordNet for semantic similarity (knowledge-based).
    - Feed those similarity scores as features into a supervised or neural model (machine learning).
- **Advantages:**
    - Better coverage of senses (from lexical resources).
    - Improved accuracy (from learning from data).
- **Example Framework:** A neural WSD system using pre-trained word embeddings (data-driven) + WordNet sense relations (knowledge-driven).

Knowledge-Based Methods (Lesk's Algorithm)

The Lesk Algorithm is one of the earliest and most well-known knowledge-based methods for Word Sense Disambiguation (WSD)**.**

It uses dictionary definitions (called glosses) of words to determine the most appropriate sense of an ambiguous word based on overlapping words in their definitions.

Working Principle:

1. For each possible sense of the ambiguous word, take its definition (gloss) from a lexical resource such as WordNet**.**
2. Also take the glosses of the context words (the words surrounding the ambiguous word).
3. Measure the overlap between the words in these glosses (i.e., count how many words are common).
4. The sense with the highest overlap is chosen as the correct meaning.

Example:

Sentence:

"He went to the bank to deposit his money."

Step 1: Identify ambiguous word

- Ambiguous word: bank
- Possible senses (from WordNet or dictionary):
    1. $Bank_1$: Financial institution where money is kept.
    2. $Bank_2$: The land beside a river.

Step 2: Collect glosses

- Gloss of $Bank_1$: "A financial institution that accepts deposits and lends money."
- Gloss of $Bank_2$: "The sloping land beside a body of water."

Context words: *went, deposit, money*

- Glosses:
    - Deposit: "To put money into a bank."
    - Money: "Medium of exchange used for buying and selling."

Step 3: Find overlaps

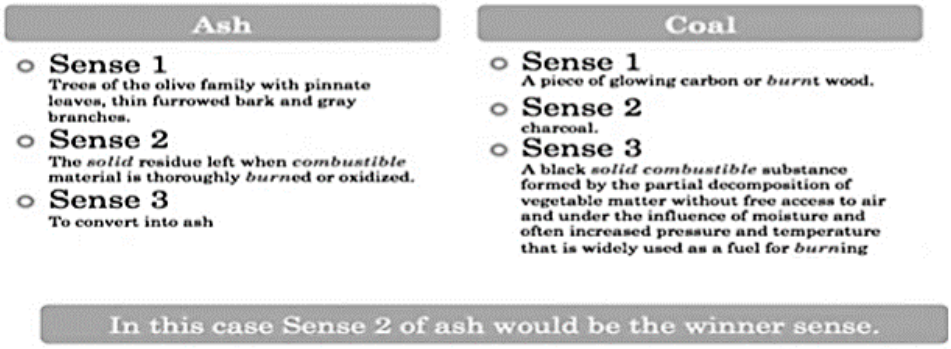
- For $Bank_1$: Words like *money, deposit, financial, institution* appear in both glosses. → High overlap.
- For **$Bank_2$:** Almost no overlap with *money* or *deposit*. → Low overlap.

Step 4: Choose sense with maximum overlap

→ $Bank_1$ (financial institution) is selected as the correct meaning.

Example 1:

*On burning **coal** we get ash.*



Example 2:

*"He sat on the **river** bank."*

Dictionary Definitions (simplified):
Bank (n1): "The land alongside or sloping down to a river or lake."
Bank (n2): "A financial institution that accepts deposits and channels the money into lending activities."

Context: "He sat on the river bank."

Overlap Calculation:
n1: Overlap words might include "river", "land".
n2: No overlap.
The sense "n1" (the land alongside or sloping down to a river or lake) will be selected because it has a higher overlap with the context words

Advantages of Lesk's Algorithm

1. **Simple and Intuitive:**
    - Easy to understand and implement — it relies on word overlap between dictionary definitions.
2. **No Training Data Needed:**
    - It is a knowledge-based method, so it doesn't require manually annotated corpora or machine learning models.
3. **Domain-Independent:**
    - Works for any text as long as dictionary or lexical resources (like WordNet) are available.
4. **Uses Existing Lexical Knowledge:**
    - Makes effective use of dictionary definitions, thesauri, or WordNet glosses.
5. **Language Flexibility:**
    - Can be adapted for any language that has a structured dictionary or lexical database.

Limitations of Lesk's Algorithm

1. **Dependence on Dictionary Quality:**
    - The performance heavily depends on how detailed and rich the dictionary or WordNet glosses are.
2. **Limited Overlap:**
    - Often, there is little or no overlap between glosses, leading to poor disambiguation.
3. **Ignores Word Order and Syntax:**
    - It only counts overlapping words, not their position or grammatical relationships.
4. **Computationally Expensive (Extended Lesk):**
    - Comparing glosses for all possible sense combinations can become time-consuming for long sentences.
5. **Ambiguity in Context Words:**
    - The context words themselves might be ambiguous, which can lead to inaccurate overlaps.
6. **Lack of Semantic Depth:**
    - The algorithm uses surface word overlap, not deeper semantic similarity (e.g., "money" and "cash" might be related but don't overlap literally).

Supervised Method (Naïve Bayes Method):

The Naïve Bayes classifier is a probabilistic model based on Bayes' Theorem**,** which assumes that the features (context words) are independent of each other given the word sense.

In WSD, it predicts the most likely sense of an ambiguous word based on the context in which it appears.

Bayes' Theorem Formula:

$$P(S_i|C) = \frac{P(C|S_i) \cdot P(S_i)}{P(C)}$$

Where:

- S_i = possible sense of the word
- C = context (neighboring words, POS tags, etc.)
- $P(S_i|C)$ = probability that the sense is S_i given the context
- $P(C|S_i)$ = probability of observing the context given the sense
- $P(S_i)$ = prior probability of that sense
- $P(C)$ = probability of the context (same for all senses, so ignored in comparison)

Algorithm Steps:

1. **Collect training data:**

   A labeled corpus where each occurrence of the ambiguous word is tagged with its correct sense.

2. **Extract features:**

   For each instance, extract features such as:

   - Surrounding words (context words)
   - Part of Speech (POS) tags
   - Collocations or syntactic patterns

3. **Calculate Probabilities:**

   - P(Si)P(S_i)P(Si): how often each sense occurs (prior probability).
   - P(C|Si)P(C|S_i)P(C|Si): how likely the context words appear with that sense.

4. **Apply Naïve Independence Assumption:**

   Assume that all context words are independent:

   P(C|Si)=∏j=1nP(wj|Si)P(C|S_i) = \prod_{j=1}^{n} P(w_j | S_i)P(C|Si)=j=1∏nP(wj|Si)

5. **Compute P(Si|C)P(S_i | C)P(Si|C)** for all senses, and choose the sense with the highest probability.

Example 1:

Sentence:

“He went to the bank to deposit money.”

Step 1: Identify ambiguous word

- Word: *bank*
- Possible senses:
   1. **Bank₁:** Financial institution
   2. **Bank₂:** Land beside a river

Step 2: Training data (simplified)

From a corpus, we learn how often context words occur with each sense:

Context Word	P(word | Bank₁)	P(word | Bank₂)
money	0.5	0.01
deposit	0.4	0.02

| river | 0.05 | 0.45 |

| water | 0.05 | 0.52 |

Also, prior probabilities:

$P(Bank_1) = 0.6$, $(Bank_2) = 0.4$

Step 3: Context words: *deposit, money*

Compute probabilities:

For **Bank₁**:

$$P(Bank_1|context) \propto P(Bank_1) \times P(deposit|Bank_1) \times P(money|Bank_1)$$

$$= 0.6 \times 0.4 \times 0.5 = 0.12$$

For **Bank₂**:

$$P(Bank_2|context) \propto P(Bank_2) \times P(deposit|Bank_2) \times P(money|Bank_2)$$

$$= 0.4 \times 0.02 \times 0.01 = 0.00008$$

Result:

$$P(Bank_1|context) > P(Bank_2|context)$$

→ **Predicted sense:** *Bank₁ (financial institution)*

Example 2:

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w|c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	fish smoked fish	f
	2	fish line	f
	3	fish haul smoked	f
	4	guitar jazz line	g
Test	5	line guitar jazz jazz	?

V = {fish, smoked, line, haul, guitar, jazz}

Priors:

$P(f) = \frac{3}{4}$

$P(g) = \frac{1}{4}$

Conditional Probabilities:

P(line|f) = (1+1) / (8+6) = 2/14

P(guitar|f) = (0+1) / (8+6) = 1/14

P(jazz|f) = (0+1) / (8+6) = 1/14

P(line|g) = (1+1) / (3+6) = 2/9

P(guitar|g) = (1+1) / (3+6) = 2/9

P(jazz|g) = (1+1) / (3+6) = 2/9

Choosing a class:

P(f|d5) ∝ 3/4 * 2/14 * (1/14)² * 1/14

≈ 0.00003

$(f|d5) \propto (p|f) \times (line|f) \times (guitar|f) \times (jazz|f) \times (jazz|f)$

P(g|d5) ∝ 1/4 * 2/9 * (2/9)² * 2/9

≈ 0.0006

$(g|d5) \propto (p|g) \times (line|g) \times (guitar|g) \times (jazz|g) \times (jazz|g)$

∴ Resultant class is **g**

Example 3:

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Conditional Probabilities:

$P(\text{Chinese}|c) = (5+1) / (8+6) = 6/14 = 3/7$

$P(\text{Tokyo}|c) = (0+1) / (8+6) = 1/14$

$P(\text{Japan}|c) = (0+1) / (8+6) = 1/14$

$P(\text{Chinese}|j) = (1+1) / (3+6) = 2/9$

$P(\text{Tokyo}|j) = (1+1) / (3+6) = 2/9$

$P(\text{Japan}|j) = (1+1) / (3+6) = 2/9$

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

$\therefore$ Resultant class is **c**

Advantages:

Advantage	Description
Simple & Efficient	Easy to implement, fast even on large datasets
Performs Well in Practice	Despite independence assumption, often yields good accuracy
Statistical & Objective	Learns directly from data, not dependent on dictionary glosses
Handles Multiple Features	Can combine word, POS, and collocation features

Limitations:

Limitation	Description
Needs Labeled Data	Requires a large annotated corpus where word senses are tagged
Independence Assumption	Assumes features (context words) are independent, which is often false in language
Sparse Data Problem	Some word-context combinations may not appear in training data
Domain Dependence	Works best when training and test data are from the same domain

Supervised Method (Decision List):

The Decision List approach is a supervised learning method for Word Sense Disambiguation (WSD) proposed by Yarowsky (1994)**.**

It works by learning a ranked list of rules (features) that can be used to determine the most likely sense of an ambiguous word based on its context**.**

Basic Idea:

A decision list is an ordered list of "if–then" rules that map specific contextual features (such as surrounding words, collocations, or part-of-speech tags) to the most probable sense of the target word.

When an ambiguous word is encountered:

1. The algorithm scans the list from top to bottom.
2. The first rule whose condition matches the context is used.
3. The corresponding sense is assigned to the word.

Step-by-Step Process:

Step 1: Training Data

Start with a sense-tagged corpus**,** where each occurrence of the ambiguous word is labeled with its correct sense.

Example word: "bank"

Possible senses:

- **Sense 1:** Financial institution
- **Sense 2:** River side

Step 2: Feature Extraction

From the training corpus, extract contextual features**,** such as:

- Words immediately before or after the target word (collocations)
- Part-of-speech tags
- Surrounding window of words (e.g., ±2 words)

Example training sentences:

1. "He deposited money in the **bank**." → Sense 1 (financial)
2. "The fisherman sat on the **bank** of the river." → Sense 2 (river side)
3. "She withdrew cash from the **bank**." → Sense 1
4. "They camped near the **bank** of the lake." → Sense 2

Extracted features:

Context Word	Sense
deposit	Financial
money	Financial
cash	Financial
river	River side
lake	River side
fisherman	River side

Step 3: Rule Formation (Decision List Construction)

Each feature is associated with the most likely sense based on its frequency in training data. For each feature, calculate a log-likelihood score that measures how strongly it indicates a particular sense.

$$Score(feature) = \log_2 \left(\frac{P(sense_1|feature)}{P(sense_2|feature)} \right)$$

Then, order the features by their scores (absolute value) — higher scores come first.

Example Decision List:

Rule (Feature)	Assigned Sense	Score
If context contains “money” → Sense 1 (Financial)	Financial	3.5
If context contains “deposit” → Sense 1 (Financial)	Financial	3.1
If context contains “river” → Sense 2 (River side)	River	3.0
If context contains “lake” → Sense 2 (River side)	River	2.8
If context contains “cash” → Sense 1 (Financial)	Financial	2.7

Step 4: Disambiguation (Testing Phase)

Now, given a new sentence**:**

“He kept his savings in the bank.”

The algorithm checks the decision list:

- “money” → not present
- “deposit” → not present
- “river” → not present
- “lake” → not present
- “cash” → not present

- But maybe “savings” → often appears with “money,” so it may infer Sense 1 (Financial institution)**.**

The algorithm assigns: Bank = Financial Institution

Example 1:

Let’s disambiguate the word "bank" (it can mean river bank or financial bank).

Training Data:

Sentence	Sense of “bank”
He deposited money in the bank.	Financial
The river overflowed the bank.	River
She went to the bank to withdraw cash.	Financial
They picnicked on the bank of the river.	River

Test Data:

“He withdrew cash from the bank.”

Feature Extraction:

Feature (word near “bank”)	Sense	Frequency
“money”	Financial	1
“river”	River	2
“deposit”	Financial	1
“withdraw”	Financial	1
“picnic”	River	1

Compute Log-likelihood Ratio:

$$Conditional\ Probability_{(money|financial)} = P(money|financial) = \frac{count(money|financial)}{count(financial)}$$

$$Conditional\ Probability_{(money|river)} = P(money|river) = \frac{count(money|river)}{count(river)}$$

$$Score\ (log-likelihood)_{money} = \log_2 \frac{P(money|financial)}{P(money|river)}$$

Laplace Smoothing

If count is zero:

$$\hat{P}(w|c) = \frac{count(w,c)+1}{count(c)+|V|}$$

V=17

Feature	Sense	Freq.	Conditional Probability (Financial)	Conditional Probability (River)	Score (log-likelihood)
“money”	Financial	1	1/3	(0+1)/(2+17)	
“withdraw”	Financial	2	2/3	(0+1)/(2+17)	
“deposit”	Financial	1	1/3	(0+1)/(2+17)	
“river”	River	1	(0+1)/(3+17)	1/3	
“picnic”	River	1	(0+1)/(3+17)	1/3	

Create the Decision List:

Rank	Feature	Sense	Score (log-likelihood)
1	“money”	Financial	
2	“withdraw”	Financial	
3	“deposit”	Financial	
4	“river”	River	
5	“picnic”	River	

Test:

“He withdrew money from the bank.”

Feature found: **“money”**

If score(cash → Financial) = 3.5

→ Predict **Financial sense**

Example 2:

Let's disambiguate the word "bat" (Sense 1: Flying mammal, Sense 2: Sports equipment).

Training Data:

Sentence	Sense of "bat"
The bat flew out of the cave.	Flying mammal
He swung the bat and hit the ball.	Sports equipment
Bats use echolocation to find insects.	Flying mammal
She bought a new bat for the match.	Sports equipment
The bat hung upside down.	Flying mammal
He kept his cricket bat in the bag.	Sports equipment

Test Data:

"He hit the ball with the bat."

"The bat flew into the dark cave."

Feature Extraction:

Feature	S_1: Flying mammal	S_2: Sports equipment	Total
"flew"	2	0	2
"cave"	1	0	1
"insects"	1	0	1
"cricket"	0	2	2
"match"	0	1	1
"ball"	0	1	1

Compute Log-likelihood Scores:

Feature	Calculation	Score	Assigned Sense
flew	$\log_2(1 / 0.0001)$*	≈ +13.3	Flying mammal
cave	$\log_2(1 / 0.0001)$*	≈ +13.3	Flying mammal
insects	$\log_2(1 / 0.0001)$*	≈ +13.3	Flying mammal
cricket	$\log_2(0.0001 / 1)$*	≈ -13.3	Sports equipment
match	$\log_2(0.0001 / 1)$*	≈ -13.3	Sports equipment
ball	$\log_2(0.0001 / 1)$*	≈ -13.3	Sports equipment

* (We use a tiny value like 0.0001 to avoid division by zero.)

Create the Decision List:

Rank	Feature	Sense	Score
1	"flew"	Flying mammal	+13.3
2	"cave"	Flying mammal	+13.3
3	"insects"	Flying mammal	+13.3
4	"cricket"	Sports equipment	-13.3
5	"match"	Sports equipment	-13.3
6	"ball"	Sports equipment	-13.3

Testing:

"He hit the ball with the bat."
Context features: **"ball"**, **"hit"**
"ball" → found in Decision List → sense = **Sports equipment**
Predicted sense: *bat = Sports equipment*

"The bat flew into the dark cave."
Features: **"flew"**, **"cave"**
Both → Flying mammal
Predicted sense: *bat = Flying mammal*

Example 3:

Training Data

Sense	Training Examples (Keyword in Context)
A	used to strain microscopic plant life from the ...
A	zonal distribution of plant life ...
A	close-up studies of plant life and natural ...
A	too rapid growth of aquatic plant life in water ...
B	computer manufacturing plant and adjacent ...
B	discovered at a St. Louis plant manufacturing ...
B	copper manufacturing plant found that they ...
B	copper wire manufacturing plant, for example ...
B	cement manufacturing plant in Alpena ...

Resultant Decision List

Log L	Collocation	Sense
10.12	plant growth	A
9.68	car plant	B
9.64	plant height	A
9.61	union plant	B
9.54	equipment plant	B
9.50	assembly plant	B
9.41	nuclear plant	B
9.34	flower plant	A
9.23	Job plant	B
9.02	fruit plant	A

Sense A: biological meaning, Sense B: industrial/manufacturing meaning, Log L: Indicates the log-likelihood score.
Thus, during disambiguation, if a word like "growth" or "flower" appears near "plant," Sense A is chosen; if "assembly" or "nuclear" appears, Sense B is selected.

Key Properties:

Property	Description
Learning Type	Supervised
Knowledge Source	Sense-tagged corpus
Model Type	Rule-based list (ordered by reliability)
Output	Most probable sense according to matched rule

Advantages:

1. **Interpretable:**
   - Produces human-readable rules (easy to understand and analyze).
2. **Efficient:**
   - Quick to apply at runtime (linear scan of rules).

3. **High Accuracy:**
    - Performs well when features strongly correlate with senses.
4. **Domain-Specific Adaptability:**
    - Can adapt easily to particular domains by training on relevant data.

Limitations:

1. **Requires Labeled Data:**
    - Needs a large, sense-tagged corpus for training (costly to create).
2. **Limited Generalization:**
    - May fail if the test context differs significantly from the training data.
3. **Feature Dependence:**
    - Effectiveness depends on how well the features capture contextual clues.
4. **Data Sparsity:**
    - Rare features or senses may not be represented enough to form reliable rules.

Difficulties in Word Sense Disambiguation (WSD):

1. Different Text-Corpus or Dictionary:

- One issue with word sense disambiguation is determining what the senses are because different dictionaries and the sources divide words into distinct senses.
- Some academics have proposed employing a specific lexicon and its set of senses to address this problem. In general, however, research findings based on broad sense distinctions have outperformed those based on limited ones.
- The majority of researchers are still working on fine-grained WSD.

2. Part-of-speech tagging:

- POS tagging and sense tagging have been shown to be very tightly coupled in any real test, with each potentially constraining the other.
- Both disambiguating and tagging with words are involved in WSM POS tagging.
- Sometimes, algorithms designed for one do not always work well for the other, owing to the fact that a word's part of speech is mostly decided by the one to three words immediately adjacent to it, whereas a word's sense can be determined by words further away.

Sense Inventories:

- These are the collection of abbreviations and acronyms with their possible senses.
- Some of the examples used in Word Sense Disambiguation are:

- *Princeton WordNet:* is a vast lexicographic database of English and other languages that is manually curated. For WSD, this is the de facto standard inventory.
- *BabelNet:* is a multilingual dictionary that covers both lexicographic and encyclopedic terminology.
- *Wiktionary:* a collaborative project aimed at creating a dictionary for each language separately, is another inventory that has recently gained popularity.

****