

18CSE359T Natural Language Processing

- Offered to Final Year B.Tech. CSE
- By Dept. of C.Tech.

Unit 3 Topics

- Semantic Relations
- Semantic Role Labeling
- Semantic Frames
- Ontology and Semantics
- Semantic Network and Knowledge Graph
- Intent Detection and Classification
- Paraphrase Extraction
- Discourse
- Coreference Resolution
- Text Coherence
- Discourse Structure
- Coherence
- Discourse Planning

Semantic Relations

Semantic Relations

- Semantic relationships are the **associations** that there exist between the meanings of words (**semantic relationships at word level**), between the meanings of phrases, or between the meanings of sentences (**semantic relationships at phrase or sentence level**). Following is a description of such relationships.
- **Types of Semantic Relationships:**
 - Semantic Relationships at Word Level
 - Semantic Relationships at Phrase or Sentence Level

Semantic Relations

- At word level, we will study semantic relationships like the following:
 - **Synonymy**
 - **Antonymy**
 - **Homonymy**
 - **Polysemy**
 - **Metonymy**

Synonymy

- Synonymy is the semantic relationship that **exists between two (or more) words** that have the **same (or nearly the same) meaning** and belong to the **same part of speech**, but are **spelled differently**.
- In other words, we can say that synonymy is the **semantic equivalence between lexical items**. The (pairs of) words that have this kind of semantic relationship are called **synonyms**, or are said to be synonymous.

E.g - Synonymy

- big = large
- hide = conceal
- small = little
- couch = sofa
- to begin = to start
- kind = courteous
- beginning = start
- fast = quickly = rapidly
- Pairs of words that are synonymous are believed to **share all (or almost all) their semantic features or properties.**

Synonymy

- However, no two words have exactly the same meaning in all the contexts in which they can occur.
- For example, the verbs **employ** and **use** are synonymous in the expression: *We used/employed effective strategies to solve the problem.*
- However, only **use** can be used in the following sentence: **We used a jimmy bar to open the door.** If we used employ, the sentence would sound awkward ***We employed a jimmy bar to open the door.**
- In short, we can say that there are **no absolute synonyms**, i.e., pairs of words that have the same meaning (or share the same semantic features) in all the situational and syntactic contexts in which they can appear.

Antonymy

- Antonymy is the **semantic relationship** that exists between two (or more) words that have **opposite meanings**.
- The pairs of words which have **opposite meanings** are called **antonyms**.
- Antonymous pairs of words usually belong to the **same grammatical category** (i.e., both elements are nouns, or both are adjectives, or both are verbs, and so on).
- They are said to share almost all their semantic features except one.

Antonymy

- **Adjective Antonyms: Adjective Pairs: "Tall" and "Short"**
 - These words are antonyms.
 - Both belong to the **grammatical category** of adjectives.
 - They share semantic features **related to height** (e.g., describing a person's stature).
 - The distinguishing feature is the degree of height, where "tall" implies a greater height, and "short" implies a lesser height.
- This example illustrates the antonymous relationship between adjectives "tall" and "short" based on their shared semantic features and the **differing feature of height.**

Types of Antonymy

- **Complementary antonyms**, also known as **relational antonyms** or **paired antonyms**, are **pairs of words that together cover all possible options** or points on a spectrum within a specific semantic field.
- In other words, they are opposite words that **create a complete set of contrasting terms** within a particular context.
- Complementary antonyms do not necessarily imply **absolute opposition**
- Example is "parent" and "child." "Parent" refers to an adult who has offspring, while "child" refers to the offspring of a parent. These words together **create a complete set of relationships within the context of family dynamics.**
- consider the antonyms "on" and "off." These two words represent opposite states within the context of a switch or a light. "On" represents the state when something is activated or functioning, while "off" represents the state when something is deactivated or not functioning. Together, they **cover all possible states of the switch or light, making them complementary antonyms.**

Types of Antonymy

- **Absolute opposition** refers to a situation where two things or concepts are in complete and direct contradiction to each other, **with no room for any intermediate states, gradations, or compromises.**
- It represents a **unambiguous and clear-cut contrast between two opposing ideas, qualities, or entities.** In cases of absolute opposition, there is **no middle ground or overlap between the opposing elements.**
- For example, in the context of life and death, "alive" and "dead" are in absolute opposition. **If something is alive, it is not dead,** and vice versa. There is **no intermediate state between being alive and being dead.**
- Absolute opposition is often used to emphasize the sharp distinction between two concepts, and it is a fundamental concept in logic and philosophy. It is in contrast to situations where there may be shades of gray or intermediate possibilities between two contrasting ideas, which are not considered examples of absolute opposition.

Types of Antonymy

3) Gradable or Scalar Antonyms:

- Gradable or scalar antonyms are pairs of words that contrast with respect to the **degree or range of a particular semantic property**.
- Each term represents an endpoint or extreme on a scale, and **there can be intermediate points between them**, allowing for a middle ground.
- **Examples** include hot/cold, big/small, tall/short, good/bad, strong/weak, beautiful/ugly, happy/sad, fast/slow.

Antonymy

Antonyms can also be categorized based on their morphological relationship:

a. Morphologically Unrelated Antonyms

- Morphologically unrelated antonyms are **pairs of words** where the words have **different linguistic roots** and do not share a common morphological structure.
- These antonyms are **not formed by adding prefixes, suffixes, or changes in word structure**.
- Instead, they have **distinct origins** and are typically **unrelated in terms of etymology**.
- Examples include "happy" and "sad," "day" and "night," or "fast" and "slow."
- These pairs of words do not share linguistic elements in their formation.

Antonymy

Antonyms can also be categorized based on their morphological relationship:

b. Morphologically Related Antonyms

- Morphologically related antonyms are pairs of words where the words have a **common linguistic root** and often **involve the addition of prefixes, suffixes, or changes in word structure to create the opposite meaning.**
- These antonyms are formed through **morphological processes**, where a word's structure is altered to convey the opposite meaning.
- Common prefixes used to create morphologically related antonyms include "un-" (e.g., "happy" and "unhappy") and "dis-" (e.g., "order" and "disorder").
- Examples include "possible" and "impossible," "like" and "dislike," or "do" and "undo." In these pairs, one word is created by modifying the other through morphological changes.

Antonymy

Morphologically related antonyms can be formed in the following ways:

b.1. By using the **word not**; e.g., alive/not alive, happy/not happy, beautiful/not beautiful.

b.2. By adding **negative prefixes** such as un-, im-, in- il-, ir-, non-, mis-, dis-, a-.

E.g., happy/unhappy, do/undo, lock/unlock, entity/nonentity, conformist /nonconformist, tolerant/intolerant, decent/indecent, please/displease, like /dislike, behave/mishave, hear/mishear, moral/amoral, political/apolitical, legal/illegal, logical/illogical, probable/improbable, relevant/irrelevant.

b.3. By adding **negative suffixes** such as –less. E.g., careful/careless, joyful/joyless

Homonymy

- Homonymy is the relationship that exists between two (or more) words which belong to the **same grammatical category, have the same spelling**, may or may not have the same pronunciation, **but have different meanings and origins (semantically unrelated)**.
- **E.g.,**
 - to lie (= to rest, be, remain, be situated in a certain position) and to lie (= not to tell the truth);
 - “Bat” -> bat can be an implement to hit a ball and bat is a nocturnal flying mammal also.
 - bank (= the ground near a river) and bank (= financial institution);
 - lead [li...d] (= the first place or position, an example behavior for others to copy) and lead [led] (= heavy metal);
 - bass [beɪs] (= musical instrument) and bass [beɪs] (= edible fish). The pairs of words that exhibit this kind of relationship are called homonyms

Homonymy

- In isolated spoken sentences, homophonic homonyms can also give rise to **lexical ambiguity**.
- For example, in the following sentences **it is almost impossible** to know the intended meanings of bank and bear.
- Notice the following sentences:
 - **John went to the bank (the financial institution or the ground by the river?)**
 - **Mary can't bear (have or tolerate?) children.**

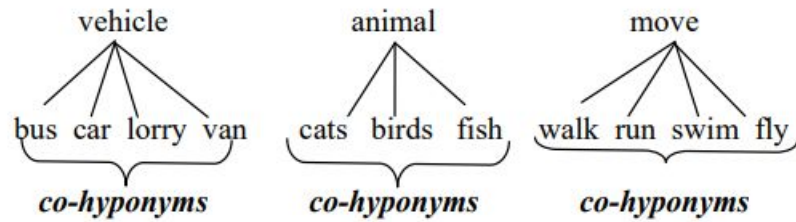
Hyponymy

- Hyponymy is a way of describing **how certain words are related to one another in terms of their meanings.**
- We say that the term **whose meaning is included in the meaning of the other term(s) is the general term;**
- Linguists refer to it as a **superordinate or hypernym.**
- Linguists usually refer to it as a hyponym If the meaning of a **superordinate term** is included in the meaning of several other more specific words, the set of specific terms which are hyponyms of the same superordinate term and are called **cohyponyms.**

Hyponymy

Superordinate:

Hyponyms



Polysemy

- Polysemy is a semantic relationship that occurs **when a single word has multiple meanings or senses** that are conceptually and historically related.
- These different meanings often share a **common thread or origin** but have evolved into distinct but related senses over time.
- "Foot" can mean both a part of the body and the lower part of something (like the base of a mountain). These meanings are conceptually related because they both involve a lower portion or support.
- "Plain" can refer to something being clear (e.g., plain water), unadorned (e.g., plain dress), or obvious (e.g., plain fact). These meanings are conceptually related in the sense of simplicity or clarity.
- "Nice" can mean pleasant, kind, or friendly. These meanings are conceptually related in the sense of positive social interactions or qualities

Metonymy

- Metonymy is a figure of speech in which one word or phrase is substituted with another with which it is closely associated.
- Unlike synonyms, metonyms do not share similar meanings but are related by context or association.
- Examples:
- "The White House announced a new policy." (where "The White House" represents the President or the administration)
- "The pen is mightier than the sword." (where "pen" represents written words and "sword" represents military force)
- "Hollywood is releasing a new movie." (where "Hollywood" represents the American film industry)

Metonymy ...

- Metonymy involves a shift in meaning based on a close relationship, often part-to-whole, cause-to-effect, or container-to-contained relationships. This relationship can be based on physical proximity, functionality, or symbolic connections.
- Key Points:
- It is different from metaphor, which draws a comparison between two unrelated things.
- It enhances language by providing a more vivid or concrete description, often adding layers of meaning to the communication.

Semantic Relationships at Phrase/Sentence Level

- A paraphrase is a restatement or rewording of a text or statement, often with the goal of **conveying the original message** or idea in different words or language while preserving its meaning.
- Paraphrase involves a relation of semantic equivalence between syntactically different phrases or sentences.
- For instance:
 - Original Sentence 1: "John wrote a letter to Mary."
 - Original Sentence 2: "A dog bit John."
 - Paraphrased Sentence 1: "John wrote Mary a letter."
 - Paraphrased Sentence 2: "John was bitten by a dog."

Semantic Relationships at Phrase or Sentence Level

- Like synonymy, paraphrase is never perfect; **there are always differences in emphasis or focus**. There are two kinds of paraphrase:
 - **1. Lexical paraphrase**. It is the use of a **semantically equivalent term** in place of another in a given context. This is also known as synonymy.
 - E.g., John is happy. = John is cheerful; She is a talented musician = She is a skilled instrumentalist
 - **2. Structural paraphrase**. It is the **use** of a **phrase or sentence** in place **of another phrase or sentence semantically equivalent** to it, although they have different syntactic structure. E.g., John showed the pictures to me. John showed me the pictures.

Semantic Role Labeling

Semantic Role Labeling

- Semantic Role Labeling (SRL) is an essential natural language processing task where the objective is to **identify the semantic roles or arguments linked to a predicate, typically a verb**, within a sentence.
- The goal of SRL is to provide a more structured representation of the meaning of a sentence by labeling the constituents of the sentence with their semantic roles, such as agent, patient, or instrument.
- This process enhances the understanding of **how different elements in a sentence relate to one another**, enabling computers to better grasp the meaning of natural language text. <https://demo.allennlp.org/semantic-role-labeling>

Semantic Role Labeling

- In linguistics, a semantic predicate, also known as the **main verb**, is a word or phrase that expresses the **main action, state, or occurrence in a sentence**.
- For this to be analyzed by computer software, we use semantic role labeling (SRL), also known as **thematic role labeling**.
- SRL is an NLP task that involves **assigning semantic roles to words or phrases** in a sentence and capturing their relationships to the main predicate.
- The goal of SRL is to understand the underlying meaning and roles played by different entities, such as agents, patients, and locations, in expressing an action or event through a sentence. SRL plays a crucial role in revealing the **underlying structure of a sentence**, enabling more profound **analysis and comprehension of the text**.
- Its applications contribute to various NLP tasks by uncovering the semantic relationships within sentences. SRL can be divided into four subtasks, each serving a specific purpose in the overall process

Why is Semantic Role Labeling important?

- Semantic Role Labeling is important because it helps AI systems understand the relationships between different entities and actions in a sentence, **enabling more accurate interpretation and reasoning about natural language.**
- SRL is often used in tasks such as information extraction, question answering, and text summarization.
- Semantic role labeling aims to model the predicate-argument structure of a sentence and is often described as answering "Who did what to whom". BIO notation is typically used for semantic role labeling.

Semantic Role Labeling Example

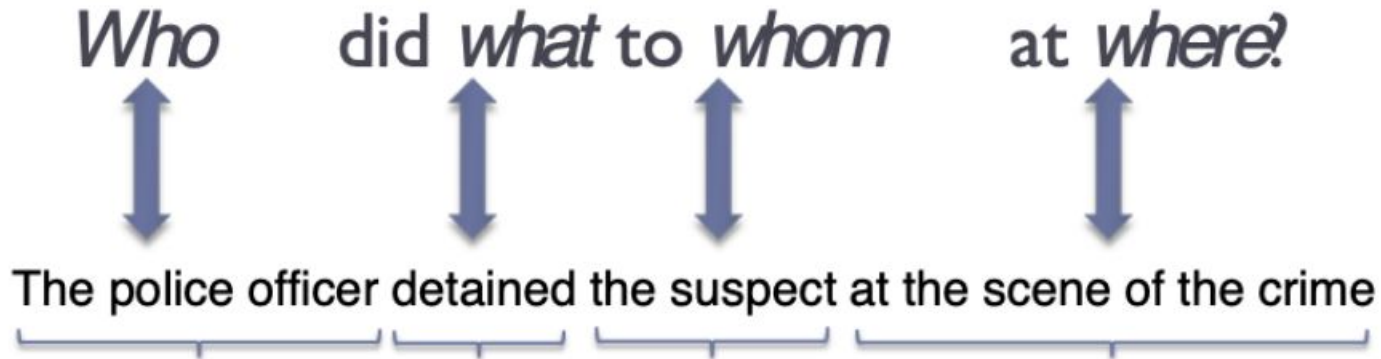
Sentence: "John ate an apple with a fork."

- In this sentence, the **verb "ate" is the predicate.**
- Semantic Role Labeling would involve identifying the semantic roles of the various constituents:
 1. **Agent:** The entity performing the action. In this case, "John" is the agent because he is the one doing the eating.
 2. **Patient / Theme:** The entity that undergoes the action. In this example, "an apple" is the patient because it is what is being eaten.
 3. **Instrument:** The tool or means used to perform the action. Here, "a fork" is the instrument because it is the tool John uses to eat the apple.
- So, the SRL for this sentence would look something like this:
 - **Agent (John) ate**
 - **Patient (an apple)**
 - **Instrument (with a fork)**
- Semantic Role Labeling helps extract this structured information, making it clear **who is doing what to whom and with what.**
- This structured representation of the sentence's meaning is valuable for various natural language processing applications.

Semantic Role

Thematic Role	Definition	Example
AGENT	The volitional causer of an event	<i>The waiter</i> spilled the soup.
EXPERIENCER	The experiencer of an event	<i>John</i> has a headache.
FORCE	The non-volitional causer of the event	<i>The wind</i> blows debris from the mall into our yards.
THEME	The participant most directly affected by an event	Only after Benjamin Franklin broke <i>the ice</i> ...
RESULT	The end product of an event	The city built a <i>regulation-size baseball diamond</i> ...
CONTENT	The proposition or content of a propositional event	Mona asked “ <i>You met Mary Ann at a supermarket?</i> ”
INSTRUMENT	An instrument used in an event	He poached catfish, stunning them <i>with a shocking device</i> ...
BENEFICIARY	The beneficiary of an event	Whenever Ann Callahan makes hotel reservations <i>for her boss</i> ...
SOURCE	The origin of the object of a transfer event	I flew in <i>from Boston</i> .
GOAL	The destination of an object of a transfer event	I drove <i>to Portland</i> .

Semantic Role Labeling - Predicate-argument structure

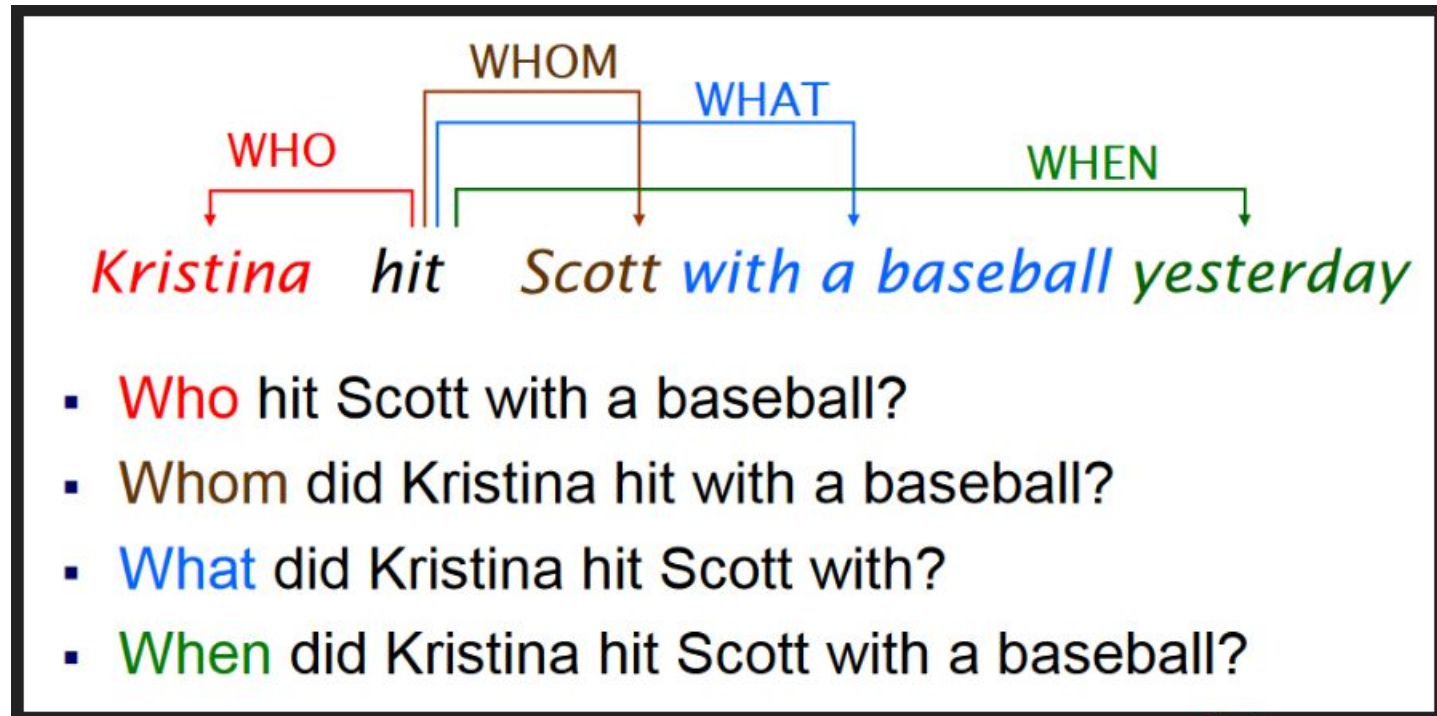


Understanding a sentence = knowing who did what
(to whom, when, where, why...)

Verbs corresponds to predicates (what was done)

Their **arguments** (and modifiers) identify who did it, to whom, where, when, why, etc.)

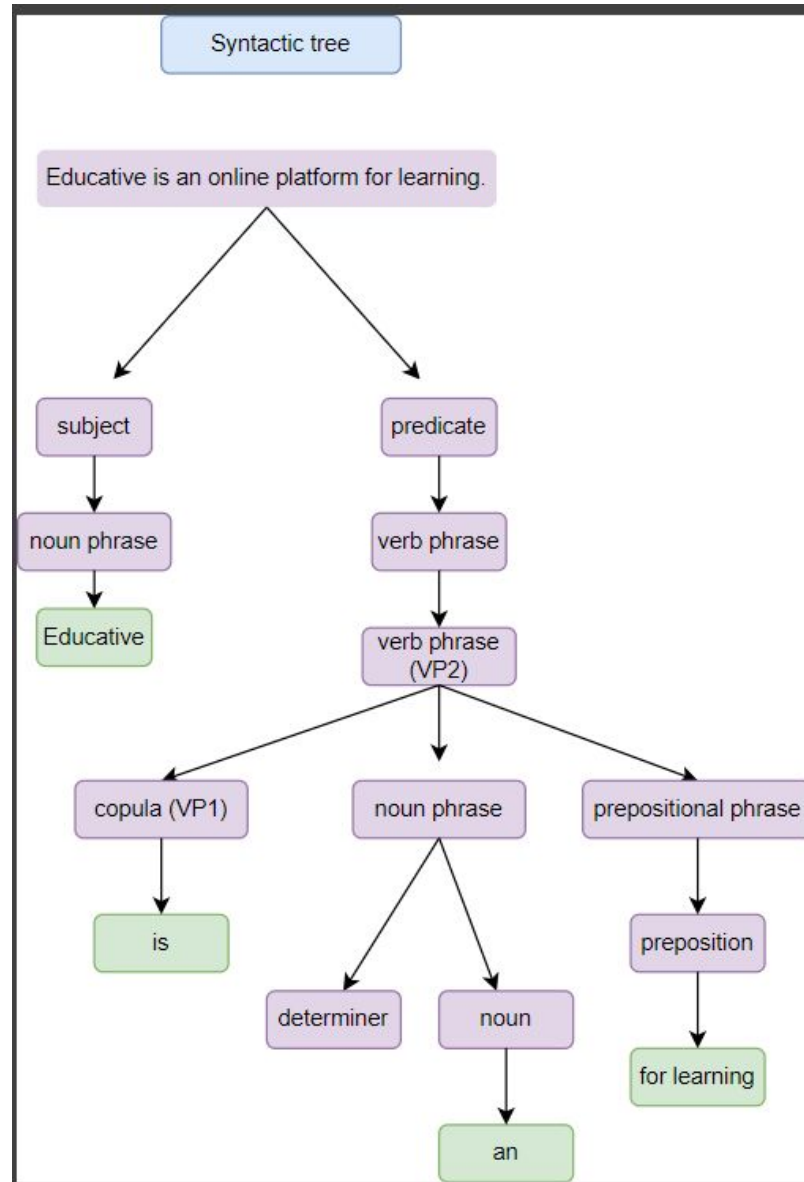
Semantic Role Labeling - Predicate-argument structure



Overall steps in SRL

- Before we get into the critical steps in the process, we have to focus on the preprocessing. This step involves tokenizing the text into individual words or subword units and performing preprocessing tasks like sentence segmentation, part-of-speech tagging, and syntactic parsing.
- In syntactic parsing, there are different types of parse trees, such as constituent parse trees and dependency parse trees. These trees capture the syntactic structure of the sentence and provide the foundation for subsequent steps in the SRL process, such as predicate identification and argument identification. Let's see an example of a syntactic tree.

Overall steps in SRL



Step 1 : Predicate detection

- This subtask involves **identifying the predicates or main verbs in a sentence.**
- The goal is to **locate the words** that express the **main actions or events** in the sentence. For example, in the sentence "Educative is a **hands-on learning platform** for software developers of all levels, the phrase "**is a hands-on learning platform**" would be identified as the **predicate**.

Step 2 : Predicate disambiguation

- Sometimes, a sentence may contain **multiple potential predicates**, making it necessary to **disambiguate and determine the correct one**. Predicate disambiguation resolves any ambiguity in **identifying the primary verb**. For example, in the sentence "**Educative offers various courses for learners of all ages**" the word "offers" can be either a **verb or a noun**. Predicate disambiguation helps in correctly identifying "**offers**" as the verb.
- For the above steps, we can use any **pruning algorithm** for discarding words and selecting the right ones in the syntactic tree constructed. Candidates that are evidently not arguments for a given predicate are eliminated to optimize training time and, more significantly, enhance performance.

Step 3 : Argument identification

- After identifying the predicate, the next step is to identify the **words or phrases** that serve as **arguments for the predicate**.
- **Arguments are the entities or elements** that participate in the action or state expressed by the predicate.
- For example, in the sentence "Educative offers a wide range of courses for learners of all backgrounds, the noun phrase "a wide range of courses" would be identified as the argument of the predicate "**offers.**"

Step 4 : Argument classification

- Once the **arguments are identified**, this subtask involves **classifying or labeling the roles or semantic functions** of each argument.
- For example, in the sentence "Educative offers a wide range of courses for learners of all backgrounds, the argument "Educative" would be assigned the role of **"subject"** or **"agent"** since it is the entity performing the action expressed by the predicate.

Semantic Role Labeling Applications

- **Semantic Parsing:** SRL is often a component of semantic parsing, which aims to convert natural language sentences into formal representations, such as logical forms or knowledge graph queries.
- **Textual Entailment:** SRL can be used to determine if one sentence logically entails another by analyzing the roles of words and their relationships in the sentences.
- **Text Summarization:** SRL can assist in generating concise and informative summaries by identifying the key actions and participants in a text.
- **Information Extraction:** SRL is used to extract structured information from unstructured text. It helps identify relationships between entities and actions, making it easier to populate databases and knowledge graphs with relevant data.

Example of Semantic Role Labeling in Python

```
# Install the AllenNLP library
!pip install allennlp

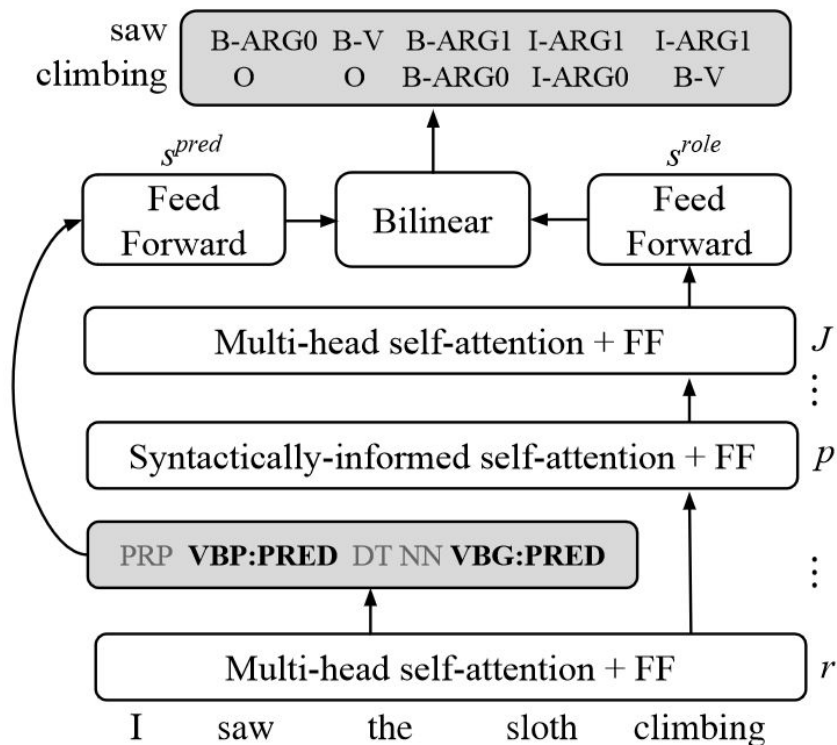
from allennlp.predictors.predictor import Predictor

# Load the semantic role labeling model
predictor = Predictor.from_path("https://storage.googleapis.com/allennlp-public-models/bert-base-srl-2020-03-24.tar.gz")

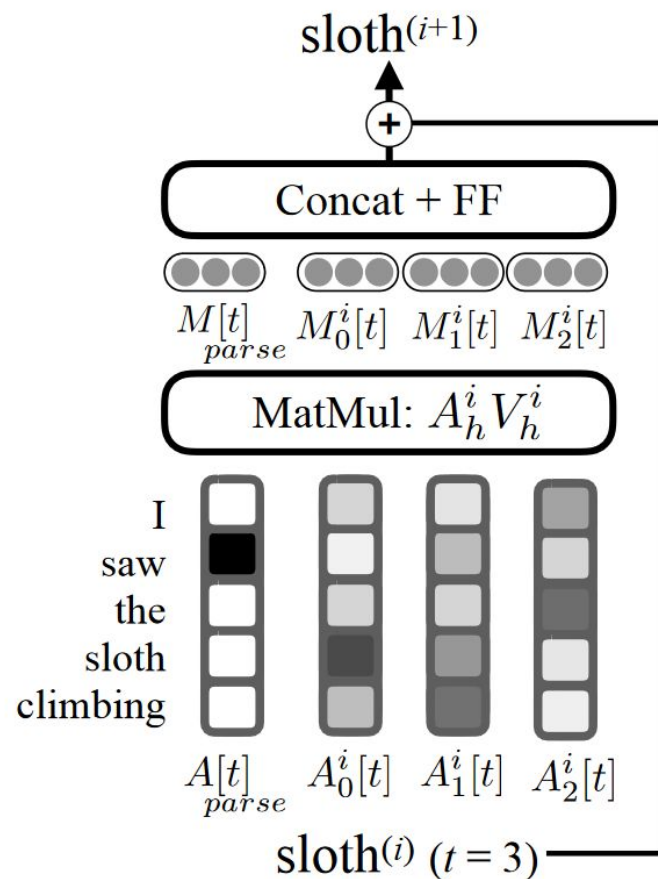
# Define an input sentence
sentence = "The cat chased the dog."

# Perform semantic role labeling
srl_output = predictor.predict(sentence)
print(srl_output)
```

Which are the NN approaches to SRL?



(a) LISA's architecture has multiple self-attention layers



(b) Syntactically-informed self-attention for the query word *sloth*.

Which are the NN approaches to SRL?

- He et al. (2017) used deep BiLSTM with highway connections and recurrent dropout.
- CNN+BiLSTM
- Self Attention
- LSTM

Use of SRL

Reading Comprehension

Read and answer the questions



My Garden

I have a garden. My garden is beautiful. It has many plants and one big tree. There is one rose plant with beautiful roses. One jasmine plant and marigold plant. The tree is of neem. It gives us shade. Plants and trees are very important they help us in many ways. I water them daily and take care of them.



Questions

What all plants are there is the garden?

Who gives shade?

How we should take care of the plants and trees?

Semantic Frame

Semantic Frame

- What is a frame? **Structures representation of concept.**
- A semantic frame is a cognitive and linguistic concept used to **represent structured knowledge** about a particular concept, domain, or situation.
- It serves as a **framework** for organizing, categorizing, and interpreting information related to that concept.
- All information relevant to a particular concept is stored in a single representation.
- This is like a data structure with similar properties and possibilities for knowledge representation as a semantic network
- It is used to understand **what attributes and features are associated with a concept.**

Semantic Frame

A semantic frame typically includes:

- 1. Core Elements:** These are the **essential components or attributes associated** with a **specific concept**. Core elements help define the concept and include roles, properties, and relationships between elements.
- 2. Default Values:** Each core element may have default values associated with it.
- 3. Scripted Events:** Semantic frames include information about typical events or scenarios associated with the concept, helping to understand how the concept functions in context.

Semantic Frame

$\forall x \text{ mammal}(x) \Rightarrow \text{has_part}(x, \text{head})$


$\forall x \text{ elephant}(x) \Rightarrow \text{mammal}(x)$

$\text{elephant}(\text{clyde})$

\therefore

$\text{mammal}(\text{clyde})$

$\text{has_part}(\text{clyde}, \text{head})$

MAMMAL:	
subclass:	ANIMAL
has_part:	head 
*furry:	yes
ELEPHANT	
subclass:	MAMMAL
has_trunk:	yes
*colour:	grey
*size:	large
*furry:	no
Clyde	
instance:	ELEPHANT
colour:	pink
owner:	Fred
Nellie	
instance:	ELEPHANT
size:	small

(*) used to denote defaults that can be over-ridden by:

- An instance or subclass

Semantic Frame

- Let's consider an example of a semantic frame for the concept of "**car.**" A simplified semantic frame for this concept might include the following elements:

1. Core Elements:

1. **Vehicle Type:** This element specifies that the concept is a type of vehicle.
2. **Wheels:** Represents the number of wheels typically associated with a car.
3. **Engine:** Indicates the presence of an engine, which is essential for the car's operation.
4. **Seats:** Describes the seating capacity of the car.
5. **Steering Wheel:** Represents the presence of a steering wheel for controlling the car's direction.

2. Default Values:

1. For "Wheels," the default value might be "four," as most cars have four wheels.
2. For "Engine," the default value would be "yes," indicating the presence of an engine.
3. For "Seats," the default value could be "five," representing a typical five-seat configuration.
4. For "Steering Wheel," the default value is "yes."

3. Scripted Events:

1. "Driving": Describes the typical action associated with a car, which is driving.
2. "Parking": Represents another common action involving cars.
3. "Refueling": Describes the process of filling the car's fuel tank.

Semantic Frame

- This semantic frame for "car" provides a **structured representation of the concept**, allowing individuals to understand **what attributes and features are associated with a car**. It helps in **recognizing and processing information** related to cars in various **contexts / applications**.
- For instance, when encountering the sentence "The red car with four seats and a powerful engine was parked in the driveway," the semantic frame for "car" can help identify and extract information about the car's color, seating capacity, and engine characteristics. This structured representation is valuable in natural language understanding and various NLP applications.

Semantic Frame

- Semantic frame plays a crucial role in applications that require a deeper understanding of the meaning of words and the relationships between them, such as question answering.
- **Example Text: "John gave Mary a beautiful bouquet of flowers for her birthday."**
- We wanted to be able to ask a computer, for example, **Question: "Who gave Mary a beautiful bouquet of flowers for her birthday?"**
- But what if this computer can parse those sentences into semantic frames?

Semantic Frame Parsing:

1. **Predicate: "gave"**
2. **Roles/Entities:**
 1. Agent/Subject: "John"
 2. Recipient/Indirect Object: "Mary"
 3. Theme/Direct Object: "a beautiful bouquet of flowers"
 4. Purpose/Time: "for her birthday"

Semantic Frame

- **Semantic Frame in Question Answering:**
 1. **Question Parsing:** The question is asking about the entity that performed the action and the recipient of that action.
 2. **Semantic Matching:** The system matches the question's semantics with the parsed semantic frame of the text.
 3. **Answer Extraction:** Based on the semantic roles identified in the frame, the system extracts the answer, **which is "John."**
- In this example, **semantic frame parsing enables the question-answering system** to understand the underlying meaning of the text and **identify that "John" is the entity that performed the action of giving, and "Mary" is the recipient.** This deeper understanding of the text's meaning allows the system to provide an accurate answer to the question.
- Semantic frame parsing is a powerful tool for NLP applications because it helps bridge the gap between the surface-level text and the underlying semantics, making it possible to answer questions, extract information, and perform other language understanding tasks with greater accuracy.

Semantic Frame

- Next example illustrates the power of semantic frame parsing in **disambiguating and understanding sentences with similar meanings but different structures**.
- Semantic frame parsing allows a computer to recognize the underlying semantic roles and relationships within sentences, making it easier to answer questions accurately.
- In example:
 - **"The price of bananas increased 5%"**
 - **"The price of bananas rose 5%"**
 - **"There has been a 5% rise in the price of bananas"**
- All of these sentences convey the same essential information: there was an increase in the price of bananas by 5%. While the surface structure varies, the semantic frame remains consistent.
- As you mentioned, a computer equipped with semantic frame parsing capabilities can identify a common frame and recognize the following frame elements:
 - Theme: "The price of bananas"
 - Distance: "5%"

Semantic Frame

- With this understanding, a computer can answer questions like "How much has the price of bananas increased?" by extracting the "Distance" frame element, which is 5%. This demonstrates how semantic frame parsing can help computers understand and respond to questions about the meaning of sentences, even when the sentence structures vary.
- Semantic frame parsing is a valuable tool in natural language processing for disambiguating language and enabling machines to grasp the intended meaning of text, leading to more accurate and context-aware responses in various applications, including question answering.

Semantic Frame

- A semantic frame is a **structured representation of a category or concept** within a sentence, and it provides a set of expectations or rules for what components or elements are likely to be present in that part of the sentence.
- These frames are like templates that help individuals understand and interpret language by specifying the typical features, roles, and relationships associated with a particular concept.
- Consider the semantic frame for the concept of "**restaurant visit.**" This frame might include the following elements:
 - Agent/Subject: The person or group of people visiting the restaurant.
 - Action/Predicate: The action being performed, which is "visit" in this case.
 - Location: The place or restaurant being visited.
 - Time: The time when the visit occurred.

Semantic Frame

- Now, let's use this semantic frame to analyze a few sentences:
 1. "John and Mary went to the Italian restaurant last night."
 1. Agent/Subject: "John and Mary"
 2. Action/Predicate: "went"
 3. Location: "the Italian restaurant"
 4. Time: "last night"
 2. "I'm planning to visit that new sushi place tomorrow."
 1. Agent/Subject: "I"
 2. Action/Predicate: "planning to visit"
 3. Location: "that new sushi place"
 4. Time: "tomorrow"
 3. "The restaurant visit was a delightful experience."
 1. Agent/Subject: None explicitly mentioned (implied or unspecified)
 2. Action/Predicate: "was"
 3. Location: "the restaurant visit"
 4. Time: None mentioned (not relevant in this context)

Semantic Frame

- The typical pipeline to solve the semantic frame task is to **identify targets, classify which frame, and identify arguments.**
- Early works in establishing automatic frame semantic labeling involve two steps: **identify frame elements** boundaries in the sentence and **frame labeling.** This early work used a lot of grammatical features as input for frame labeling. Phrase type (noun phrase, verb phrase, and clause), grammatical function, position, voice, and headword. The final result of this system, on identifying frame element boundaries, achieved 66% and on frame labeling, could achieve 80% accuracy.
- A more recent model has used a **neural network method** to do frame semantic parsing, to see if they can reduce the usage of syntactical features. They split the task into 3 parts: **target identification, frame labeling, and argument identification.**

Semantic Frame

- **Target identification is a task to determine which words or phrases to be labeled.**

A target can also be called a **predicate** and it could be a noun or a verb.

- They used 3 features to do target identification: tokens, part of speech of that tokens, and tokens lemma.
- They also combine pre-trained embedding of tokens, using **GloVe** representation, and trained token embedding.
- Then they used **bi-lstm layer**, which the last layer will be used to predict if the next token is a target.
- Frame labeling task here means the same thing with the earlier method. Here, the authors suggest using bi-lstm based classifier. The model should take at least, the tokens, lemmas, part of speech tags, and the target position, a result of an earlier task

Semantic Frame

- Here the interesting is the **argument identification part**.
- Argument identification is not probably what “argument” some of you may think, but rather refer to the **predicate-argument** structure.
- In other words, given we found a predicate, which words or phrases connected to it. It is essentially the same as semantic role labeling, who did what to whom.
- The main difference is semantic role labeling assumes that all predicates are verbs, while in semantic frame parsing it has no such assumption.

What semantic Frame used for?

Advantages of Frames

- Makes programming easier by grouping related knowledge
 - Can be easily implemented using object-oriented programming techniques
- Fairly intuitive for many applications and non-developers
 - Similar to human knowledge organization
 - Suitable for causal knowledge
 - Easier to understand than logic or rules
- Expressive power, flexibility
 - Easy to set up slots for new properties and relations
 - Easy to include default information and detect missing values
 - Facets allow non-monotonic reasoning
- Inheritance is easily controlled
- Frames are useful for simulating commonsense knowledge, which is a very difficult area for computers to master

Semantic Role Labeling vs Semantic Frame

Aspect	Semantic Role Labeling (SRL)	Semantic Frames
Definition	Assigning semantic roles or labels to words or phrases in a sentence to identify their grammatical and semantic relationships within the sentence's predicate.	Structured representations of knowledge that capture information about a specific concept, event, or scenario, including roles, properties, and relationships associated with that concept.
Focus	Identifying roles of individual arguments (agents, patients, instruments, etc.) in relation to the predicate in a specific sentence.	Providing a broader framework for representing information about a concept or event, encompassing various roles, attributes, and events related to the concept.
Purpose	Understanding who is doing what to whom in a sentence. Provides a deeper understanding of the syntactic and semantic structure of a single sentence.	Organizing and representing knowledge about a particular domain or concept comprehensively. Not limited to a single sentence but applicable across multiple sentences and documents.
Example	In the sentence "John ate a pizza," SRL identifies "John" as the agent, "ate" as the predicate, and "a pizza" as the patient.	A semantic frame for "restaurant visit" includes roles for agent, location, time, and scripted events like "ordering food" and "paying the bill."

Ontologies and Semantics

How often does your virtual assistant misunderstand what you say?

- While natural language understanding (NLU) applications are evolving, they still frequently misinterpret spoken and written language.
- The reason for this is language **ambiguity**.
- Words in different contexts can mean different things.
- This makes things tricky for NLU applications that need to understand language intent.

Why ontology?

- When you consider the integral nature of NLU to drive smart machines and autonomous systems, taking the right NLU approach is critical to optimizing outcomes and decision making.
- To build effective NLU programs, it's essential to have a **deep understanding** of **language structure**. This includes knowledge of syntax (sentence structure), semantics (word meanings), and pragmatics (contextual interpretation). NLU systems must be able to analyze and interpret these aspects of language to provide accurate results.
- **Word classification**, often referred to as part-of-speech tagging, is a fundamental task in NLU. It involves categorizing words into their grammatical roles, such as nouns, verbs, adjectives, etc. This information is essential for understanding the grammatical structure of sentences and extracting meaning from them.
- So it's important to understand how ontology works with NLU.

Ontology and Taxonomy in NLU

- **Taxonomy:** Taxonomy refers to the **hierarchical categorization of concepts or entities**. In the context of NLU, it can be used to create structured hierarchies of words or concepts, making it easier to organize and retrieve information. For example, in a taxonomy related to animals, you might have categories like "Mammals," "Birds," and "Reptiles."
- **Ontology:** Ontology goes beyond taxonomy by defining not only the **hierarchical relationships between concepts** but also **the relationships between their properties and attributes**. It provides a more detailed and semantically rich representation of knowledge. In NLU, ontologies help in understanding the meaning of words in context and capturing complex relationships between concepts. For instance, in an ontology for the medical domain, you could represent not only the hierarchy of diseases but also the relationships between symptoms, treatments, and patient profiles.

Ontology and Taxonomy in NLU

- Both taxonomy and ontology are valuable tools in NLU. Taxonomy helps organize and categorize information hierarchically, while ontology adds semantic depth by defining relationships and attributes between concepts.
- Combining these approaches can enhance the ability of NLU systems to understand and work with language, leading to more effective outcomes and decision-making in various applications.

Ontologies and their significance in NLU

- Ontologies are structured models that define concepts, their properties, and relationships in a specific domain. They play a critical role in formalizing knowledge, ensuring clarity, consistency, and detailed representation of information.
- In an ontology, concepts like "dog" have associated properties (e.g., "has fur") and relationships (e.g., "is a subclass of mammal"), enhancing understanding within that domain.

Ontologies and semantics

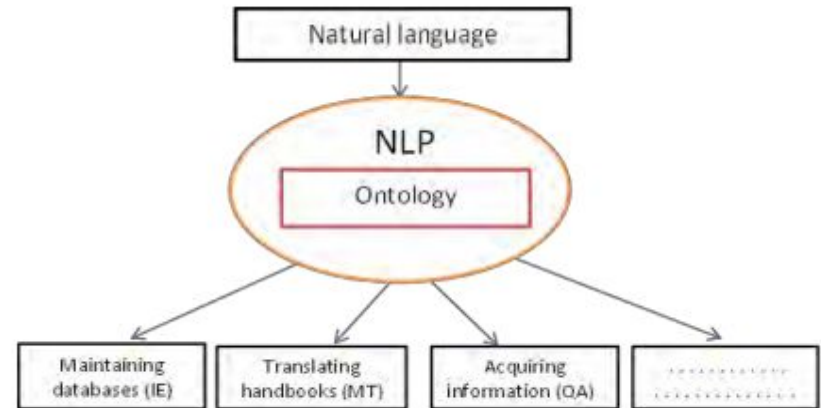
- Ontology is closely related to **semantics, which is the study of meaning in language**. In the context of ontologies, semantics are used to clarify the meaning of concepts and how they relate to each other. This semantic richness enables machines to understand the context and nuances of language.
- Ontologies are particularly valuable for **disambiguating word meanings**. Example of the word "diamond" illustrates this well. Without context, "diamond" could refer to a baseball player, a jeweler, or a card suit. However, with the aid of an ontology, a machine can determine the correct interpretation based on the context provided by the surrounding words or the specific domain it's working in.

How Ontology Impacts NLU?

- Ontologies play a pivotal role in advancing NLU and enhancing various aspects of AI technologies. **The benefits of ontologies on NLU:**
 - **Precise Understanding:** Ontologies enable precise language understanding, helping machines grasp nuances and context, thus enhancing language processing accuracy and advancing NLU capabilities for businesses.
 - **Efficiency and Accuracy:** Ontologies improve NLU efficiency and accuracy through structured language analysis, aiding AI in extracting valuable insights from text and speech data.
 - **Improved Entity Analysis:** Ontologies enhance entity recognition, enabling AI systems to categorize entities in text or speech more accurately, especially in named entity recognition.

Relation between Natural language, NLP and ontology

- The idea involves integrating various software tools to semi-automatically create Natural Language Ontologies (NLOs) for specific domains.
- This approach is ambitious due to the general and challenging nature of NLO construction.
- Adapting and imposing constraints are essential steps, given the unique characteristics of the specific natural language under consideration.



Relation between natural language, NLP and ontology

Applications of Ontology

- 1. Comprehensive Domain Description:** Ontologies serve as tools to thoroughly analyze and describe a specific domain, aligning the description with the requirements and needs of users. This comprehensive representation aids in understanding the domain's intricacies.
- 2. Common Understanding:** Ontologies help establish a shared and common understanding of how information is structured within a domain. This shared understanding is valuable for effective communication and collaboration among individuals and software agents.
- 3. Knowledge Reuse:** Ontologies facilitate the reuse of domain knowledge. By capturing and formalizing knowledge about a domain, ontologies make it easier to apply that knowledge in various contexts, reducing redundancy and enhancing efficiency.
- 4. Separation of Knowledge:** Ontologies allow for the separation of domain knowledge from operational knowledge. This separation is useful in keeping domain-specific information distinct from the specific procedures and functions used to operate within that domain, leading to greater flexibility and maintainability.

OWL (Web Ontology Language)

- An OWL ontology is a formal, machine-readable representation of knowledge or information within a specific domain. OWL stands for Web Ontology Language, and it is a standard language used in the field of semantic web and knowledge representation. OWL ontologies define concepts, their properties, and the relationships between them.

Here are some key aspects of OWL ontologies:

1. Concepts and Classes: OWL ontologies define classes or concepts that represent entities or categories within a domain. For example, in a medical ontology, you might have classes like "Patient," "Doctor," and "Disease."
2. Properties: OWL allows you to define properties to describe the characteristics or relationships between classes. Properties can be binary (relating two individuals or a class and an individual), such as "hasDoctor" linking a patient to their doctor.
3. Individuals: Individuals are specific instances or members of classes. For instance, "John Smith" could be an individual belonging to the class "Patient."

OWL (Web Ontology Language)

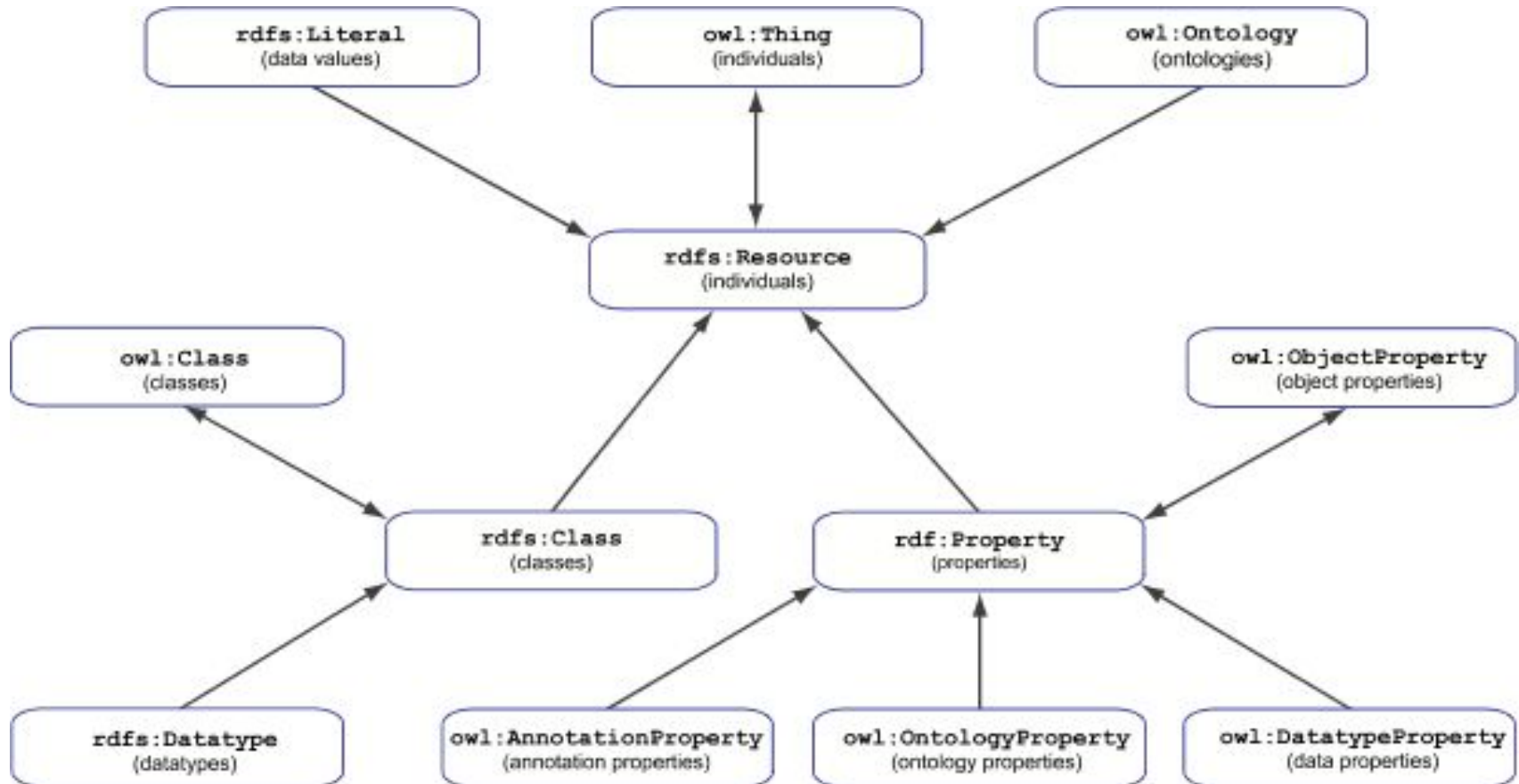
Here are some key aspects of OWL ontologies:

4. Relationships: OWL ontologies specify relationships between individuals and classes using properties. These relationships can represent attributes (e.g., "has Age" linking an individual to their age) or more complex associations.
5. Inference: One of the powerful features of OWL is its support for automated reasoning and inference. Ontology reasoners can use the defined rules and axioms to infer new knowledge or check the consistency of the ontology.
6. RDF Serialization: OWL ontologies are often serialized using RDF (Resource Description Framework) syntax, which is a standard for representing information on the web. This makes OWL ontologies compatible with the semantic web and linked data principles.

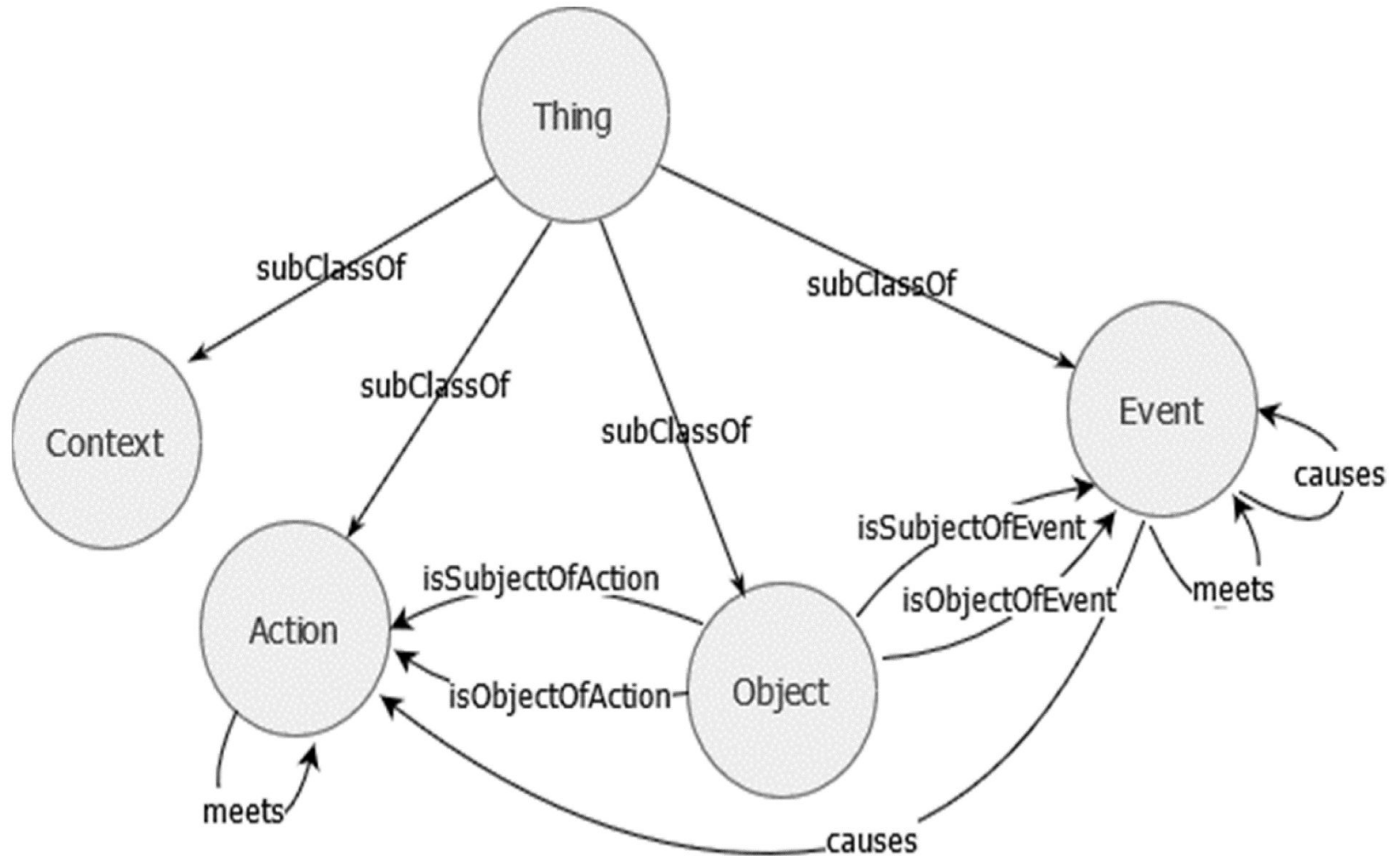
OWL (Web Ontology Language)

- OWL ontologies are used in various applications, including knowledge representation, data integration, semantic search, and building intelligent systems that can reason about domain-specific knowledge. They are a fundamental component of the semantic web, helping to make information on the web more understandable and machine-processable.

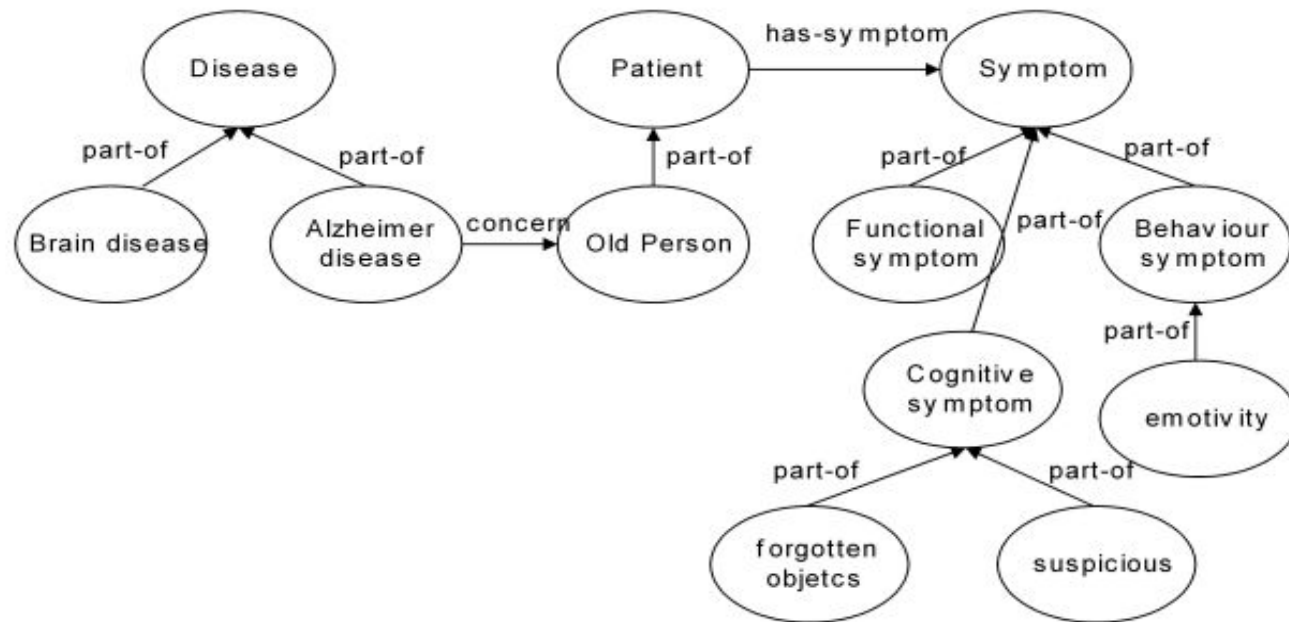
Ontology Structure



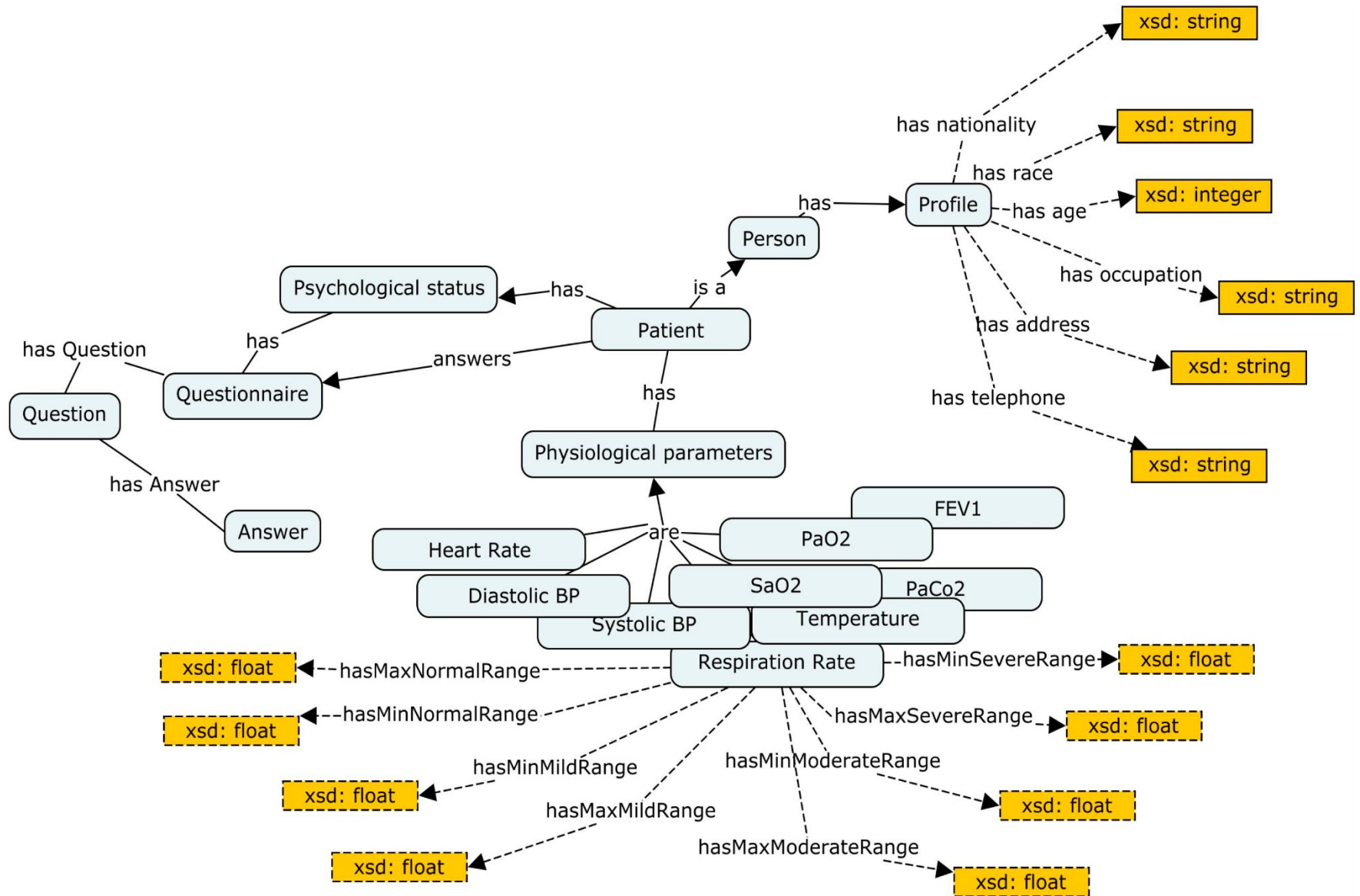
Ontology Structure



Example of Ontology for medical domain



Example of Ontology for patient



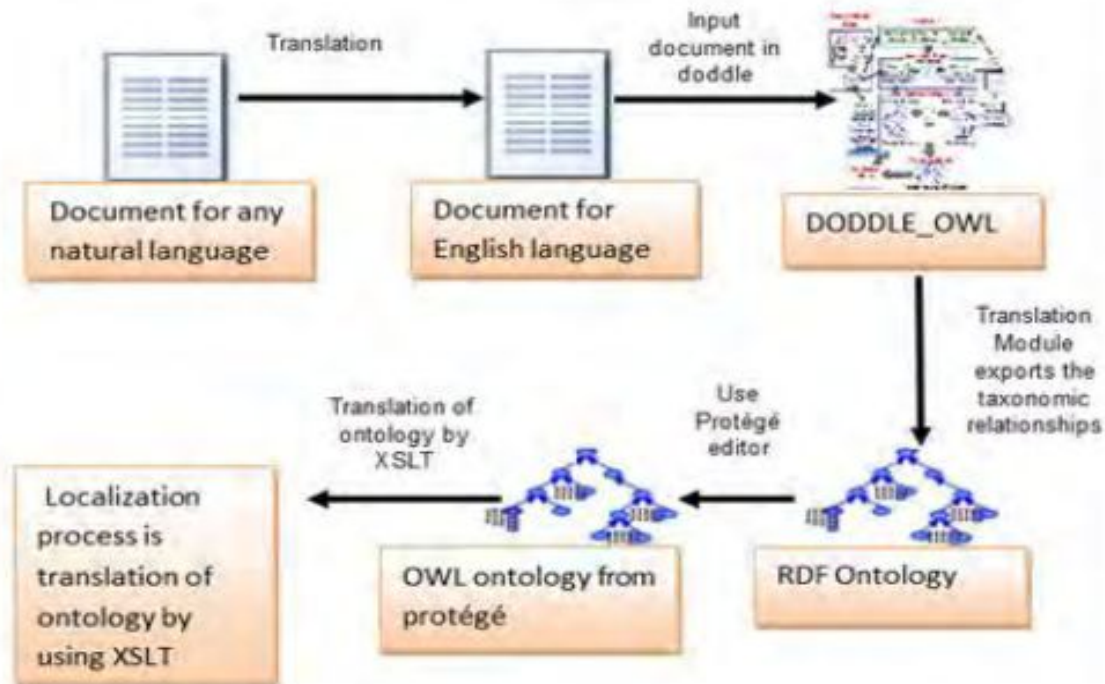
DODDLE-OWL Overview

- DODDLE-OWL (Domain Ontology rapid DeveLopment Environment – OWL extension) is a specialized tool for quickly creating domain ontologies for the Semantic Web.
- Originally intended for Japanese and English, it has been expanded to support other languages with WordNet dictionaries by translating text into English and modifying the ontology.
- This tool simplifies ontology development and knowledge representation across different languages and domains.

Construction of domain NLO for different languages

- **Ontology Selection Module:** Users choose reference ontologies from sources like WordNet, EDR (vocabulary or technical terminology dictionary), and existing OWL ontologies as a foundation.
- **Input Document Selection Module:** Users select domain-specific English documents. Some words are extracted from these documents, with optional part-of-speech (POS) selection.
- **Input Term Selection Module:** Extracted terms form a list, including compound words, POS, Term Frequency (TF - how often a term appears in a document), and Inverse Document Frequency (IDF - rarity of a term across the entire document collection).
- **Input Concept Selection Module:** Users map the word senses of input terms to concepts in the chosen reference ontologies from the Ontology Selection Module.
- **Hierarchy Construction Module:** This module automatically generates the initial concept hierarchy for the ontology. It does this by referencing the selected reference ontologies and the documents provided in earlier steps.

Building NLO



Example

- Input Text:
 - Programming is a beautiful art. Programming is a course in the Faculty of Mathematics. Programming is a discipline very effective and it is learned in many colleges in the world. Now we can use a lot of new programming languages and make many different programs.

DOODLE OWL Document and Graph

```
</owl:Class>
<owl:Class rdf:about="discipline">
  <rdfs:subClassOf>
    <owl:Class rdf:about="activity"/>
  </rdfs:subClassOf>
  <rdfs:label xml:lang="en">discipline</rdfs:label>
  <rdfs:comment xml:lang="en">training to improve strength
or self-control</rdfs:comment>
</owl:Class>
<owl:Class rdf:about="course">
  <rdfs:label xml:lang="en">course</rdfs:label>
  <rdfs:comment xml:lang="en">a line or route along which
something travels or moves; "the hurricane demolished
houses in its path"; "the track of an animal"; "the
course of the river"</rdfs:comment>
  <rdfs:subClassOf>
```

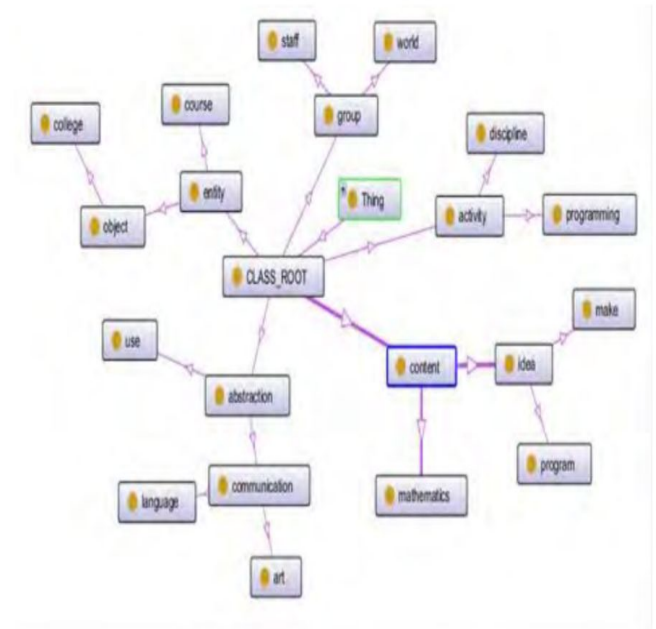


Fig. 8. Ontology graph for English words

Semantic Network and Knowledge Graph

Semantic network vs ontology

- A semantic network and an ontology are both knowledge representation formalisms that can be used to represent knowledge about a domain. However, they have different strengths and weaknesses.
- A semantic network is a graph-based representation of knowledge. The nodes in the graph represent concepts, and the edges represent relationships between concepts. Semantic networks are good at representing the meaning of words and phrases, and they can be used to represent commonsense knowledge. However, they can be difficult to scale to large domains.
- An ontology is a formal, explicit specification of a shared conceptualization. Ontologies are good at representing the relationships between concepts, and they can be used to represent domain-specific knowledge. However, they can be difficult to create and maintain.

Semantic network vs ontology

- A semantic network and an ontology are both knowledge representation formalisms that can be used to represent knowledge about a domain. However, they have different strengths and weaknesses.
- A semantic network is a graph-based representation of knowledge. The nodes in the graph represent concepts, and the edges represent relationships between concepts. Semantic networks are good at representing the meaning of words and phrases, and they can be used to represent commonsense knowledge. However, they can be difficult to scale to large domains.
- An ontology is a formal, explicit specification of a shared conceptualization. Ontologies are good at representing the relationships between concepts, and they can be used to represent domain-specific knowledge. However, they can be difficult to create and maintain.

Semantic network vs ontology

Feature	Semantic Network	Ontology
<i>Type of representation</i>	Graph	Formal specification
<i>Strengths</i>	Represents the meaning of words and phrases, represents commonsense knowledge	Represents the relationships between concepts, represents domain-specific knowledge
<i>Weaknesses</i>	Can be difficult to scale to large domains	Can be difficult to create and maintain

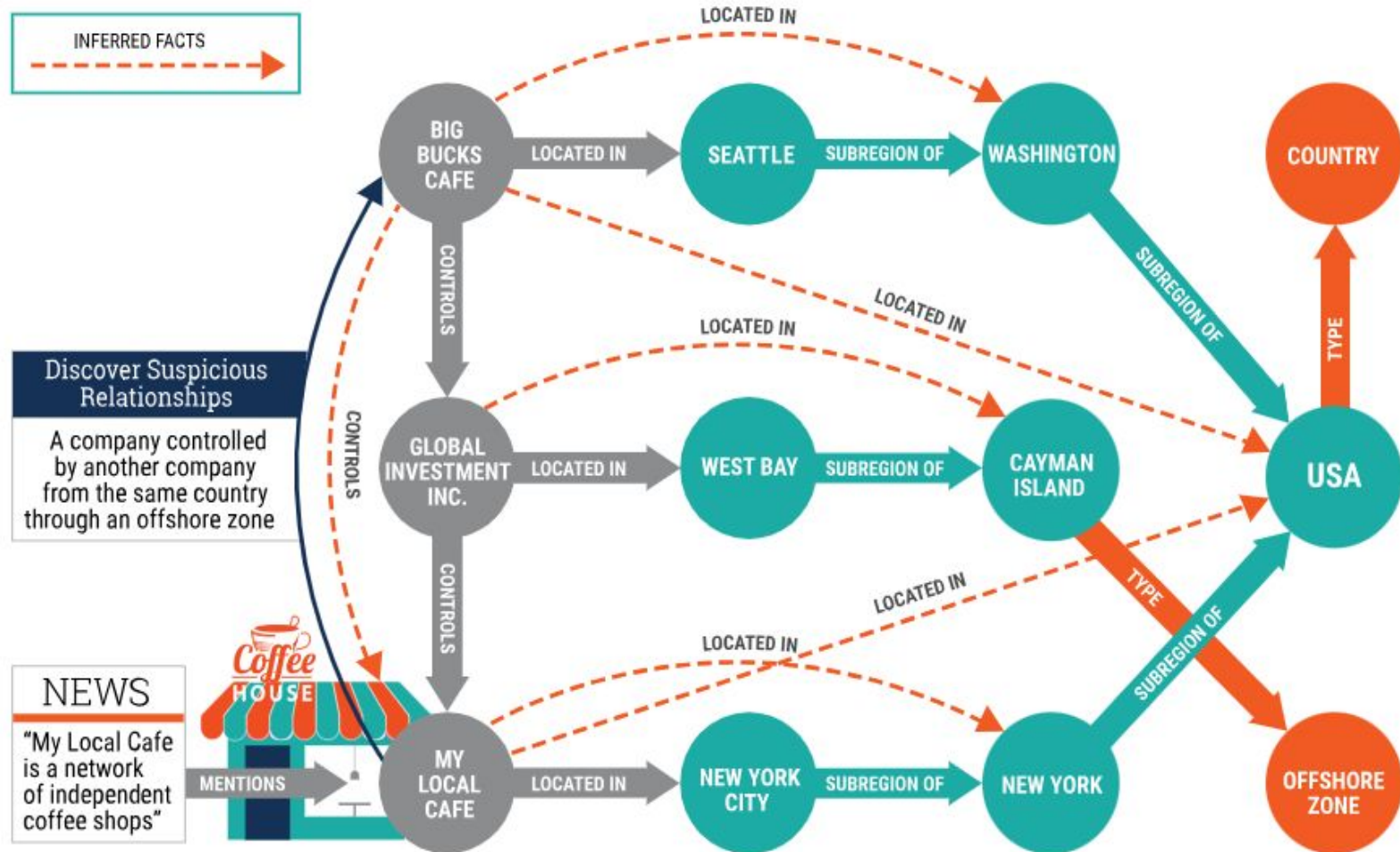
Semantic network vs ontology

- In general, semantic networks are a good choice for representing knowledge that is not well-structured or that is not domain-specific. Ontologies are a good choice for representing knowledge that is well-structured and that is domain-specific.
- Here are some examples of how semantic networks and ontologies are used in practice:
 - Semantic networks are used in natural language processing to represent the meaning of words and phrases.
 - Ontologies are used in the semantic web to represent the relationships between concepts.
 - Ontologies are also used in artificial intelligence to represent domain-specific knowledge.

Knowledge Graph

- A knowledge graph, also known as a semantic network, represents a network of real-world entities—i.e. **objects, events, situations, or concepts**—and illustrates the relationship between them. This information is usually stored in a graph database and visualized as a graph structure, prompting the term knowledge “graph.”
- A knowledge graph is made up of three main components: nodes, edges, and labels. Any object, place, or person can be a node. An edge defines the relationship between the nodes.
- Knowledge graphs put data in context via linking and semantic metadata and this way provide a framework for data integration, unification, analytics and sharing.

Knowledge Graph



Knowledge Graph

- The heart of the knowledge graph is a knowledge model – a collection of interlinked descriptions of concepts, entities, relationships and events where:
- Descriptions have formal semantics that allow both people and computers to process them in an efficient and unambiguous manner;
- Descriptions contribute to one another, forming a network, where each entity represents part of the description of the entities related to it;
- Diverse data is connected and described by semantic metadata according to the knowledge model.

Knowledge Graph

- KGs are used in a variety of NLP models, including:
 - Question answering: KGs can be used to answer questions by finding the relevant information in the graph.
 - Semantic parsing: KGs can be used to parse natural language queries into a form that can be understood by a machine.
 - Machine translation: KGs can be used to improve the accuracy of machine translation by providing additional context about the words and phrases being translated.
 - Natural language generation: KGs can be used to generate natural language text by retrieving information from the graph and organizing it in a coherent way.
 - Text summarization: KGs can be used to summarize text by extracting the most important information from the graph.

Knowledge Graph

- In question answering, a KG can be used to find the answer to a question by finding the relevant information in the graph. For example, if you ask the question "What is the capital of France?", a KG can be used to find the node for France and then find the edge that connects France to its capital, Paris.
- In semantic parsing, a KG can be used to parse natural language queries into a form that can be understood by a machine. For example, if you ask the question "What is the color of the sky?", a KG can be used to parse the query into a logical form that represents the meaning of the query.
- In machine translation, a KG can be used to improve the accuracy of machine translation by providing additional context about the words and phrases being translated. For example, if you are translating the sentence "The cat sat on the mat" from English to French, a KG can be used to provide the information that cats are typically furry and have four legs. This information can be used to choose the correct translation of the word "cat" in French.
- In natural language generation, a KG can be used to generate natural language text by retrieving information from the graph and organizing it in a coherent way. For example, if you want to generate a summary of the article "The history of France", a KG can be used to retrieve information about the key events in French history and then organize this information into a coherent narrative.
- In text summarization, a KG can be used to summarize text by extracting the most important information from the graph. For example, if you want to summarize the article "The history of France", a KG can be used to extract the key facts about French history and then organize these facts into a concise summary.

What is Knowledge Graph?

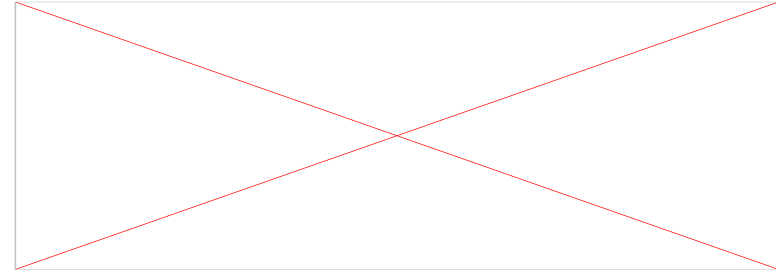
- A knowledge graph is a way of storing data that resulted from an information extraction task.
- Many basic implementations of knowledge graphs make use of a concept we call triple,
- That is a set of three items(a subject, a predicate and an object) that can use to store information about something.

Knowledge graph (KG)

- A knowledge graph (KG) is a structured representation of knowledge that is typically modeled as a graph.
- The nodes in the graph represent entities, and the edges represent relationships between entities.
- A KG can be used to represent a wide variety of knowledge, including facts about the real world, concepts, and relationships between concepts.

Knowledge Representation

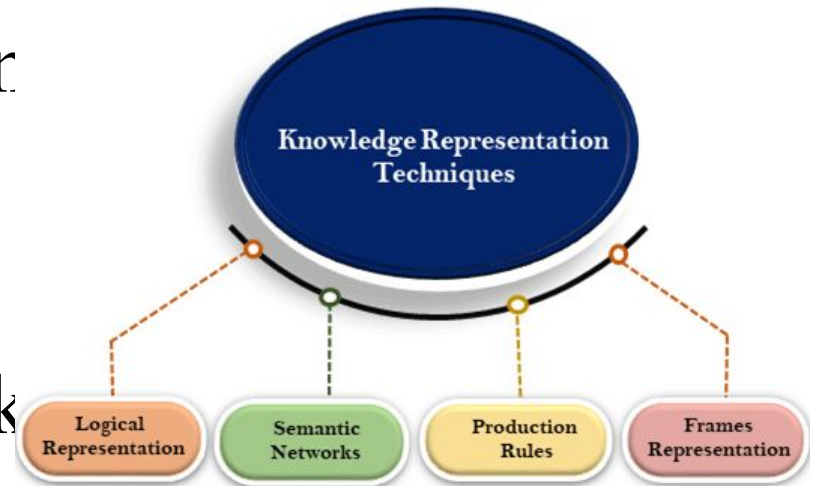
- can define a graph as a set of nodes and edges.
- Node A and Node B here are two different entities.
- These nodes are connected by an edge that represents the relationship between the two nodes.
- Now, this is the smallest knowledge graph



Techniques of knowledge representation

- There are mainly four ways of knowledge representation which are given as follows:

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules



1. Logical Representation

- Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation.
- Logical representation means drawing a conclusion based on various conditions.
- This representation lays down some important communication rules.
- It consists of precisely defined syntax and semantics which supports the sound inference.
- Each sentence can be translated into logics using syntax and semantics.

Pros and Cons of Logical Representation

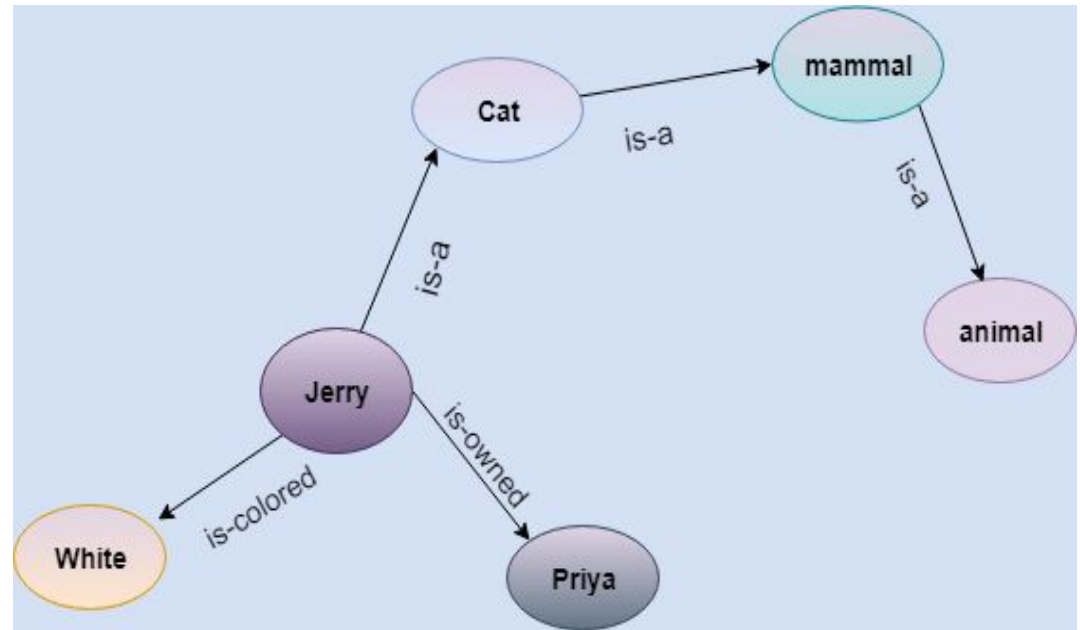
- **Advantages of logical representation:**
- Logical representation enables us to do logical reasoning.
- Logical representation is the basis for the programming languages.
- **Disadvantages of logical Representation:**
- Logical representations have some restrictions and are challenging to work with.
- Logical representation technique may not be very natural, and inference may not be so efficient.

2. Semantic Network Representation

- Semantic networks are alternative of predicate logic for knowledge representation.
- In Semantic networks, we can represent our knowledge in the form of graphical networks.
- This network consists of nodes representing objects and arcs which describe the relationship between those objects.
- Semantic networks can categorize the object in different forms and can also link those objects.
- Semantic networks are easy to understand and can be easily extended.
- This representation consist of mainly two types of relations:
 1. IS-A relation (Inheritance)
 2. Kind-of-relation

Example

- Statements:
 1. Jerry is a cat.
 2. Jerry is a mammal
 3. Jerry is owned by Priya.
 4. Jerry is brown colored.
 5. All Mammals are animal.



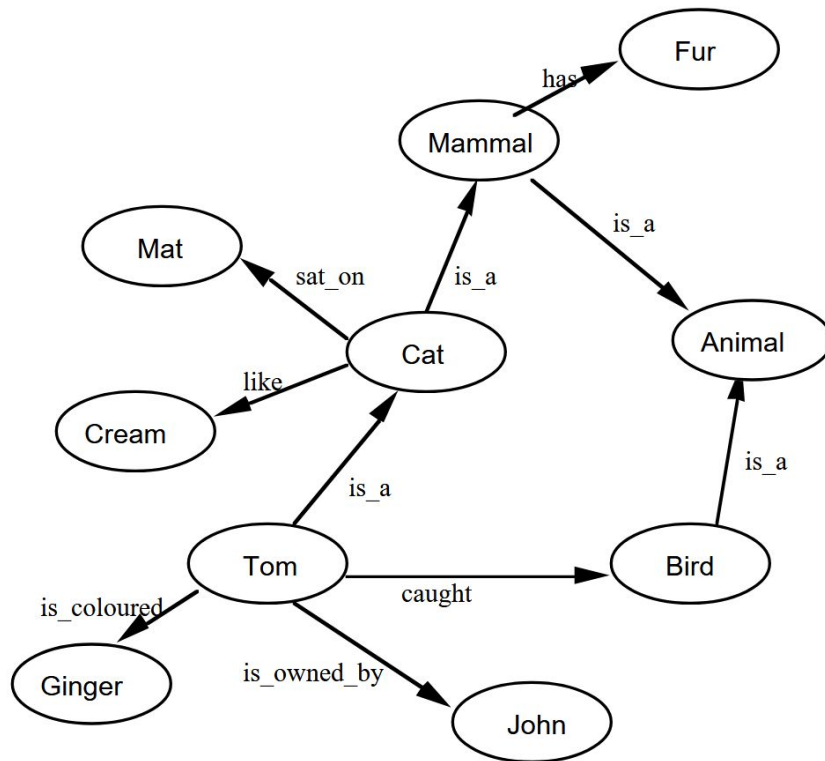
Semantic Network

- A semantic network and an ontology are both knowledge representation formalisms that can be used to represent knowledge about a domain. However, they have different strengths and weaknesses.
- A semantic network is a graph-based representation of knowledge. The nodes in the graph represent concepts, and the edges represent relationships between concepts. Semantic networks are good at representing the meaning of words and phrases, and they can be used to represent commonsense knowledge. However, they can be difficult to scale to large domains.
- An ontology is a formal, explicit specification of a shared conceptualization. Ontologies are good at representing the relationships between concepts, and they can be used to represent domain-specific knowledge. However, they can be difficult to create and maintain.

Semantic networks and Ontologies

Feature	Semantic Network	Ontology
Type of representation	Graph	Formal specification
Strengths	Represents the meaning of words and phrases, represents commonsense knowledge	Represents the relationships between concepts, represents domain-specific knowledge
Weaknesses	Can be difficult to scale to large domains	Can be difficult to create and maintain

Example



is intended to represent the data:

Tom is a cat.

Tom caught a bird.

Tom is owned by John.

Tom is ginger in colour.

Cats like cream.

The cat sat on the mat.

A cat is a mammal.

A bird is an animal.

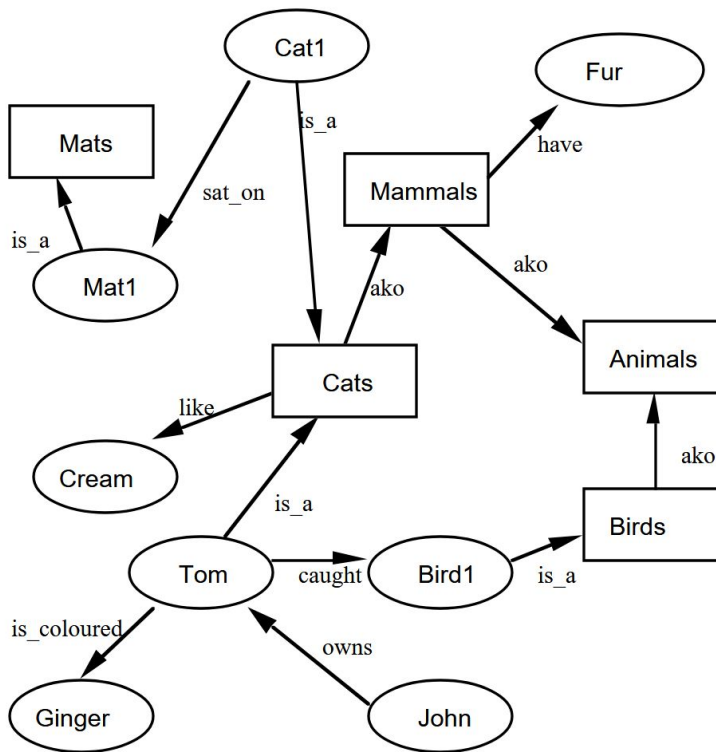
All mammals are animals.

Mammals have fur

Different Representations

- We use of quantifiers, names and predicates to makes it clear what we mean so:
- Tom is a cat is represented by $\text{Cat}(\text{Tom})$
- The cat sat on the mat is represented by
- $\exists x \exists y (\text{Cat}(x) \wedge \text{Mat}(y) \wedge \text{SatOn}(x,y))$
- A cat is a mammal is represented by
- $\forall x (\text{Cat}(X) \rightarrow \text{Mammal}(x))$

Revised Semantic Networks



A direct Prolog representation can be used, with classes represented by predicates, thus:

```
cat(tom).
cat(cat1).
mat(mat1).
sat_on(cat1,mat1).
bird(bird1).
caught(tom,bird1).
like(X,cream) :- cat(X).
mammal(X) :- cat(X).
has(X,fur) :- mammal(X).
animal(X) :- mammal(X).
animal(X) :- bird(X).
owns(john,tom).
is_coloured(tom,ginger)
```

Pros and CONS of Semantic representation

- **Drawbacks in Semantic representation:**

1. Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible in the worst case scenario that after traversing the entire tree, we find that the solution does not exist in this network.
2. Semantic networks try to model human-like memory (Which has 10^{15} neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.
3. These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
4. Semantic networks do not have any standard definition for the link names.
5. These networks are not intelligent and depend on the creator of the system.

- **Advantages of Semantic network:**

1. Semantic networks are a natural representation of knowledge.
2. Semantic networks convey meaning in a transparent manner.
3. These networks are simple and easily understandable.

3. Frame Representation

- A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world.
- Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations.
- It consists of a collection of slots and slot values.
- These slots may be of any type and sizes.
- Slots have names and values which are called facets.
- **Facets:** The various aspects of a slot is known as **Facets**.
- Facets are features of frames which enable us to put constraints on the frames.
- Example: IF-NEEDED facts are called when data of any particular slot is needed.
- A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values
- A frame is also known as **slot-filter knowledge representation** in artificial intelligence.

3. Frame Representation

- Frames are derived from semantic networks and later evolved into our modern-day classes and objects.
- A single frame is not much useful. Frames system consist of a collection of frames which are connected.
- In the frame, knowledge about an object or event can be stored together in the knowledge base.
- The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Example

- Let's suppose we are taking an entity, Peter. Peter is an engineer as a profession, and his age is 25, he lives in city London, and the country is England. So following is the frame representation for this:

Slots	Filter
Name	Peter
Profession	Doctor
Age	25
Marital status	Single
Weight	78

Pros and Cons of Frame representation

- **Advantages of frame representation:**

1. The frame knowledge representation makes the programming easier by grouping the related data.
2. The frame representation is comparably flexible and used by many applications in AI.
3. It is very easy to add slots for new attribute and relations.
4. It is easy to include default data and to search for missing values.
5. Frame representation is easy to understand and visualize.

- **Disadvantages of frame representation:**

1. In frame system inference mechanism is not be easily processed.
2. Inference mechanism cannot be smoothly proceeded by frame representation.
3. Frame representation has a much generalized approach.

4. Production Rules

- Production rules system consist of (**condition, action**) pairs which mean, "If condition then action". It has mainly three parts:
 - The set of production rules
 - Working Memory
 - The recognize-act-cycle
- In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out.
- The condition part of the rule determines which rule may be applied to a problem.
- And the action part carries out the associated problem-solving steps.
- This complete process is called a recognize-act cycle.
- **Example:**
- **IF (at bus stop AND bus arrives) THEN action (get into the bus)**
- **IF (on the bus AND paid AND empty seat) THEN action (sit down).**
- **IF (on bus AND unpaid) THEN action (pay charges).**
- **IF (bus arrives at destination) THEN action (get down from the bus).**

Pros and Cons of Production Rules

- **Advantages of Production rule:**

1. The production rules are expressed in natural language.
2. The production rules are highly modular, so we can easily remove, add or modify an individual rule.

- **Disadvantages of Production rule:**

1. Production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses.
2. During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

Applications of Knowledge Graphs

- **Data Governance:** managing these various contexts and data silos, tracking data lineage, ownership and usage patterns by focusing on the value, quality and usability of the data.
- **Automated Fraud Detection:** Machine learning algorithms have already helped financial institutions to monitor fraud and the risk behind it in a more automated way.
- **Knowledge Management:** Knowledge graphs have an interesting application in finance knowledge management in that they can be used to aggregate and represent data from various sources, such as stock quotes, corporate financial reports, news, and social network data, among others.

Intent Detection and Classification

Intent Detection and Classification

- In many applications, particularly in chatbots, virtual assistants, and other human-computer interaction systems, users communicate with a system or application using natural language, such as text or speech.
- Intent classification or detection is the process of identifying and categorizing the intent behind a user's query in a chatbot conversation and categorizes them such as requests or approvals.
- The intent is the **underlying purpose or action the user wants to achieve with their input.**
- It represents **what the user is trying to do or the question they are asking.**
- An intent classifier automatically analyzes texts and categorizes them into intents.
- Its basically used to find the purpose behind the text in hand. Highly utilized in chatbots

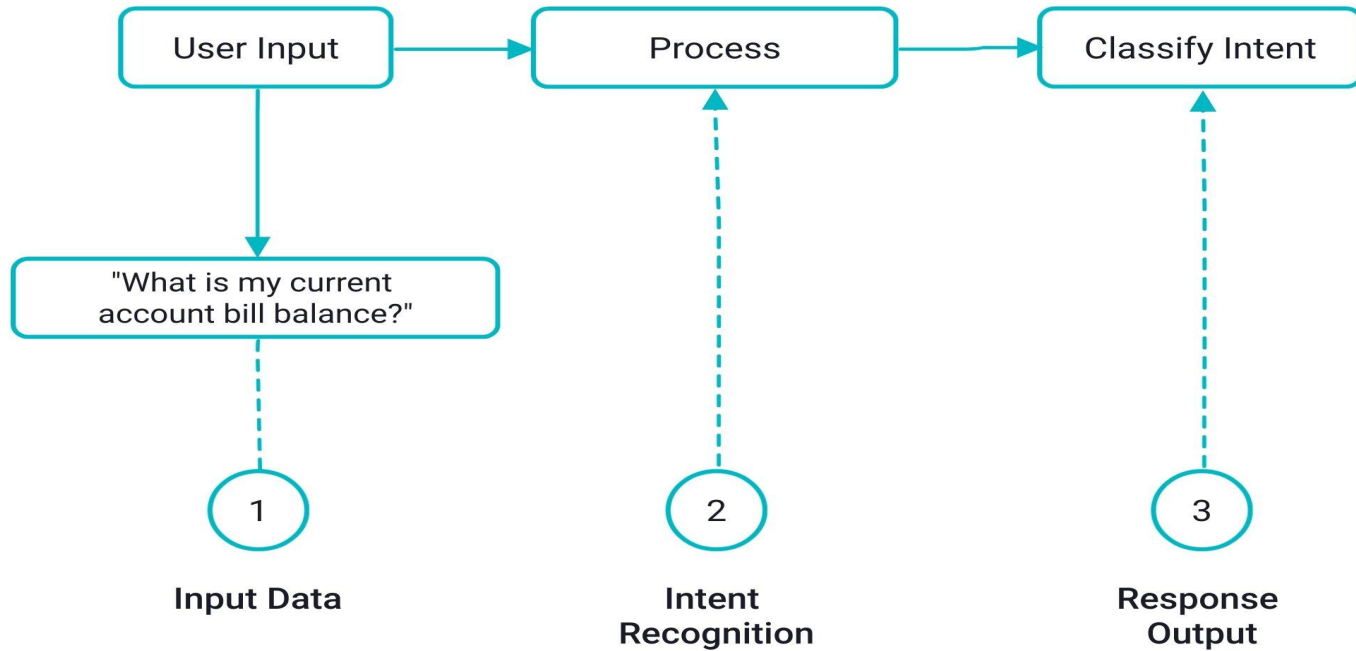
Intent Detection and Classification

- Example: if a user types or says, **"What's the weather forecast for tomorrow?"** in a weather app or chatbot, the intent behind this input is likely **"Get Weather Forecast."**
- Intent detection and classification algorithms are used to recognize that the user's goal is to **retrieve weather information for a specific day.**
- Example: "I'd like to book a table for two at an Italian restaurant in downtown at 7 PM on Saturday.“ ->main intent **is to book a restaurant reservation.**
- Once the intent is identified, the system can proceed to gather additional information from the user, such as the date, time, restaurant preferences, and contact details, to complete the reservation process effectively.

Intent Detection and Classification

- Intent recognition, also commonly referred to as intent classification, uses machine learning and natural language processing to associate text data and expression to a given intent.
- In other words, intent recognition takes a **given query as an input** and associates it to the target class.
- For example, during a telephone prompter in an automated call, the model learns from the speech data what service a customer is looking for, based on key phrases, such as “**pay my bill**” or “**speak to a representative.**”
- Hence, intent recognition can be thought of as the process of **classifying spoken or written text based on what the user wishes to achieve.**

Intent Detection and Classification



How Does Intent Classification Work?

1. Data Gather

- Intent classification involves steps like data acquisition and preparation.
- Data can be in text or speech form, with speech data converted to text.
- Data from logs or consider crowdsourcing, outsourcing, or synthetic data to use as training data.
- It's a **supervised ML problem**, requiring labeled training data reflecting labels like "purchase," "account closure," or "pay bill."
- Labeling can be time-consuming.

How Does Intent Classification Work?

1. Example of Data

- class: **GetWeather** - *example: is it windy in boston, mb right now*
- class: **SearchCreativeWork** - *example:play hell house song*
- class: **BookRestaurant** - *example: book a restaurant for eight people in six hours*
- class: **PlayMusic** - *example: play the song little robin redbreast*
- class: **AddToPlaylist** - *example: add step to me to the 50 clásicos playlist*
- class: **RateBook** - *example: give 6 stars to of mice and men*
- class: **SearchScreeningEvent** - *example : find fish story*

How Does Intent Classification Work?

2. Build Model

- Intent classifiers employ the following techniques:
 1. **Rule-based** pattern matching
 2. **Machine learning classification** approaches include **decision trees, naive Bayes, and logistic regression.**
 3. **Deep learning and artificial neural networks**

How Does Intent Classification Work?

2. Build Model

- There exist pre-trained models that are open-source and available for use, namely the **(Bidirectional Encoder Representations from Transformers) BERT** model.
- Whether you use existing models or create your own, incorporating **contextual word embeddings is essential for better predictions.**
- Word embeddings represent **words as vectors**, where similar words have similar representations, learned during training.
- **Word2vec**, a powerful method, creates such embeddings using **Skip Gram** and **CBOW** techniques, both employing neural networks.

How Does Intent Classification Work?

3. Train and Validate

- Once the dataset is **processed and labeled**, the model is ready to be trained.
- After model training, it is a good idea to **test** against both a test and validation set.
- This process will test the trained model on a set of **unlabeled data** to see how well the model performs.
- The performance on the validation step is a good indication if the model needs **further adjusting or perhaps more quality data**.
- Once a model has been validated, it is ready to provide **intent recommendations**.

Common Use Cases of Intent Classification

- **Chatbots** streamline conversations to fulfill user goals.
- Intent recognition plays a pivotal role in **meeting customer service, sales, and marketing objectives**.
- The chatbot's quality depends on the training data, crucial for a useful and pleasant user experience. Therefore, a chatbot's effectiveness hinges on its ability to **understand intent and provide appropriate responses**.
- Chatbots determine user intent by **collecting data**, followed by processing it.
- Processing involves **syntax and semantic analysis** to structure text and understand context.
- Classifiers classify intent using labeled data, and chatbots generate responses based on these **predictions** in a conversational manner

Common Use Cases of Intent Classification

- Intent recognition helps identify prospects with clear **purchasing intent from emails or calls**. Prioritizing leads based on intent, such as highly interested or needing support, can significantly impact a **business's ability to scale, acquire more customers, and boost revenue**.
- Businesses often have a wealth of user data from various sources like **website interactions and phone logs**. This data is valuable for **training intent classifiers**, enabling companies to automate customer support and response times.
- For example, a telephone bot can identify user intent and direct them to the right channel, reducing the need for **human intervention** in handling common queries. Intent recognition enables **businesses to scale** and efficiently meet customer needs in customer support.

Chatbot Intent Training step

Chatbot Intent Training Steps



Steps for Intent Classification with BERT

- Load data from csv and preprocess it for training and test
- Load a BERT model from TensorFlow Hub
- Build your own model by combining BERT with a classifier
- Train your own model, fine-tuning BERT as part of that
- Save your model and use it to recognize the intend of instructions
- BERT and other Transformer encoder architectures have been shown to be successful on a variety of tasks in NLP (natural language processing).
- They compute vector-space representations of natural language that are suitable for use in deep learning models.
- The BERT family of models uses the Transformer encoder architecture to process each token of input text in the full context of all tokens before and after, hence the name: **Bidirectional Encoder Representations from Transformers.**

Intent Classification with BERT

- https://hannibunny.github.io/mlbook/transformer/intent_classification_with_bert.html
- <https://medium.com/holler-developers/intent-detection-using-sequence-models-ddae9cd861ee>

Paraphrase Extraction

What is paraphrase?

- Paraphrase is just an alternative representation of the same meaning.
- A paraphrase is a **restatement or rewording of a text or statement**, often with the goal of **conveying the original message** or idea in different words or language while **preserving its meaning**.
- Two sentences are paraphrases if their meanings are **equivalent** but their words and syntax are **different**.
- For instance:
 - Original Sentence 1: "John wrote a letter to Mary."
 - Original Sentence 2: "A dog bit John."
 - Paraphrased Sentence 1: "John wrote Mary a letter."
 - Paraphrased Sentence 2: "John was bitten by a dog."

Types of paraphrase

- **Surface Paraphrases**

- Lexical level
 - Example - solve and resolve
- Phrase level
 - Example - look after and take care of
- Sentence level
 - Example - The table was set up in the carriage shed and The table was laid under the cart-shed
- Discourse level – Group of sentence

- **Structural paraphrases**

- Pattern level
 - Example - [X] considers [Y] and [X] takes [Y] into consideration
- Collocation level
 - Example - (turn on, OBJ lighth) and (switch on, OBJ light)

Paraphrase Extraction / Generation

- Paraphrase extraction involves **identifying and extracting** pairs or sets of sentences or phrases from a text corpus that convey similar or equivalent meanings. The goal is to find different expressions or ways of saying the same thing within a given body of text.
- Paraphrase extraction is valuable in various NLP applications, including information retrieval, question answering, text summarization, and machine translation.
- In essence, paraphrase extraction aims to find and catalog different formulations of the same concept or idea within a text dataset. These extracted paraphrases can then be used to enhance the performance of various language processing tasks.

How is Paraphrase Extraction / Generation done?

- Paraphrase Generation is the process of presenting and conveying information of original sentence/phrase in alternative words and order, which may be performed through two main methods:
- **Rule-based model:** in which rules are created manually to transform original text into semantically equivalent text or paraphrases (e.g. WordNets or thesaurus for replacing words in the original text with their synonyms). This may also include changing active voice into passive, adding or deleting function words, co-reference substitution, or changing part-of-speech, among others.
- rule-based, which involves manually creating rules to transform the original text into equivalent semantic versions.

How is Paraphrase Extraction / Generation done?

2. Machine Learning based model: where paraphrases are created automatically from the data.

- **Deep Learning and Generative Adversarial Networks (GANs)**, as well as **Reinforcement Learning models** are only examples of the techniques used for automatic paraphrasing.
- In fact, paraphrasing can even be treated as a language translation challenge, often performed using a bilingual corpus pivoting back and forth.
- But, everything changed since the creation of **Transformers**, a **novel Artificial Neural Network model** that completely revolutionized the paraphrasing landscape, as well as many others NLP tasks.

Transformer-based architectures for Paraphrase Extraction / Generation

- **GPT:** Developed by OpenAI, GPT models require minimal input text to generate large, relevant, and sophisticated outputs.
- **BERT:** Google's BERT, while conceptually simple, achieves state-of-the-art results on various NLP tasks, including question answering and named entity recognition.
- **PEGASUS:** PEGASUS employs a pre-training self-supervised objective for Transformer models, excelling in abstractive summarization tasks and showing strong performance in paraphrasing.

How to choose the right model?

- Evaluation of NLP systems can be classified into intrinsic and extrinsic methods, which can be performed either automatically or manually.
- In an intrinsic evaluation, quality of NLP systems outputs is evaluated against predetermined ground truth (reference text) whereas an extrinsic evaluation is aimed at evaluating systems outputs based on their impact on the performance of other NLP systems.

Common applications where paraphrases are valuable

1. Text Augmentation:

Paraphrases can be used to expand or diversify textual data for tasks like text generation, machine translation, and text summarization. They help create more varied and contextually rich content.

2. Information Retrieval:

In search engines, paraphrases can improve query expansion and result diversification. Users might use different phrasings to search for the same information.

3. Text Simplification:

Paraphrases can simplify complex texts, making them more understandable for readers with limited language skills or cognitive abilities.

4. Plagiarism Detection:

Paraphrase detection is essential for identifying cases of plagiarism, where a piece of text is reworded to avoid detection.

5. Question Generation and Answering:

In question-answering systems, paraphrases of questions can help improve the system's ability to find relevant answers.

6. Dialogue Systems:

In chatbots and virtual assistants, paraphrases can help in providing more natural and varied responses to user queries.

7. Language Generation:

Paraphrases can be used to diversify the output of language models, making their responses more engaging and contextually appropriate.

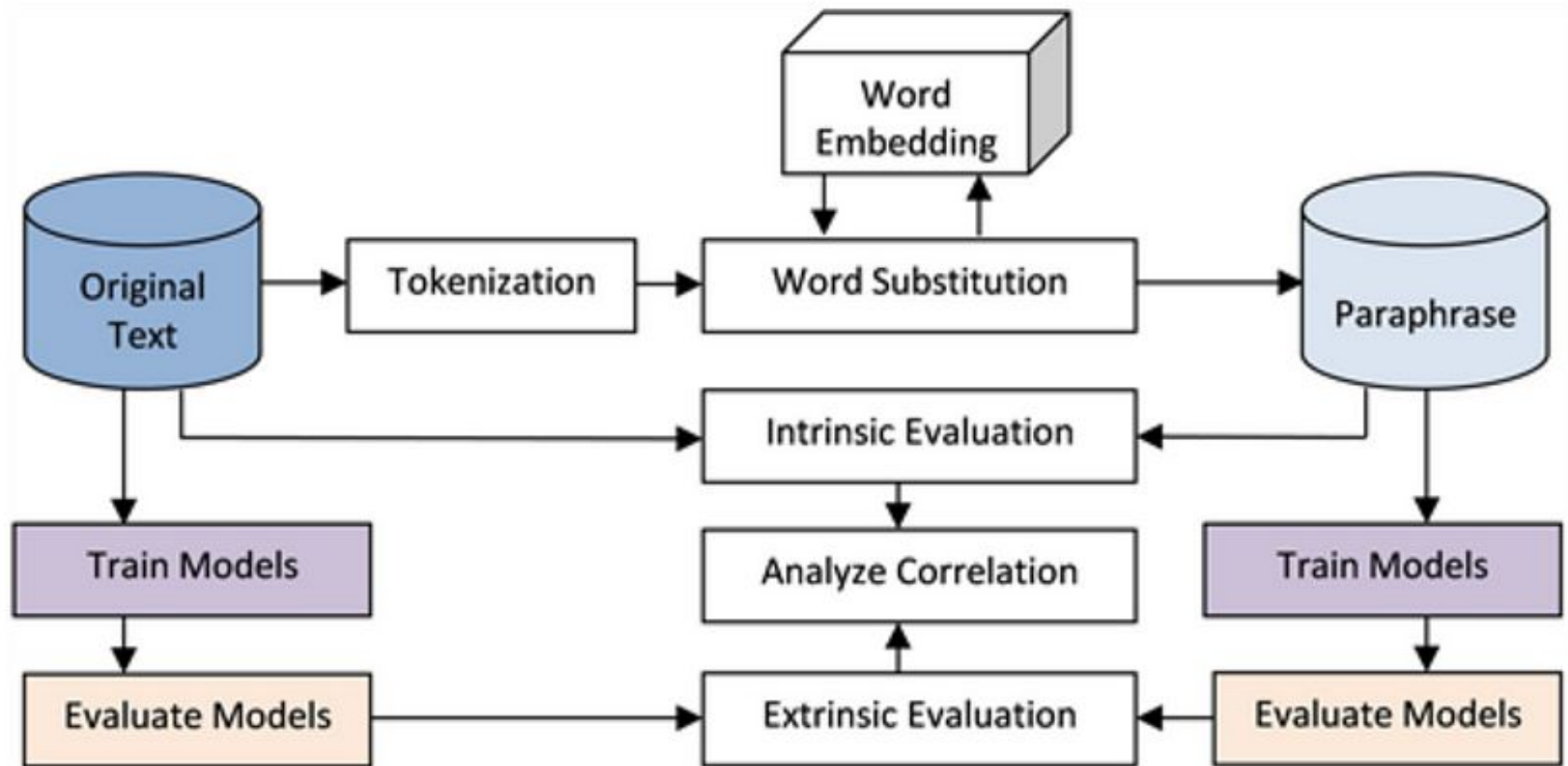
Intrinsic evaluation vs extrinsic evaluation

- **Intrinsic Evaluation:** This checks how well the generated paraphrases relate to the original text in terms of meaning and quality. It's like asking, "Are these paraphrases good on their own?"
- **Extrinsic Evaluation:** This tests if the generated paraphrases are actually useful for specific tasks, like improving question-answering or text classification. It's like asking, "Do these paraphrases help get things done?"
- You can use both methods together to get a full picture of how well the paraphrase system works.
- These metrics don't need to run separately, and can actually be integrated into the same performance method.

Intrinsic evaluation vs extrinsic evaluation

- **Intrinsic Evaluation:** This checks how well the generated paraphrases relate to the original text in terms of meaning and quality. It's like asking, "Are these paraphrases good on their own?"
- **Extrinsic Evaluation:** This tests if the generated paraphrases are actually useful for specific tasks, like improving question-answering or text classification. It's like asking, "Do these paraphrases help get things done?"
- You can use both methods together to get a full picture of how well the paraphrase system works.
- These metrics don't need to run separately, and can actually be integrated into the same performance method.

Intrinsic evaluation vs extrinsic evaluation



Intrinsic evaluation vs extrinsic evaluation

- Intrinsically they compared the generated paraphrases directly with the original text, discouraging overlapping words and encouraging the substitution of words with alternative words.
- Extrinsically, they performed two sentiment classification models (one with original sentences and the other with paraphrases), and then compared their prediction results.
- But paraphrasing is complex, and we also need to take into account things like grammatical correctness and fluency.

Applications of Paraphrase Identification

- Machine Translation
 - Simplify input sentences
 - Alleviate data sparseness
- Question Answering
 - Question reformulation
- Information Extraction
 - IE pattern expansion
- Information Retrieval
 - Query reformulation
- Summarization
 - Sentence clustering
 - Automatic evaluation
- Natural Language Generation
 - Sentence rewriting
- Others
 - Changing writing style
 - Text simplification
 - Identifying plagiarism

Discourse

“On Monday, John went to Einstein’s. He wanted to buy lunch. But the cafe was closed. That made him angry, so the next day he went to Green Street instead”

- ‘Discourse’: any linguistic unit that consists of multiple sentences.
- Discourse – The **coherent and structured groups of sentences or text** that form a meaningful unit of communication.
- Discourse refers to the larger context or structure that holds together spoken or written language.
- When we are dealing with Natural Language Processing, the provided language consists of structured, collective, and consistent groups of sentences, which are termed discourse in NLP.
- These units can be conversations, articles, documents, or any other form of text that conveys a coherent message or idea.

Discourse

- The relationship between words makes the training of the NLP model quite easy and more predictable than the actual results.
- Understanding discourse is essential for NLP tasks because it helps NLP models capture the context, relationships between words, and the flow of information in a text.
- Discourse analysis in NLP involves studying how language is structured and organized within conversations, articles, documents, or any other form of text.

Why study discourse?

- Understanding discourse is essential for NLP tasks because it helps NLP models capture the context, relationships between words, and the flow of information in a text.
- Discourse analysis in NLP involves studying how language is structured and organized within conversations, articles, documents, or any other form of text.
- **For natural language understanding:**
 - Most information is not contained in a single sentence.
 - The system has to aggregate information across paragraphs or entire documents.
- **For natural language generation:**
 - When systems generate text, that text needs to be easy to understand — it has to be coherent.
What makes text coherent?

How can we understand discourse?

“On Monday, John went to Einstein’s. He wanted to buy lunch. But the cafe was closed. That made him angry, so the next day he went to Green Street instead”

- **Understanding discourse requires (among other things):**
 - 1) doing coreference resolution:
 - ‘the cafe’ and ‘Einstein’s’ refer to the same entity.
 - He and John refer to the same person.
 - That refers to ‘the cafe was closed’.
 - 2) identifying discourse (‘coherence’) relations:
 - ‘He wanted to buy lunch’ is the reason for ‘John went to Einstein.’

Discourse Analysis / models / parsing

- Discourse parsing focuses on understanding the structure and organization of larger texts or conversations.
- It aims to identify how sentences are connected, how information flows, and how various elements contribute to discourse coherence and cohesion.
- Coreference resolution plays a critical role in discourse parsing because it helps in tracking the referential relationships between entities and maintaining coherence within a discourse.

Discourse models

- **An explicit representation of:**
 - the events and entities that a discourse talks about
 - the relations between them (and to the real world).
 - This representation is often written in some form of logic.
 - What does this logic need to capture?
- **Discourse models should capture:**
 - Physical entities: John, Einstein's, lunch
 - Events: On Monday, John went to Einstein's involve entities, take place at a point in time
 - States: It was closed. involve entities and hold for a period of time
 - Temporal relations: afterwards between events and states

key aspects of discourse

1. Coherence:

- Discourse aims to establish coherence in communication, ensuring that the **sentences within a text are logically connected** and contribute to the overall understanding of the message.

2. Context:

- Discourse analysis considers the **context in which language is used**, including the social, cultural, and situational factors that influence communication. Understanding context is crucial for **interpreting** meaning accurately.

3. Structure:

- Discourse often has a structured organization. For example, in a written article, there may be an introduction, body paragraphs, and a conclusion. In a conversation, there are typically turns or exchanges between speakers.

4. Flow of Information:

- Discourse analysis studies **how information flows within a text or conversation**. It examines how sentences or segments build upon each other to convey ideas and maintain continuity.

key aspects of discourse

5. Coherence Devices:

- Discourse analysis looks at linguistic devices and markers used to establish relationship between sentences, such as transitional phrases ("however," "therefore"), pronouns, and conjunctions (but, for, or, so etc).

6. Anaphora and Coreference:

- Anaphora and coreference resolution are important aspects of discourse analysis. They involve determining what pronouns or references (e.g., "he," "it") refer to within a text or conversation. E.g. the word “it” in the sentence “she wanted it” depends upon the prior discourse context Discourse Integration

7. Discourse Structure:

- Identifying discourse structure can involve segmenting text into paragraphs, dialogue turns, or topic shifts, helping to organize and understand the content better.

8. Pragmatics:

- Discourse analysis also delves into pragmatics, which deals with how language is used in context. This includes understanding implied meaning, indirect speech acts, and the role of context in interpretation.

Coreference Resolution

- Coreference resolution is the task of **finding all referring expressions** like — (he, I, that, this..., or any subject or noun / called **mentions**) is referred to which entity (referents like any person, thing, subject etc...)
- After finding and grouping these **mentions** we can resolve them by replacing, as stated above, pronouns with noun phrases.
- **Example:**

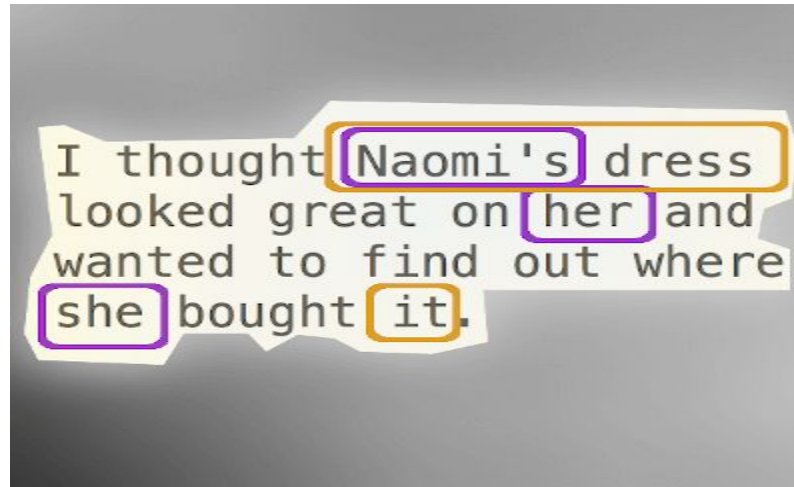
"I voted for Trump because he was most aligned with my values", John said.

The original sentence

"John voted for Trump because Trump was most aligned with John's values", John said.

The sentence with resolved coreferences

Coreference Resolution - Example



There are **two entities** in this sentence:

1.Naomi

2.Naomi's dress

- Naomi, her and she all refer to a single entity.
- Naomi's dress and it both refer to another entity.

Our brains will instantly recognize this, but a computer will not.

Demo:

<https://lvngd.com/blog/coreference-resolution-python-spacy-neuralcoref/#:~:text=Coreference%20resolution%20is%20a%20task,Naomi>

Coreference Resolution

- Coreference resolution is an exceptionally versatile tool and can be applied to a variety of NLP tasks such as text understanding, information extraction, machine translation, sentiment analysis, or document summarization.
- It is a great way to **obtain unambiguous sentences** which can be much more easily understood by computers.

Types of Coreference resolution

- **Anaphora**
- **Cataphora**
- **Split Antecedents**

Types of Coreference resolution

- **Anaphora** - “Anaphora is the use of an expression whose interpretation depends specifically upon another (antecedent) expression” or you can say “when the referring expression(anaphor) is pointing backwards”
- *Example:- **The music** was so loud that **it** couldn't be enjoyed*
- The word "**it**" is an example of an anaphor.
- It's a pronoun that refers back to the noun "**music**" mentioned earlier in the sentence.
- So, "it" is used to avoid repeating the word "music" and still make the sentence clear.

Types of Coreference resolution

- **Cataphora** - its said to be just reverse of anaphora → “the use of an expression that depends upon a **postcedent expression**” or you can say “when the referring expression is pointing forward”
- Example:- *When **he** arrived home, **John** went to sleep.*
- **he** is referred to **John**, and “he” came before “John” in sentence.

Types of Coreference resolution

- **Split antecedents:** It's an anaphoric expression where the pronoun (2) refers to more than one antecedent (1).

Edison and Tesla ① were both inventors. They ② were also the greatest rivals.

Some other examples of coreference resolution

- **Example of named mentions instead of pronouns:-**
 1. **International Business Machines** sought patent compensation from Amazon; **IBM** had previously sued other companies.
- Well you can see IBM referred to International Business Machines.... these type of references are also there..
- 2. **Barack Obama** traveled to **Obama** ...
- So we can see “obama” and “Barack Obama” are referred to same person.

What is coreference resolution?

- Coreference resolution thus comprises **two tasks** (although they are often performed jointly):
 - (1) identifying the mentions
 - (2) clustering them into coreference chains/discourse entities

Let's see another example

“Victoria Chen, CFO of Megabucks Banking, saw her pay jump to \$2.3 million, as the 38-year-old became the company’s president. It is widely known that she came to Megabucks from rival Lotsabucks”

Let's make cluster of above example:

1. Victoria Chen, her, the 38-year-old, She
2. Megabucks Banking, the company, Megabucks
3. her pay
4. Lotsabucks

What is coreference resolution?

Let's take a real life example of question answer engine:-

Content provided to QnA engine →

“Joseph Robinette Biden Jr. is an American politician who is the 46th and current president of the United States. A member of the Democratic Party, he served as the 47th vice president from 2009 to 2017 under Barack Obama and represented Delaware in the United States Senate from 1973 to 2009.”

Now if we ask question → “Who served as the 47th vice president from 2009 to 2017?”

In QnA engine **without reference resolution** it will give us → **“he”** but that is not the answer we want right? We know here that **“he”** is referred to **Joseph Robinette Biden.** and we want his name as an answer right..? That's where coreference resolution comes in. It let's us knows that he was referred to Joseph Robinette Biden so that we work something out and replace **“he”** or any other mentions, behind the scene. So that we get appropriate answer.

<https://blog.devgenius.io/coreference-resolution-nlp-python-584c2ec50f5d>

<https://medium.com/huggingface/how-to-train-a-neural-coreference-model-neuralcoref-2-7bb30c1abdfc>

Coreference resolution task

- The **coreference resolution task** is separated into two sub tasks:
 1. Mention Detection
 2. Mention Clustering

Coreference resolution task - Mention Detection

Mention Detection:

- In this sub-task the main goal is to find all the candidates spans referring to some entities.
- For example, in the sentence below the mention detection step will color all the candidates spans in blue:

"Emma said that she thinks that Nelson really likes to dance, because he goes to the dancing studio 4 times a week."


- There are **three kinds of mentions** that will be detected in this step: **Pronoun, Named entity recognition, Noun phrases**

Coreference resolution task - Mention Detection

1. Pronouns:

- A pronoun is a word that substitutes for noun phrase and usually involves anaphora, where the meaning of the pronoun is dependent on an antecedent.
- For instance, ‘**She**’ is the pronoun in the sentence “Noa gave an amazing lecture, when she was in the conference at Madrid last year.”

Coreference resolution task - Mention Detection

ENGLISH PRONOUNS						
	Subject Pronouns	Object Pronouns	Possessive Adjectives	Possessive Pronouns	Reflexive Pronouns	
	1st person	I	me	my	mine	myself
	2nd person	you	you	your	yours	yourself
	3rd person (male)	he	him	his	his	himself
	3rd person (female)	she	her	her	hers	herself
	3rd thing	it	it	its	(not used)	itself
	1st person (Plural)	we	us	our	ours	ourselves
	2nd person (Plural)	you	you	your	yours	yourself
	3rd person and thing	they	them	their	theirs	themselves

Coreference resolution task - Mention Detection

2. Named-entity recognition (NER):

NER model locates entities in unstructured text and classifies them into pre-defined categories (such as person names, organizations, locations, products, etc).

For example:

"There was nothing about this storm that was as expected," said **Jeff Masters**, a meteorologist and founder of **Weather Underground**. "**Irma** could have been so much worse. If it had traveled 20 miles north of the coast of **Cuba**, you'd have been looking at a (Category) 5 instead of a (Category) 3."

Person

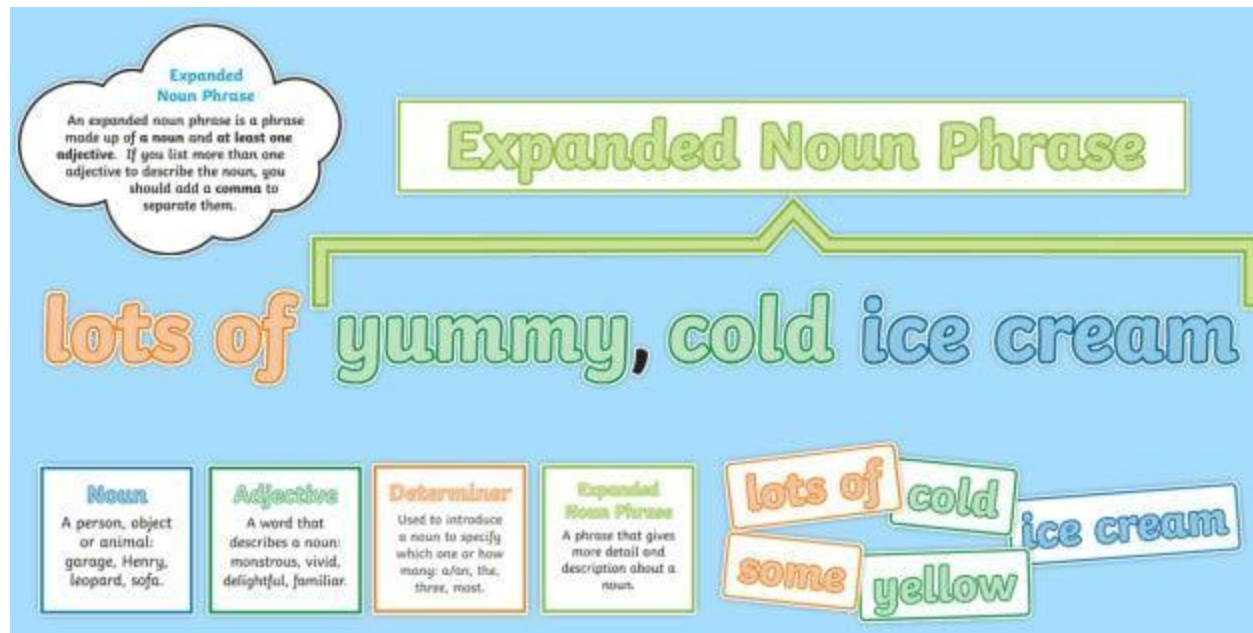
Organization

Location

Coreference resolution task - Mention Detection

3. Noun-phrases:

A noun phrase is a bunch of words that is headed by a noun and includes modifiers (e.g., 'the,' 'a,' 'of them,' 'with her').



Coreference resolution task - Mention Clustering

- Once we hold the mentions, the goal of the second sub-task is attempting to identify which ones refer to the same entity. Then, merging the mentions into the cluster corresponding to the entities presented in the text.
- **Antecedent, Anaphora, Cataphora are the basic terms of mention clustering.**

*“Gal came back late from the party, because **she** really enjoyed there.”*

Antecedent — An expression in the text that dictates the meaning to all the rest pronouns. For instance, in the example above, the pronoun *she* takes its meaning from *Gal*, so *Gal* is the antecedent of *she*.

Anaphora — Refers to an expression which its interpretation depends upon another expression in context (its antecedent). For example,

*“If you want a **cake**, there is **some** in the kitchen.”*

Cataphora — Cataphora is a type of anaphora but the pronoun appears earlier than the noun that it refers to. For example,

*“If you want **some**, there is a **cake** in the kitchen.”*

In the sentence above the pronoun *some* appears before the noun *cake*.

Coreference resolution Models – Rule based models

Types of coreference resolution models

- 1. Rule-Based Models
- 2. Mention-Pair Models
- 3. Mention-Ranking Models

1. Rule-Based Models

At 1976 Jerry Hobbs wrote a naive algorithm on coreference resolution. The problem with Hobbs rule-based attitude was that the rules couldn't capture both cases like the examples below:

“She poured water from the pitcher into the **cup** until **it** was **full**”

“She poured water from the **pitcher** into the cup until **it** was **empty**”

While in the first one “it” refers to the “cup” and in the second one “it” refers to the “pitcher”. It will work just for one of them.

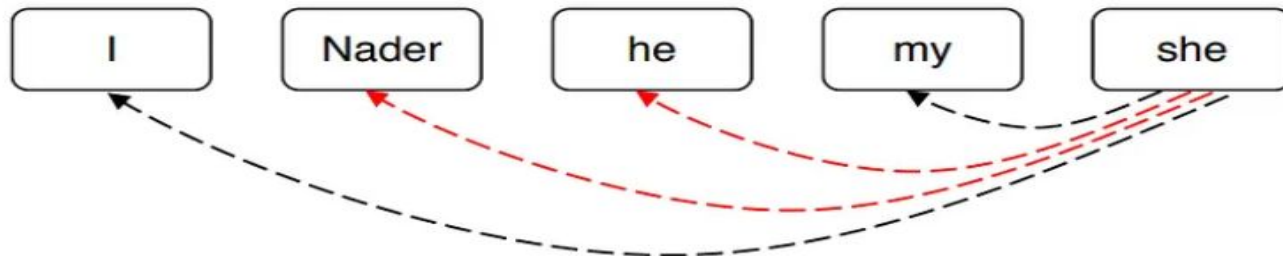
Coreference resolution Models - Mention-Pair Models

- The second kind of coreference model (Mention-Pair Models) is a binary classifier.
- The idea is to train a classifier that will find the probability of two mentions to be coreferent.
- It will compare between each two mentions in the sentence and will assign their probability to be connected. The ideal condition will be that negative examples will receive a probability that is closed to 0 and positive examples will be close to 1.

Coreference resolution Models - Mention-Pair Models

- Train a binary classifier that assigns every pair of mentions a probability of being coreferent: $p(m_i, m_j)$
 - e.g., for “she” look at all **candidate antecedents** (previously occurring mentions) and decide which are coreferent with it

*“I voted for **Nader** because **he** was most aligned with **my** values,” **she** said.*



Negative examples: want $p(m_i, m_j)$ to be near 0

CS224n: Natural Language Processing with Deep Learning

The model uses cross entropy loss for its training. The test will get a sentence and a set of mentions and it will predict for each two if they are coreferent or not. In the end of the procedure it will add coreference links between mention pairs that are above some threshold. It can also add transitive links and it will create the clusters.

Coreference resolution Models - Mention-Pair Models

Mention Pair Training

- N mentions in a document
- $y_{ij} = 1$ if mentions m_i and m_j are coreferent, -1 if otherwise
- Just train with regular cross-entropy loss (looks a bit different because it is binary classification)

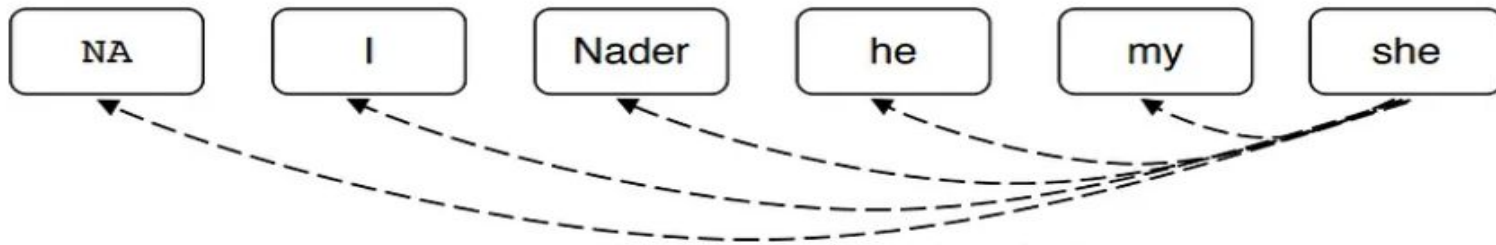
$$J = - \sum_{i=2}^N \sum_{j=1}^i y_{ij} \log p(m_j, m_i)$$

The diagram includes three arrows pointing from explanatory text to the equation:

- An arrow from "Iterate through mentions" points to the $i=2$ index in the first summation.
- An arrow from "Iterate through candidate antecedents (previously occurring mentions)" points to the $j=1$ index in the second summation.
- An arrow from "Coreferent mentions pairs should get high probability, others should get low probability" points to the $\log p(m_j, m_i)$ term.

Coreference resolution Models - Mention-Ranking Models

- The mention-ranking model assigns a score to each pair of candidate antecedent and coreferents. In the end of the procedure the model will choose just one of the pairs



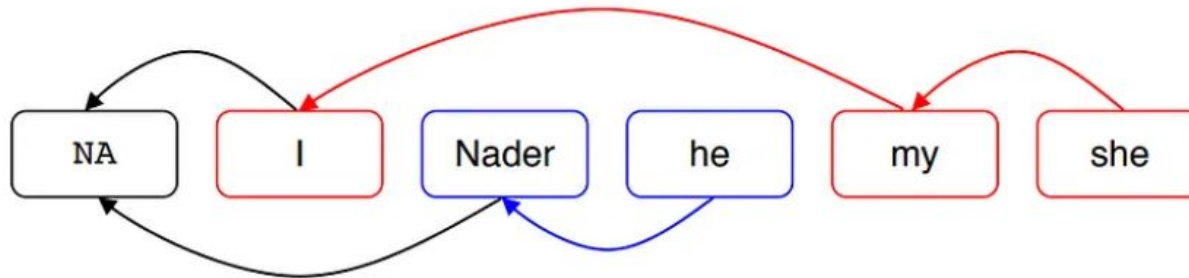
CS224n: Natural Language Processing with Deep Learning

$p(\text{NA}, \text{she}) = 0.1$
 $p(\text{I}, \text{she}) = 0.5$
 $p(\text{Nader}, \text{she}) = 0.1$
 $p(\text{he}, \text{she}) = 0.1$
 $p(\text{my}, \text{she}) = 0.2$

} Apply a softmax over the scores for candidate antecedents so probabilities sum to 1

As a result of the sequence of many local decisions of referents that were linked together, a global structure (cluster) will be created:

Coreference resolution Models - Mention-Ranking Models



CS224n: Natural Language Processing with Deep Learning

Mathematically it will look like the term below:

$$\sum_{j=1}^{i-1} \mathbb{1}(y_{ij} = 1) p(m_j, m_i)$$

Iterate through candidate antecedents (previously occurring mentions)

For ones that are coreferent to m_j ...

...we want the model to assign a high probability

Coreference resolution Models - Mention-Ranking Models

M_j is our current mention and M_i is the candidate antecedent, while we will want to maximize the probability between them. In the training level it will be part of the loss function that we will try to optimize:

$$J = \sum_{i=2}^N -\log \left(\sum_{j=1}^{i-1} \mathbb{1}(y_{ij} = 1) p(m_j, m_i) \right)$$

Iterate over all the mentions
in the document

Usual trick of taking negative
log to go from likelihood to loss

There are few ways to compute the coreferent probabilities between M_i and M_j :

1. Non-Neural Statistical Classifier
2. Simple Neural Network
3. Advanced Models using LSTMs, Attention

<https://galhever.medium.com/a-review-to-coreference-resolution-models-f44b4360a00>

Coreference Resolution vs Anaphora Resolution

- **Coreference Resolution (CR):**
 - This is like connecting the **dots between words or phrases** that talk about the same thing. Imagine you're reading a story, and the story mentions "Mary" and later says "She did something." CR helps us figure out that "**She**" is talking about "**Mary**."
 - **It's like finding out that they're both talking about the same person.**
- **Anaphora Resolution (AR):**
 - **This is a specific type of CR.**
 - It's like solving a little puzzle in a story. Anaphora means words like "he," "she," "it," or "they" that refer to something already mentioned. For example, if a story says, "**John found a wallet. He returned it,**" AR helps us know that "He" is talking about "John" and "It" is talking about the "wallet."
- So, CR is like connecting any two things that are talking about the same stuff, while AR is like solving the puzzle of what words like "he" and "it" are talking about.

Coreference Resolution vs Anaphora Resolution

Aspect	Coreference Resolution (CR)	Anaphora Resolution (AR)
What it does	Connects words/phrases about the same thing	Focuses on words like "he," "she," "it," etc., and figures out what they refer to in the text
Scope	Broader - Connects any related expressions	Specific - Deals mainly with pronouns and similar words that refer back to something mentioned earlier
Example (CR)	"Mary bought a car. She loves it."	CR identifies "She" and "it" refer to "Mary" and the "car."
Example (AR)	"John found a wallet. He returned it."	AR focuses on "He" referring to "John" and "It" referring to the "wallet."

Discourse vs Coreference

- Discourse and coreference are related through coreference resolution, which is a key subtask for understanding the structure and meaning of text.
- Coreference resolution is essential for building coherent NLP applications, including discourse parsing, information extraction, machine translation, and question answering systems, as it helps in maintaining referential coherence within a text or conversation.

Coherence

- Coherence refers to the logical and meaningful flow of information in a text or discourse. It encompasses how sentences, paragraphs, or other units of text are organized and connected to convey a clear and understandable message.
- Coherence is essential for effective communication and understanding in written and spoken language.

Coherence

Types of coherence:

- Textual coherence
- Logical coherence
- Topic coherence
- Contextual coherence

Coherence

Textual coherence: Cohesion refers to the grammatical and lexical relationships that tie sentences and paragraphs together. It includes cohesive devices such as pronouns, conjunctions, transitional words, and lexical repetition. These devices help establish connections between ideas and ensure that the text flows smoothly.

- **Ex:**

- **Pronoun Cohesion:** "John went to the store. He bought some groceries."
In this example, the pronoun "He" refers back to "John," creating cohesion.
- **Conjunction Cohesion:** "I like both ice cream and cake. However, I'm trying to eat healthier." The conjunction "however" establishes cohesion between the contrasting ideas.

Coherence

Logical coherence: Coherent texts have a clear and logical structure, typically following a specific organization pattern. This may involve using a chronological order, cause-and-effect relationships, problem-solution structures, or other logical frameworks that help readers or listeners follow the narrative or argument.

Ex:

- **Chronological Order:** "First, we gathered the ingredients. Then, we mixed them together. Finally, we baked the cake."
- **Cause-and-Effect:** "Because it rained heavily, the streets were flooded."

Coherence

Topic coherence / Topic continuity : Coherent texts maintain a consistent and well-defined topic or theme throughout. This ensures that the information presented is relevant to the central idea and avoids abrupt shifts in subject matter that can confuse readers or listeners.

Ex:

- "The benefits of exercise include improved health, increased energy levels, and weight management. Regular physical activity can lead to a longer and healthier life."

Coherence

Contextual coherence / References: Coherence relies on the appropriate use of references to previously mentioned entities, concepts, or ideas. Anaphoric references (e.g., pronouns like "he" or "it") and cataphoric references (e.g., pronouns that refer to something mentioned later) should be used consistently and clearly to maintain coherence.

Ex:

- "The dog chased its tail. Then, it barked loudly." The pronoun "it" maintains reference to the dog.
- "After turning the corner, she saw a beautiful garden. The flowers there were in full bloom." The word "there" refers to the garden mentioned earlier.

Coherence and discourse

- Coherence and discourse structure are interconnected in many ways.
- Coherence plays a crucial role in maintaining the **overall quality of discourse**.
- To create a coherent discourse, the individual sentences or elements within it need to be logically connected.
- If the sentences don't flow smoothly or if there are abrupt shifts in topics or ideas, the discourse may be perceived as incoherent or difficult to follow.
- Achieving coherence in discourse involves using various linguistic and structural techniques. This includes the use of transitional words and phrases, maintaining thematic continuity, establishing clear relationships between ideas, and ensuring that pronouns and references are used consistently. All of these elements work together to make the discourse coherent and comprehensible.

Coherence and discourse

- *Coherence is a fundamental quality that contributes to the effectiveness of discourse.*
- Discourse is the larger context or container for language communication, and coherence is the means by which the various elements within that discourse are connected and organized to convey a clear and meaningful message.

Text Coherence

Text coherence

- Text coherence in NLP refers to the degree to which the different parts of a text are connected and make sense together.
- A coherent text is easy to read and understand, and the different sentences and paragraphs are related to each other in a meaningful way.

Text coherence

- There are a number of factors that contribute to text coherence, including
 - **Logical coherence:** This refers to the logical relationships between the different sentences and paragraphs in a text. For example, a coherent text will have a clear beginning, middle, and end, and the different parts of the text will be logically connected.
 - **Semantic coherence:** This refers to the meaning of the text. A coherent text will have a clear and consistent meaning, and the different sentences and paragraphs will all contribute to this meaning.
 - **Pragmatic coherence:** This refers to the context in which the text is produced and interpreted. A coherent text will be appropriate for the audience and the purpose of communication.

Text coherence

- Text coherence is important in NLP because it is essential for understanding the meaning of a text.
- A coherent text is easier to read and understand than a non-coherent text. Additionally, text coherence is important for a variety of NLP tasks, such as machine translation, summarization, and question answering.
- A coherent news article will have a clear headline, a summary of the main points, and supporting evidence.
- A coherent scientific paper will have a clear introduction, a methodology section, a results section, and a discussion section.
- A coherent novel will have a clear plot, developed characters, and a satisfying ending.

Text coherence – Example 1

- Input text “**The cat sat on mat. The mat was red . The cat was black**”
- This **text is coherent** because the different sentences are all related to each other. The first sentence introduces the cat and the mat.
- The second sentence describes the color of the mat. The third sentence describes the color of the cat.
- All of the sentences are necessary to understand the complete meaning of the text.

Text coherence – Example 2

- Input text **“The cat sat on the mat. The ball was red. The sky was blue.”**
- This text is **incoherent** because the different sentences are not related to each other. The first sentence introduces the cat and the mat.
- The second sentence describes the color of the ball. The third sentence describes the color of the sky.
- The second and third sentences are not necessary to understand the complete meaning of the text.

Text coherence – Example 3

- Input text “The man went to the store. He bought some milk. He went home.”
- This **text is coherent** because the different sentences describe a sequence of events. The first sentence tells us that the man went to the store.
- The second sentence tells us that he bought some milk. The third sentence tells us that he went home.
- The order of the sentences is important, and all of the sentences are necessary to understand the complete meaning of the text.

Discourse Structure

- Discourse structure is a term used to describe the way in which an entire text is organised – for example, how language is used in a poem, in a newspaper article, or in a speech designed to read aloud.
- “Discourse structure” refers to the *organization and structure of language beyond the level of individual sentences*.
- It focuses on understanding how sentences, paragraphs, or utterances are connected and organized to create coherent and meaningful communication.
- Discourse structure analysis in NLP aims to **capture the relationships and coherence between different parts of a text or conversation**.

Discourse Structure

Discourse Structure includes:

- 1. Coherence Relations:** NLP systems analyze the relationships between sentences or segments of text to identify how they are logically connected. This involves determining whether sentences provide reasons, contrasts, explanations, or other forms of logical connections to one another.
- 2. Anaphora Resolution:** Discourse structure analysis involves resolving anaphoric references, such as pronouns and demonstratives, to determine what they refer to. For example, understanding that "he" in one sentence refers to "John" mentioned earlier in the text.
- 3. Coreference Resolution:** Similar to anaphora resolution, coreference resolution identifies when different expressions or words in a text refer to the same entity or concept. For instance, recognizing that "the company" and "it" both refer to the same organization in a document.

Discourse Structure

Discourse Structure includes:

- 4. Segmentation:** Identifying boundaries between different segments or topics within a text is crucial for understanding how a discourse is organized. This involves recognizing the start and end of paragraphs, sections, or topics.
- 5. Discourse Markers:** Discourse markers like "however," "therefore," and "for example" are used to signal shifts in thought or transitions between ideas. Recognizing these markers helps in understanding the overall discourse structure.
- 6. Thematic Progression:** NLP systems may analyze how themes or topics progress within a text. This involves identifying when a text introduces, develops, or concludes discussions on particular subjects.
- 7. Cohesion Analysis:** Cohesive devices, such as pronouns, lexical repetition, and transitional words, are analyzed to determine how they contribute to the overall coherence of the discourse.

Discourse Planning

- Discourse planning, a subfield of AI and NLP, centers on **producing coherent and cohesive text or speech**.
- It explores how to generate language that is both logically connected and smoothly flowing, considering the context at hand. This process takes into account the speaker's objectives, as well as the social dynamics of the conversation.
- In NLP, discourse planning involves structuring and arranging language components to facilitate clear and meaningful communication in longer texts or spoken interactions. It plays a vital role in tasks such as text generation, summarization, dialogue systems, and machine translation.

Discourse Planning

- Here are the key contents of discourse planning in NLP:

1. Context Understanding:

- Understanding the context in which the language will be used, including the speaker's goals, intentions, and the social context of the communication.

2. Content Generation:

- Generating the content or information that needs to be communicated, which can include ideas, facts, opinions, or responses to specific situations.

3. Structuring the Discourse:

- Organizing the generated content in a structured manner, determining the overall structure of the communication (e.g., introduction, body, conclusion).

4. Thematic Progression:

- Ensuring that information is presented in a logical and progressive manner, with a clear thematic flow from one point to another.

5. Coherence and Cohesion:

- Maintaining coherence by establishing logical connections between ideas and ensuring cohesion through the use of cohesive devices like pronouns, transitional words, and references.

Discourse Planning

- Here are the key contents of discourse planning in NLP:

6. Contextual Considerations:

- Adapting the language, style, and content to suit the context, including the characteristics and expectations of the audience.

7. Rhetorical Strategies:

- Employing rhetorical strategies, such as comparison, contrast, cause-and-effect, or problem-solution, to structure and convey the message effectively.

8. Discourse Markers:

- Using discourse markers (e.g., "however," "therefore") to signal transitions between ideas, shifts in thought, and the overall organization of information.

9. Anaphora and Coreference Resolution:

- Managing anaphoric references (e.g., pronouns like "he," "she") and resolving coreference (identifying when different expressions refer to the same entity) to maintain clarity and coherence.