

# Unit-III

## **Semantic and Discourse Analysis**

# Topics

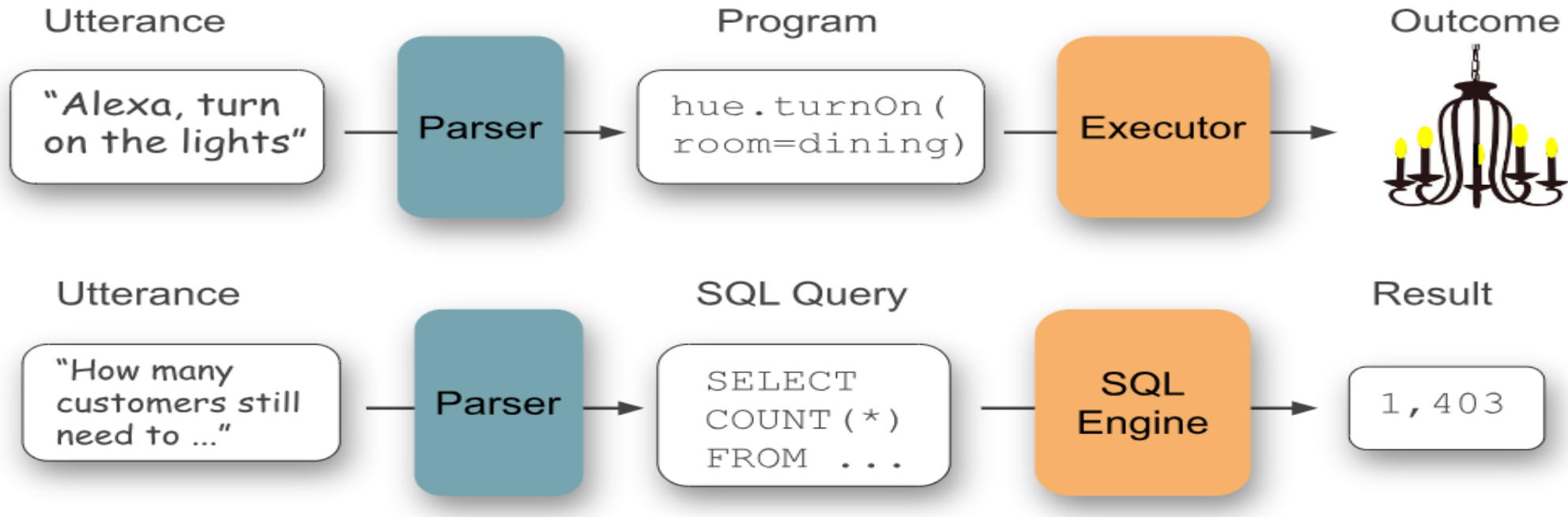
- Representing Meaning,
- Lexical Semantics,
- Word Senses,
- Relation between Senses,
- Word Sense Disambiguation,
- Word Embeddings,
- Word2Vec,
- CBOW,
- Skip-gram and GloVe,
- Discourse Segmentation,
- Text Coherence,
- Discourse Structure,
- Reference Resolution,
- Pronominal Anaphora Resolution,
- Coreference Resolution

# Semantic

## Semantic- Meaning

- Semantic parsing **analyze meaning** of the text from sentences, phrases, or complete texts.
- It goes beyond grammatical structure
- Aims to capture the semantics or the actual message that the language conveys.
- The role of this is to detect all the subjective elements in an exchange: approach, positive feeling, dissatisfaction, impatience, etc.
- Ex of sentence with semantic error:
  - The cat is a type of dog
  - The sun rises in the west
  - He drank the entire ocean
  - The car sings beautifully

# Semantic



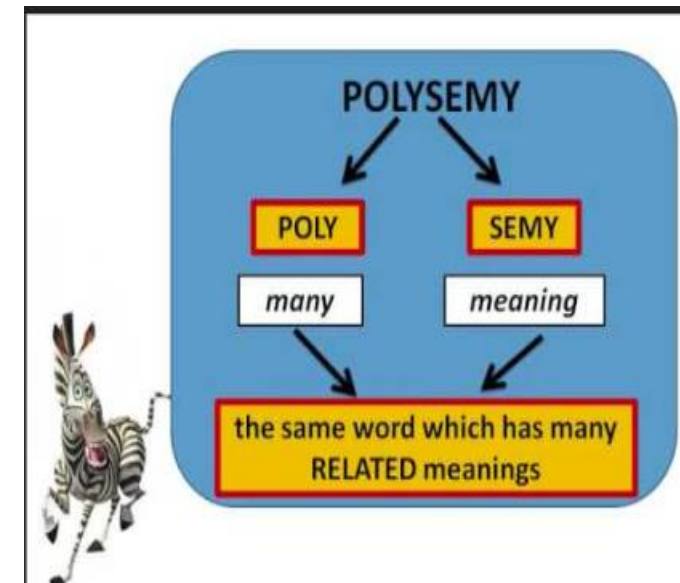
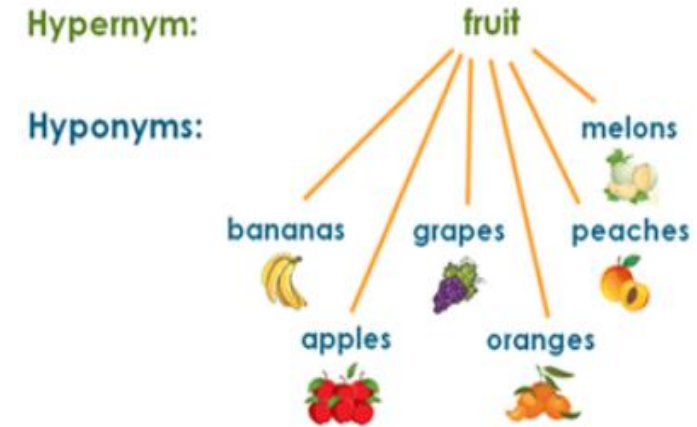
- Semantic parsing translating natural language statements into some executable meaning representation.
- Semantic parsers form the backbone of voice assistants, as shown above, or they can be used to answer questions or give natural language interfaces to databases.
- Video Link : [How Voice Recognition Works-  
https://www.youtube.com/watch?v=2RRT1YuyBCo](https://www.youtube.com/watch?v=2RRT1YuyBCo)

# How Does Semantic Analysis Work?

- Semantic analysis starts with lexical semantics, which studies individual words' meanings (i.e., dictionary definitions).
- Semantic analysis then examines relationships between individual words and analyzes the meaning of words that come together to form a sentence.
- This analysis provides a clear understanding of words in context.
- For example, it provides context to understand the following sentences:
  - “The boy ate the apple” defines an apple as a fruit.
  - “The boy went to Apple” defines Apple as a brand or store.

# Elements of Semantic Analysis

- A semantic system brings entities, concepts, relations and predicates together to provide more context to language so machines can understand text data with more accuracy.
- To better understand this, consider the following elements of semantic analysis that help support language understanding:
- **Hyponymy:** It is an instance of a generic term
  - For example: 'Color' is a hypernymy while 'grey', 'blue', 'red', etc, are its hyponyms.
- **Homonymy:** Two or more lexical terms with the same spelling and different meanings.
  - For example: 'Rose' might mean 'the past form of rise' or 'a flower', – same spelling but different meanings; hence, 'rose' is a homonymy.
- **Polysemy:** Two or more terms that have the same spelling but multiple closely related meanings.
  - It differs from homonymy because the meanings of the terms need not be closely related in the case of homonymy.
  - For example: 'man' may mean 'the human species' or 'a male human' or 'an adult male human' – since all these different meanings bear a close association, the lexical term 'man' is a polysemy.



# Elements of Semantic Analysis

- **Synonymy:** Two or more lexical terms with different spellings and similar meanings.
  - For example: (Job, Occupation), (Large, Big), (Stop, Halt)
- **Antonymy:** A pair of lexical terms with contrasting meanings.
  - they are symmetric to a semantic axis. For example: (Day, Night), (Hot, Cold), (Large, Small)
- **Meronymy:** A relationship between a lexical term and a larger entity.
  - Meronymy refers to a relationship wherein one lexical term is a constituent of some larger entity.
  - For example: 'Wheel' is a meronym of 'Automobile'

## Synonyms for Kids

 <b>Listen</b> Hear	 <b>Old</b> Ancient	 <b>Jungle</b> Forest
 <b>Write</b> Record	 <b>Nearly</b> Almost	 <b>Walk</b> Stroll
 <b>Rich</b> Wealthy	 <b>All</b> Every	 <b>End</b> Finish

## Kinds of Meronymy

Component-object	<b>Head - Body</b>
Staff-object	<b>Wood - Table</b>
Member-collection	<b>Tree - Forest</b>
Feature-Activity	<b>Speech - Conference</b>
Place-Area	<b>Palo Alto - California</b>
Phase-State	<b>Youth - Life</b>
Resource-process	<b>Pen - Writing</b>
Actor-Act	<b>Physician - Treatment</b>

# Part of Semantic Analysis

- **Lexical analysis** is the process of reading a stream of characters, identifying the lexemes and converting them into tokens that machines can read.
- **Grammatical analysis** correlates the sequence of lexemes (words) and applies formal grammar to them so part-of-speech tagging can occur.
- **Syntactical analysis** analyzes or parses the syntax and applies grammar rules to provide context to meaning at the word and sentence level.
- **Semantic analysis** uses all of the above to understand the meaning of words and interpret sentence structure so machines can understand language as humans do.



# Where does Semantic Analysis Work?



**Natural Language Processing (NLP)**



**Search Engines**



**Information Retrieval**



**Chatbots and Virtual Assistants**



**Machine Learning and AI**



**Customer Service and Support**

# Syntax vs Semantic

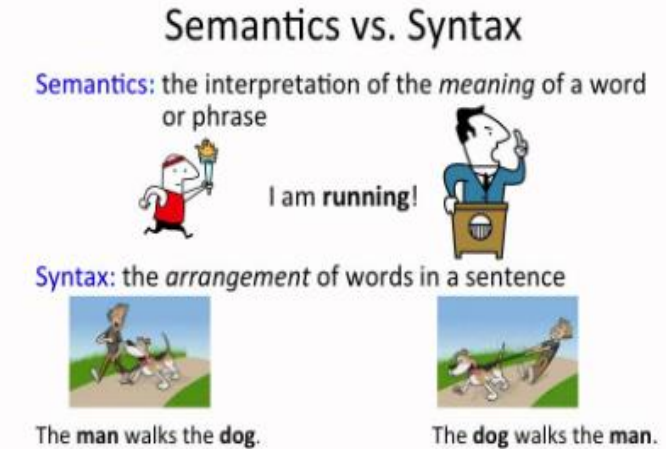
- Semantic” refers to meaning.
- Syntactic analysis focuses on “form” and syntax, meaning the relationships between words in a sentence.
- Semantic analysis focuses on “meaning,” or the meaning of words together and not just a single word.
- “parsing” means resolving a sentence into its component parts.

For example, we can build a parser that converts the natural language query “*Who was the first person to walk on the moon?*” to an equivalent (although complex!)

SQL query such as “SELECT name FROM Person WHERE moon\_walk = true ORDER BY moon\_walk\_date FETCH first 1 rows only.”

Semantic parsing is inherently more complicated than syntactic parsing because it requires understanding concepts from different word phrases.

For instance, the following sentences (adapted from [4]) should ideally map to the same formal representation.



# Semantics



## *What is Semantics?*

**The study of meaning:** Relation between symbols and their denotata.  
John told Mary that the train moved out of the station at 3 o'clock.



# Computational Semantics

## *Computational Semantics*

The study of how to automate the process of constructing and reasoning with meaning representations of natural language expressions.

*Methods in Computational Semantics generally fall in two categories:*

- **Formal Semantics:** Construction of precise mathematical models of the relations between expressions in a natural language and the world.  
*John chases a bat  $\rightarrow \exists x[bat(x) \wedge chase(john, x)]$*
- **Distributional Semantics:** The study of statistical patterns of human word usage to extract semantics.

# Semantic Parsing

## *Lexical Semantics*

### *Definition*

**Lexical semantics** is concerned with the systematic meaning related connections among lexical items, and the internal meaning-related structure of individual lexical items.

To identify the semantics of lexical items, we need to focus on the notion of **lexeme**, an individual entry in the lexicon.

### *What is a lexeme?*

**Lexeme** should be thought of as a pairing of a particular orthographic and phonological form with some sort of symbolic meaning representation.

- Orthographic form, and phonological form refer to the appropriate form part of a lexeme
- Sense refers to a lexeme's meaning counterpart.

## *Meronyms and holonyms*

### *Definition*

**Meronymy:** an asymmetric, transitive relation between senses.

$X$  is a **meronym** of  $Y$  if it denotes a part of  $Y$ .

The inverse relation is **holonymy**.

meronym	holonym
porch	house
wheel	car
leg	chair
nose	face

# 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.

# Synonymy

*Words that have the same meaning in some or all contexts.*

- filbert / hazelnut
- couch / sofa
- big / large
- automobile / car
- vomit / throw up
- water /  $H_2O$

Two lexemes are synonyms if they can be successfully substituted for each other in all situations.

# Synonyms



## *Shades of meaning*

- What is the cheapest first class *fare*?
- \*What is the cheapest first class *price*?

## *Collocational constraints*

- We frustate 'em and frustate 'em, and pretty soon they make a *big* mistake.
- \*We frustate 'em and frustate 'em, and pretty soon they make a *large* mistake.

# 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**.

# Antonyms

- Senses that are opposites with respect to one feature of their meaning
- Otherwise, they are similar!
  - dark / light
  - short / long
  - hot / cold
  - up / down
  - in / out

*More formally: antonyms can*

- define a binary opposition or at opposite ends of a scale (*long/short, fast/slow*)



# 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:

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

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.

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

# Homonymy

## Definition

**Homonymy** is defined as a relation that holds between words that have the same form with unrelated meanings.

## Examples

- Bat (wooden stick-like thing) vs Bat (flying mammal thing)
- Bank (financial institution) vs Bank (riverside)

## homophones and homographs

**homophones** are the words with the same pronunciation but different spellings.

- write vs right
- piece vs peace

**homographs** are the lexemes with the same orthographic form but different meaning. Ex: bass

## *Problems for NLP applications*

### *Text-to-Speech*

Same orthographic form but different phonological form

### *Information Retrieval*

Different meaning but same orthographic form

### *Speech Recognition*

to, two, too

*Perfect homonyms are also problematic*



## *Problems for NLP applications*

### *Text-to-Speech*

Same orthographic form but different phonological form

### *Information Retrieval*

Different meaning but same orthographic form

### *Speech Recognition*

to, two, too

*Perfect homonyms are also problematic*

# 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.**

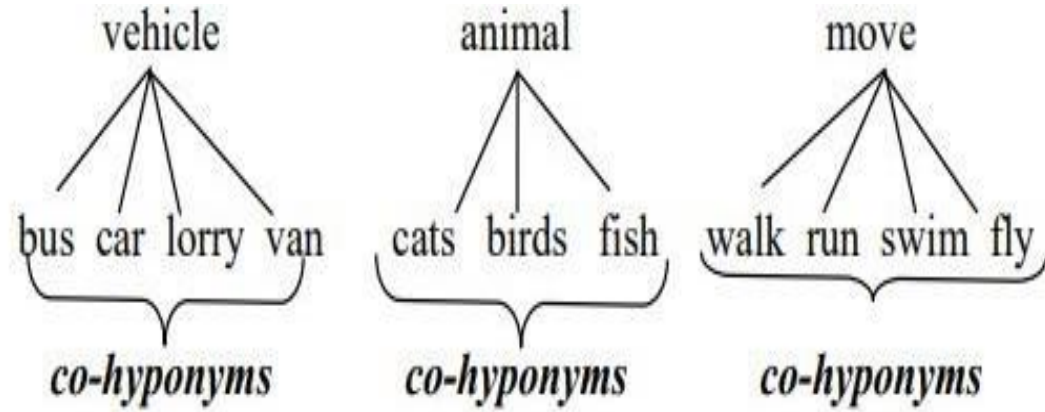
# Hyponymy

- 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*



## Hyponymy and Hypernymy

### Hyponymy

One sense is a hyponym of another if the first sense is more specific, denoting a subclass of the other

- *car* is a hyponym of *vehicle*
- *dog* is a hyponym of *animal*
- *mango* is a hyponym of *fruit*

### Hypernymy

Conversely

- *vehicle* is a hypernym/superordinate of *car*
- *animal* is a hypernym of *dog*
- *fruit* is a hypernym of *mango*

## *Hyponymy more formally*

### *Entailment*

Sense *A* is a hyponym of sense *B* if being an *A* entails being a *B*.

Ex: dog, animal

### *Transitivity*

*A* hypo *B* and *B* hypo *C* entails *A* hypo *C*

# 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



# 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 Analysis

- Semantic analysis is the process of **finding the meaning** from text.
- This analysis gives the power to computers to understand and interpret sentences, paragraphs, or whole documents, by analyzing their grammatical structure
- Identifying the relationships between individual words of the sentence in a particular context.
- The goal of semantic analysis is **to draw exact meaning or dictionary meaning** from the text.
- The work of a semantic analyzer is to check the text for meaningfulness.

# Process of semantic analysis

- Word sense disambiguation
- Lexical Disambiguation
- Structural Disambiguation

# Introductions

- **Word sense disambiguation (WSD)** in [Natural Language Processing \(NLP\)](#) is the problem of identifying which “sense” (meaning) of a word is activated by the use of the word in a particular context or scenario.
- In people, this appears to be a largely unconscious process.
- The challenge of correctly identifying words in NLP systems is common, and determining the specific usage of a word in a sentence has many applications.
- **Application:**
  - Information Retrieval,
  - Question Answering systems,
  - [Chat-bots](#), etc.

# Introductions

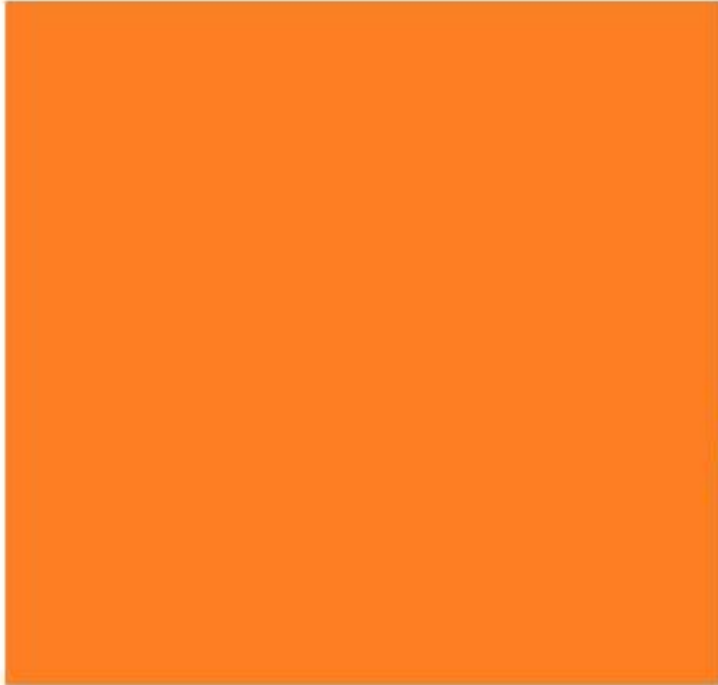
- **Word Sense Disambiguation (WSD)** is a subtask of Natural Language Processing that deals with the problem of identifying the correct sense of a word in context.
- Many words in natural language have multiple meanings
- WSD aims to disambiguate the correct sense of a word in a particular context.
- For example, the word “**bank**” can have different meanings in the sentences
  - “I deposited money in the **bank**” and
  - “The boat went down the river **bank**”.

# Word sense disambiguation

- Word sense disambiguation is the automated process of identifying in which sense is a word used according to its context.
- As natural language consists of words with several meanings (polysemic),
- **The objective here is to recognize the correct meaning based on its use.**
- For example, 'Raspberry Pi' ~~can~~ refer to a fruit, a single-board computer, or even a company (UK-based foundation).
- Hence, it is **critical to identify which meaning suits the word depending on its usage.**

# Word sense disambiguation

The word "*orange*," for example, can refer to a color, a fruit, or even a city in Florida!





# Approaches to WSD

- **Supervised learning:** This involves training a machine learning model on a dataset of annotated examples,
  - Where each example contains a target word and its sense in a particular context.
  - The model then learns to predict the correct sense of the target word in new contexts.
- **Unsupervised learning:** This involves clustering words that appear in similar contexts together,
  - Then assigning senses to the resulting clusters.
  - This approach does not require annotated data, but it is less accurate than supervised learning.
- **Knowledge-based:** This involves using knowledge base, such as a dictionary or ontology, to map words to their different senses.
  - This approach relies on the availability and accuracy of the knowledge base.
- **Hybrid:** This involves combining multiple approaches, such as supervised and knowledge-based methods, to improve accuracy

# Difficulties in Word Sense Disambiguation

- **Different Text-Corpus or Dictionary:** One issue with word sense disambiguation is determining what the senses are because different dictionaries and thesauruses divide words into distinct senses.
- Some academics have proposed employing a specific lexicon and its set of senses to address this problem.
- In general, however, research findings based on broad sense distinctions have outperformed those based on limited ones.
- The majority of researchers are still working on fine-grained WSD.
- **PoS Tagging:** Part-of-speech tagging and sense tagging have been shown to be very tightly coupled in any real test, with each potentially constraining the other.
- Both disambiguating and tagging with words are involved in WSM part-of-speech tagging.
- However, algorithms designed for one do not always work well for the other, owing to the fact that a word's part of speech is mostly decided by the one to three words immediately adjacent to it, whereas a word's sense can be determined by words further away.

# Sense Inventories for **Word Sense Disambiguation(WSD)**

- **Princeton WordNet:** is a vast lexicographic database of English and other languages that is manually curated.
- For WSD, this is the de facto standard inventory.
- Its well-organized Synsets, or clusters of contextual synonyms, are nodes in a network.
- **BabelNet:** is a multilingual dictionary that covers both lexicographic and encyclopedic terminology.
- It was created by semi-automatically mapping numerous resources, including WordNet, multilingual versions of WordNet, and Wikipedia.
- **Wiktionary:** a collaborative project aimed at creating a dictionary for each language separately,
- is another inventory that has recently gained popularity.

# Word Sense Disambiguation

## *Word Sense Disambiguation (WSD)*

### *Sense ambiguity*

- Many words have several meanings or senses
- The meaning of **bass** depends on the context
- Are we talking about music, or fish?
  - An electric guitar and **bass** player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.
  - And it all started when fishermen decided the striped **bass** in Lake Mead were too skinny.

### *Disambiguation*

- The task of disambiguation is to determine which of the senses of an ambiguous word is invoked in a particular use of the word.
- This is done by looking at the context of the word's use.

- Knowledge Based Approaches
  - Overlap Based Approaches
- Machine Learning Based Approaches
  - Supervised Approaches
  - Semi-supervised Algorithms
  - Unsupervised Algorithms
- Hybrid Approaches

# *Knowledge Based Approaches*

## *Overlap Based Approaches*

- Require a **Machine Readable Dictionary** (MRD).
- Find the overlap between the features of different senses of an ambiguous word (**sense bag**) and the features of the words in its context (**context bag**).
- The features could be sense definitions, example sentences, hypernyms etc.
- The features could also be given weights.
- The sense which has the maximum overlap is selected as the contextually appropriate sense.



# Walker's Algorithms

## Walker's Algorithm

- A Thesaurus Based approach
- **Step 1:** For each sense of the target word find the thesaurus category to which that sense belongs
- **Step 2:** Calculate the score for each sense by using the context words. *A context word will add 1 to the score of the sense if the thesaurus category of the word matches that of the sense.*
  - E.g. The money in this bank fetches an interest of 8% per annum
  - Target word: *bank*
  - Clue words from the context: *money, interest, annum, fetch*

	Sense: Finance	Sense: Location
Money	+1	0
Interest	+1	0
Fetch	0	0
Annum	+1	0
Total	3	0

Context words add 1 to the sense when the topic of the word matches that of the sense

---

## The church bells no longer rung on Sundays.

---

### church

1. one of the groups of Christians who have their own beliefs and forms of worship
2. a place for public (especially Christian) worship
3. a service conducted in a church

### bell

1. a hollow device made of metal that makes a ringing sound when struck
2. a push button at an outer door that gives a ringing or buzzing signal when pushed
3. the sound of a bell

### ring

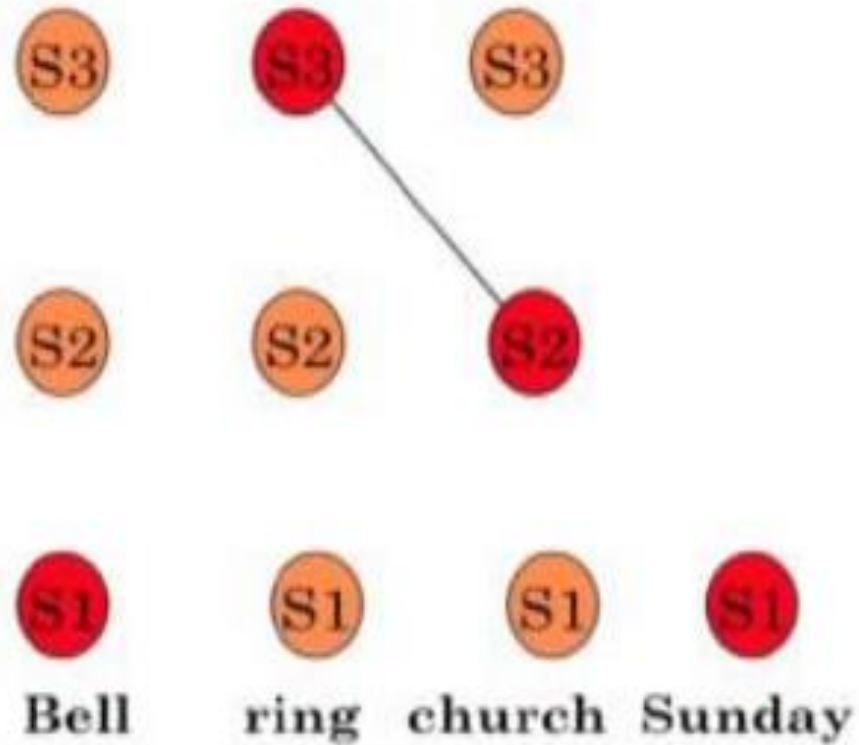
1. make a ringing sound
2. ring or echo with sound
3. make (bells) ring, often for the purposes of musical edification

### Sunday

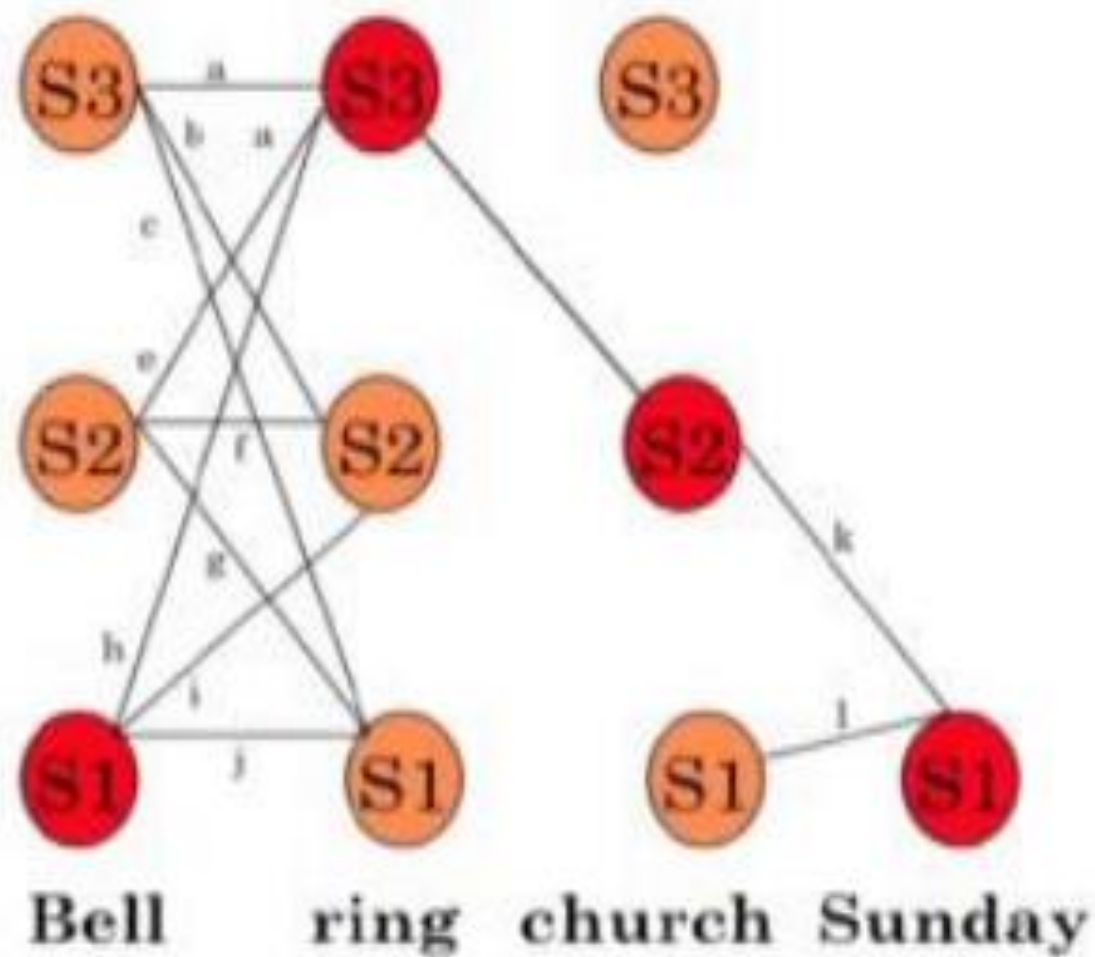
1. first day of the week; observed as a day of rest and worship by most Christians
-



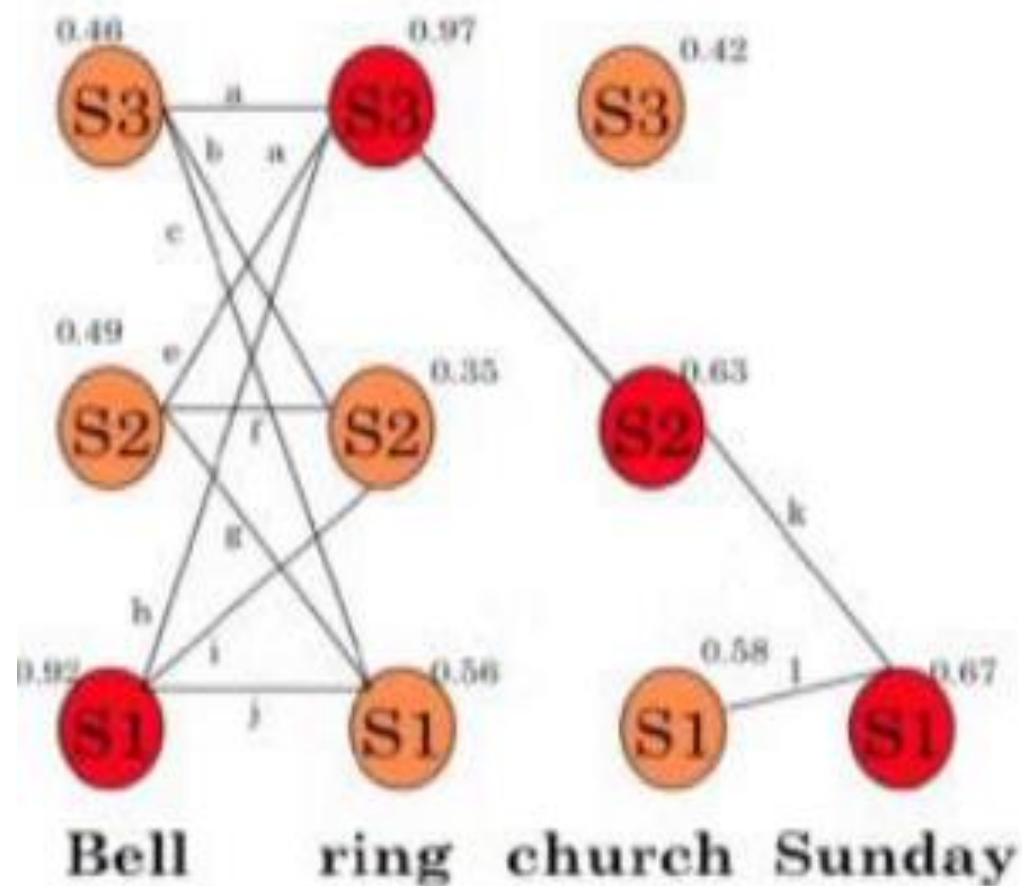
**Step 1:** Add a vertex for each possible sense of each word in the text.



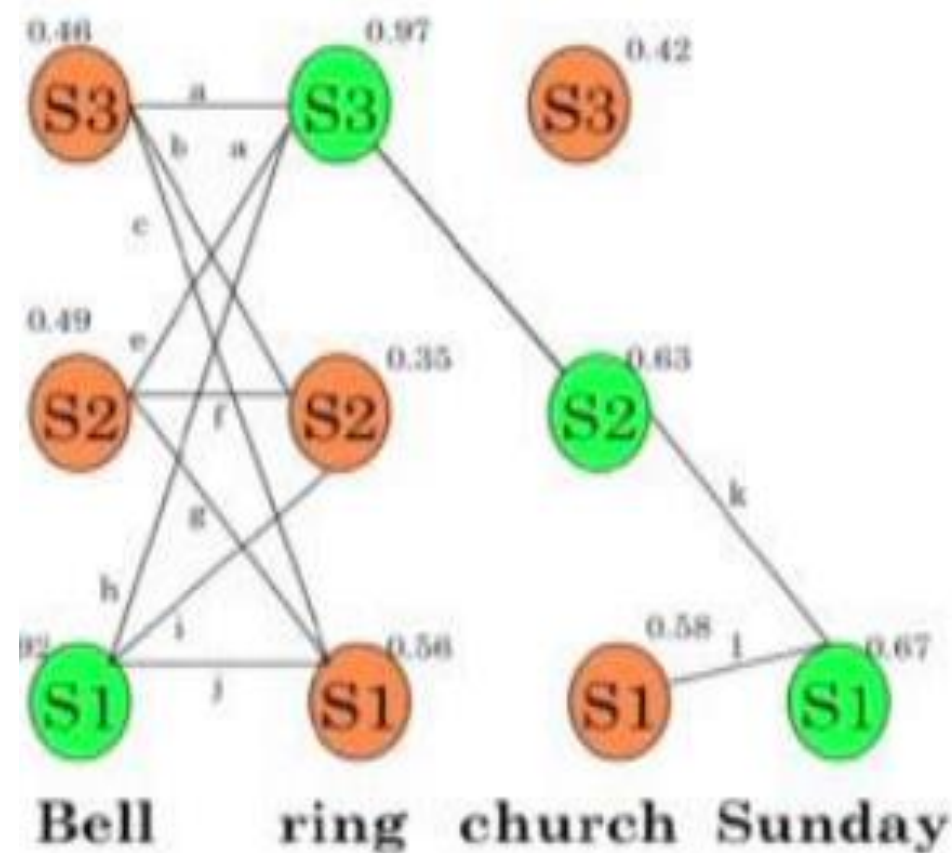
**Step 2:** Add weighted edges using definition based semantic similarity (Lesk's method).



**Step 3:** Apply graph based ranking algorithm to find score of each vertex (i.e. for each word sense).



**Step 4:** Select the vertex (sense) which has the highest score.



## Naïve Bayes for WSD

- A Naïve Bayes classifier chooses the most likely sense for a word given the features of the context:

$$\hat{s} = \arg \max_{s \in S} P(s|f)$$

- Using Bayes' law, this can be expressed as:

$$\begin{aligned}\hat{s} &= \arg \max_{s \in S} \frac{P(s)P(f|s)}{P(f)} \\ &= \arg \max_{s \in S} P(s)P(f|s)\end{aligned}$$

- The 'Naïve' assumption: all the features are conditionally independent, given the sense':

$$\hat{s} = \arg \max_{s \in S} P(s) \prod_{j=1}^n P(f_j|s)$$

## *Training for Naïve Bayes*

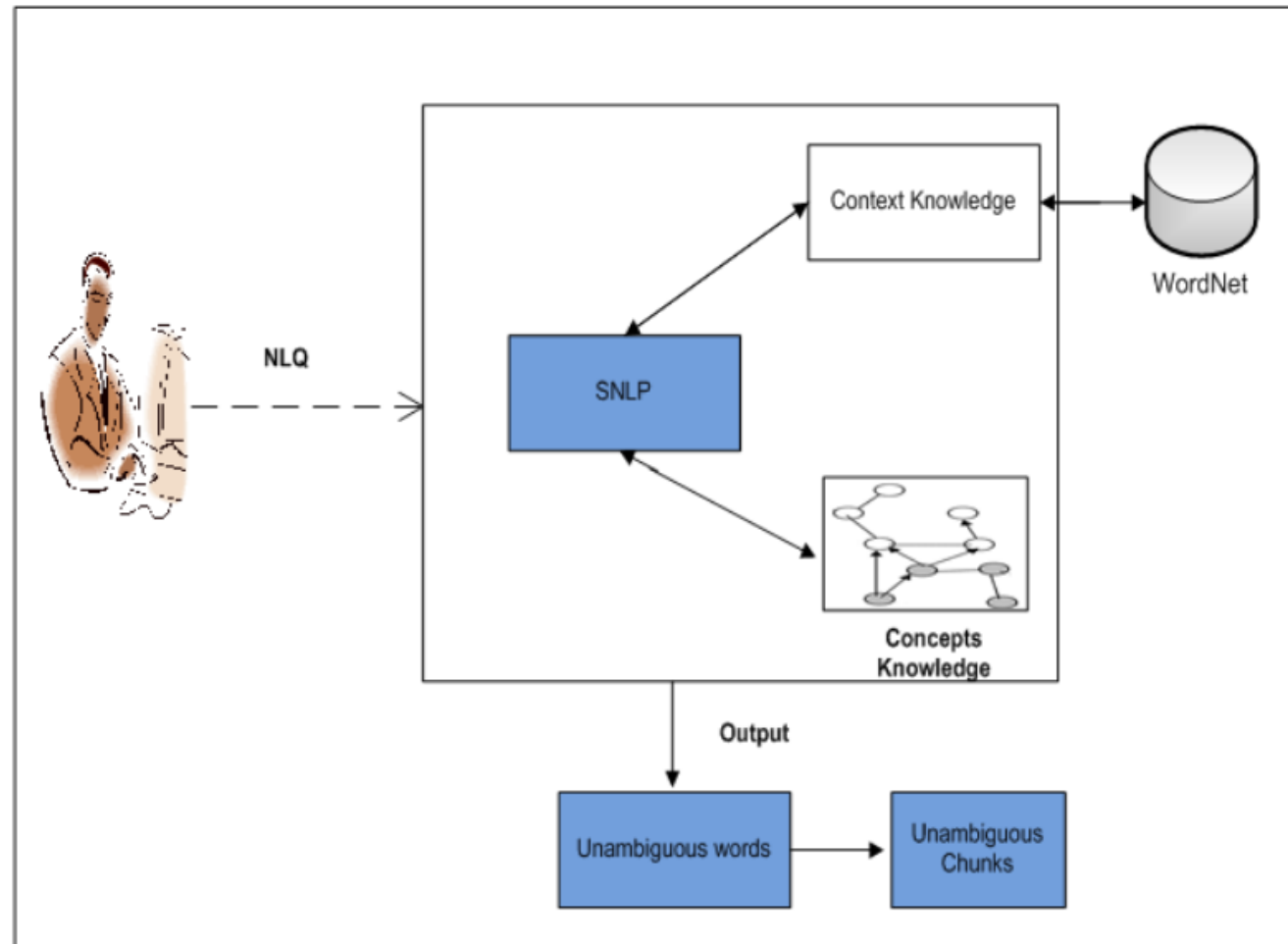
- $f$  is a feature vector consisting of:
  - POS of  $w$
  - Semantic and Syntactic features of  $w$
  - Collocation vector (set of words around it)  $\rightarrow$  next word (+1), +2, -1, -2 and their POS's
  - Co-occurrence vector
- Set parameters of Naïve Bayes using maximum likelihood estimation (MLE) from training data

$$P(s_i) = \frac{\text{count}(s_i, w_j)}{\text{count}(w_j)}$$

# Lexical Disambiguation

- The proposed approach solves lexical ambiguity in QA by considering two pieces of knowledge: context knowledge, and concepts knowledge.
- The combination of these knowledge is used to decide the most possible meaning of the word.

# Lexical Disambiguation Framework



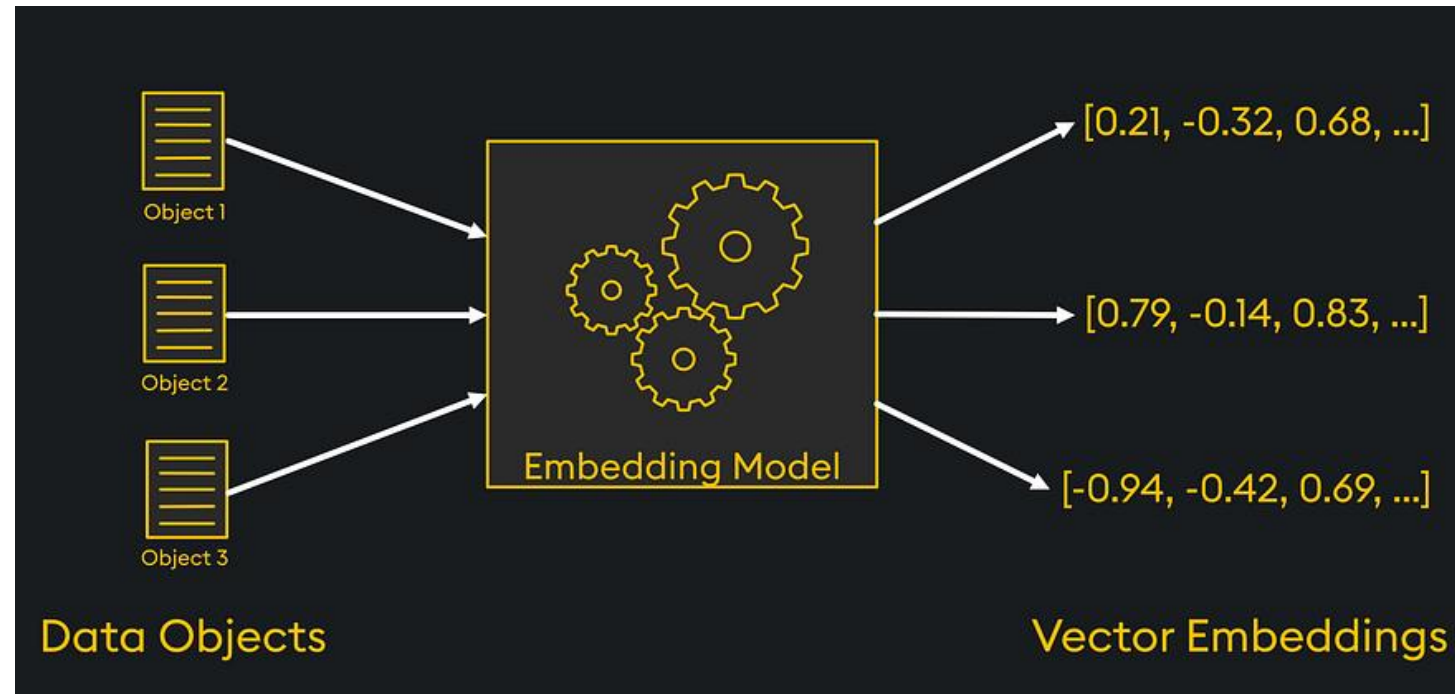


# Word Embeddings

- It is an approach for representing words and documents.
- **Word Embedding or Word Vector** is a numeric vector input that represents a word in a lower-dimensional space.
- It allows words with similar meanings to have a similar representation.
- They can also approximate meaning.
- **Word Embeddings** are numeric representations of words in a lower-dimensional space, capturing semantic and syntactic information.
- **A word vector with 50 values can represent 50 unique features.**
- **Features:** Anything that relates words to one another.
- E.g.: **Age, Sports, Fitness, Employed**, etc.
- **Each word vector has values corresponding to these features.**

# Introduction

- **Embedding** refers to mapping high-dimensional data (
- e.g., text, images) into dense, lower-dimensional vectors that preserve semantic relationships.



# Word Embeddings

## Goal of Word Embeddings

- To reduce dimensionality
- To use a word to predict the words around it
- Interword semantics must be captured

## How are Word Embeddings used?


- They are used as input to machine learning models


Take the words —> Give their numeric representation —> Use in training or inference


- To represent or visualize any underlying patterns of usage in the corpus that was used to train them.

Happy			
Sad			
Excited			
Sick			

The emoji vectors for the emojis will be:  
[happy, sad, excited, sick]

 = [1,0,1,0]

 = [0,1,0,1]

 = [0,1,0,1]

....

Emoji



# Approaches for Text Representation

## 1.1. One-Hot Encoding

One-hot encoding is a simple method for representing words in natural language processing (NLP).

In this encoding scheme, each word in the vocabulary is represented as a unique vector,

where the dimensionality of the vector is equal to the size of the vocabulary.

The vector has all elements set to 0, except for the element corresponding to the index of the word in the vocabulary, which is set to 1.

# Bag of Word (Bow)

- Bag-of-Words (BoW) is a text representation technique that represents a document as an unordered set of words and their respective frequencies.
- It discards the word order and captures the frequency of each word in the document, creating a vector representation.

# Word Embeddings

## Bag of words (BOW)

A bag of words is one of the popular word embedding techniques of text where each value in the vector would represent the count of words in a document/sentence.

In other words, it extracts features from the text. We also refer to it as vectorization.

To get you started, here's how you can proceed to create BOW.

In the first step, you have to tokenize the text into sentences.

Next, the sentences tokenized in the first step have further tokenized words.

Eliminate any stop words or punctuation.

Then, convert all the words to lowercase.

Finally, move to create a frequency distribution chart of the words.

# Limitations

- BoW ignores the order of words in the document, leading to a loss of sequential information and context making it less effective for tasks where word order is crucial, such as in natural language understanding.
- BoW representations are often sparse, with many elements being zero resulting in increased memory requirements and computational inefficiency, especially when dealing with large datasets.

# Term frequency-inverse document frequency (TF-IDF)

- Term Frequency-Inverse Document Frequency, commonly known as TF-IDF, is a numerical statistic that **reflects the importance of a word in a document** relative to a collection of documents (corpus).
- It is widely used in natural language processing and information retrieval to evaluate the significance of a term within a specific document in a larger corpus.

TF-IDF consists of two components:

- **Term Frequency (TF):** Term Frequency measures **how often a term (word) appears in a document**. It is calculated using the formula:
  - $TF(t,d) = \text{Total number of times term } t \text{ appears in document } d / \text{Total number of terms in document } d$
  - $TF(t,d) = \text{Total number of terms in document } d / \text{Total number of times term } t \text{ appears in document } d$
- **Inverse Document Frequency (IDF):** Inverse Document Frequency measures the importance of a term across a collection of documents. It is calculated using the formula:
  - $IDF(t,D) = \log(\text{Total documents} / \text{Number of documents containing term } t)$   
 $IDF(t,D) = \log(\text{Number of documents containing term } t / \text{Total documents})$
- The TF-IDF score for a term  $t$  in a document  $d$  is then given by multiplying the TF and IDF values:
  - $TF-IDF(t,d,D) = TF(t,d) \times IDF(t,D)$



# Word Embeddings

Word2Vec:

In Word2Vec **every word is assigned a vector**.

We **start with either a random vector or one-hot vector**.

**One-Hot vector:** A representation where only one bit in a vector is 1. If there are 500 words in the corpus then the vector length will be 500.

After assigning vectors to each word **we take a window size and iterate through the entire corpus**.

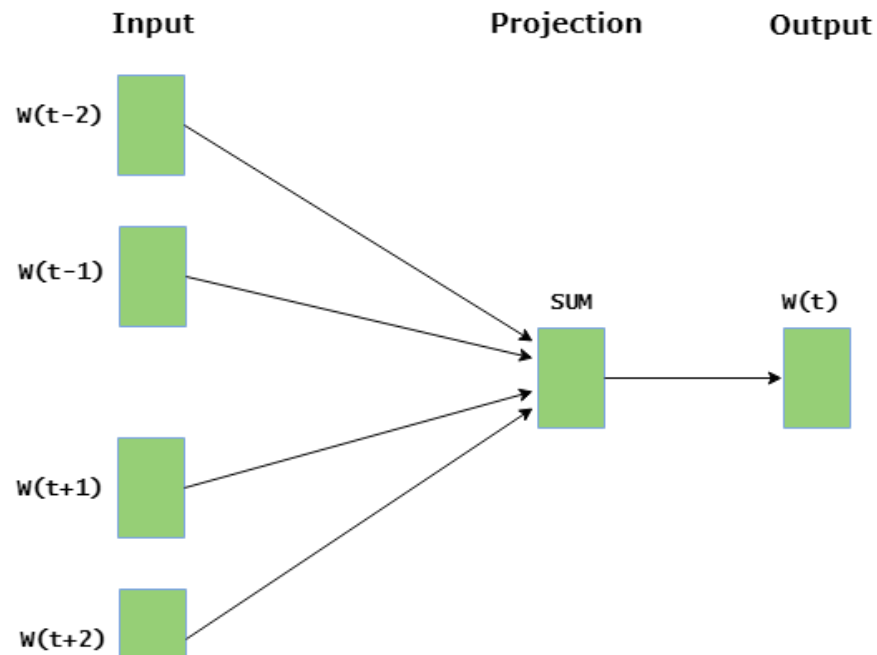
While we do this there **are two neural embedding methods** which are used:

# What is Word2Vec?

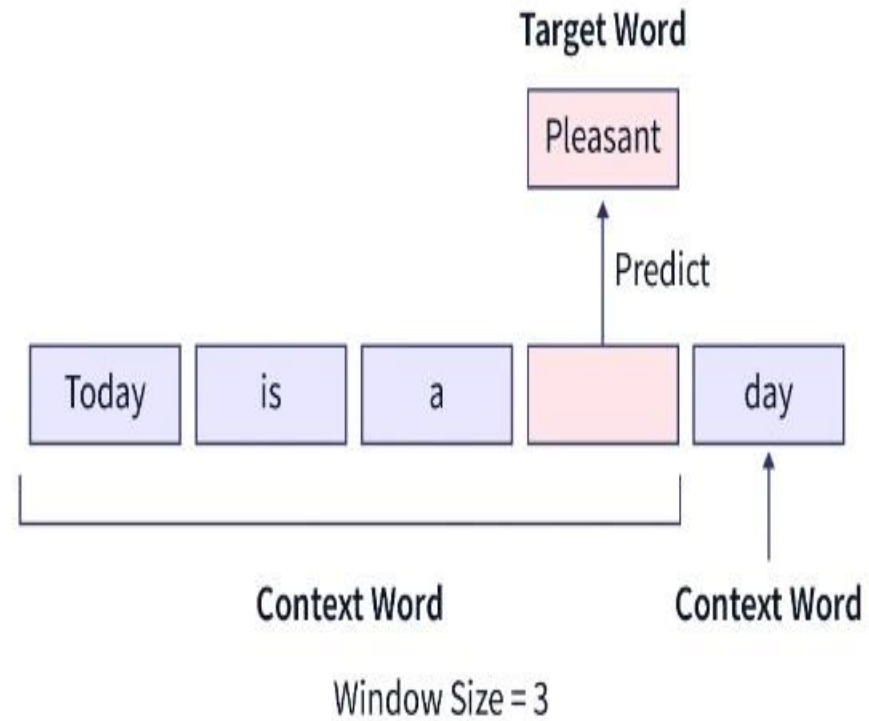
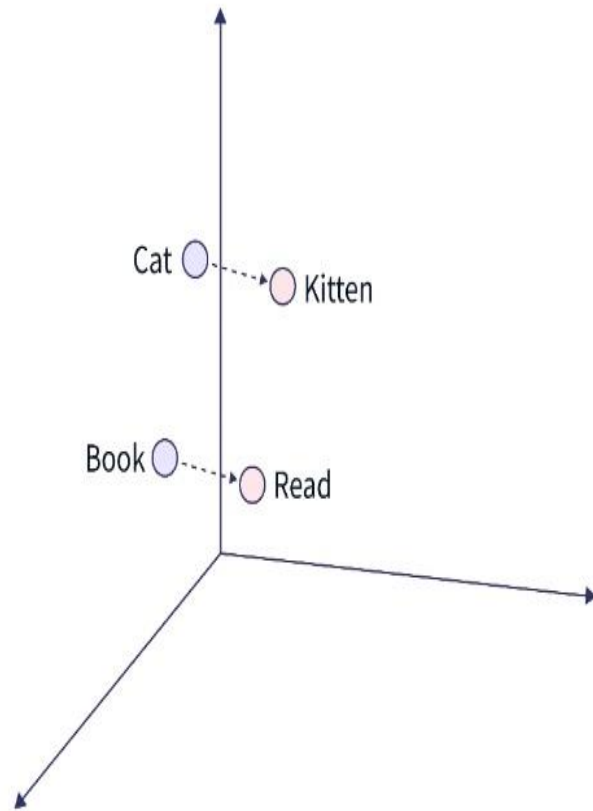
- **Word2Vec** is a widely used method in natural language processing (NLP) that allows words to be represented as vectors in a continuous vector space.
- Word2Vec is an effort to map words to high-dimensional vectors to capture the semantic relationships between words, developed by researchers at Google.
- Words with similar meanings should have similar vector representations, according to the main principle of Word2Vec.

# Continuous Bag of words (BOW)

- The continuous bag-of-words (CBOW) model is a neural network for natural language processing tasks such as **language translation and text classification**.
- It is **based on predicting a target word given the context of the surrounding words**.
- The CBOW model **takes a window of surrounding words as input and tries to predict the target word in the center of the window**.
- The model is trained on a large text dataset and learns to predict the target word based on the patterns it observes in the input data.
- The CBOW model is often combined with other natural language processing techniques and models, such as the skip-gram model, to improve the performance of natural language processing tasks.



# Continuous Bag of words (BOW)



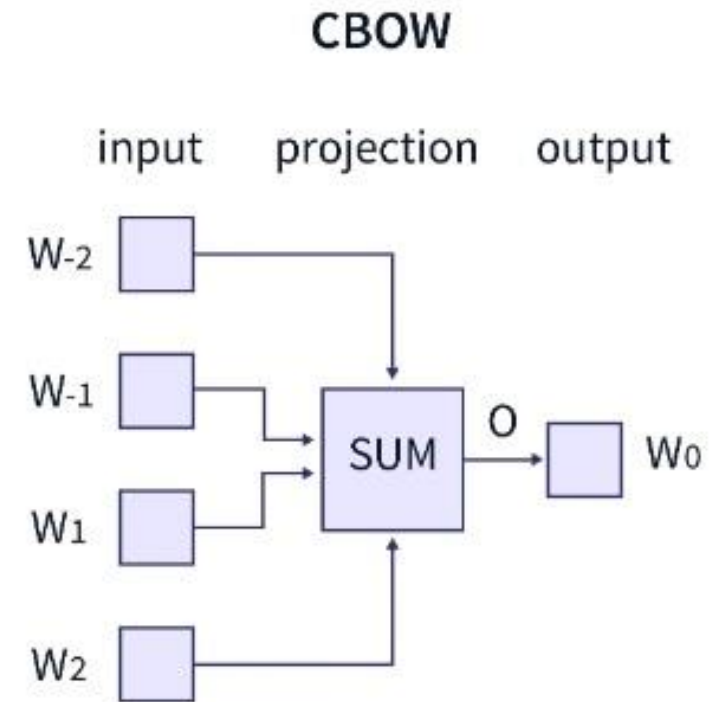
## CBOW - Architecture

The CBOW model attempts to comprehend the context of the words around the target word to predict it.

Consider the previous phrase, "It is a pleasant day." The model transforms this sentence into word pairs (context word and target word).

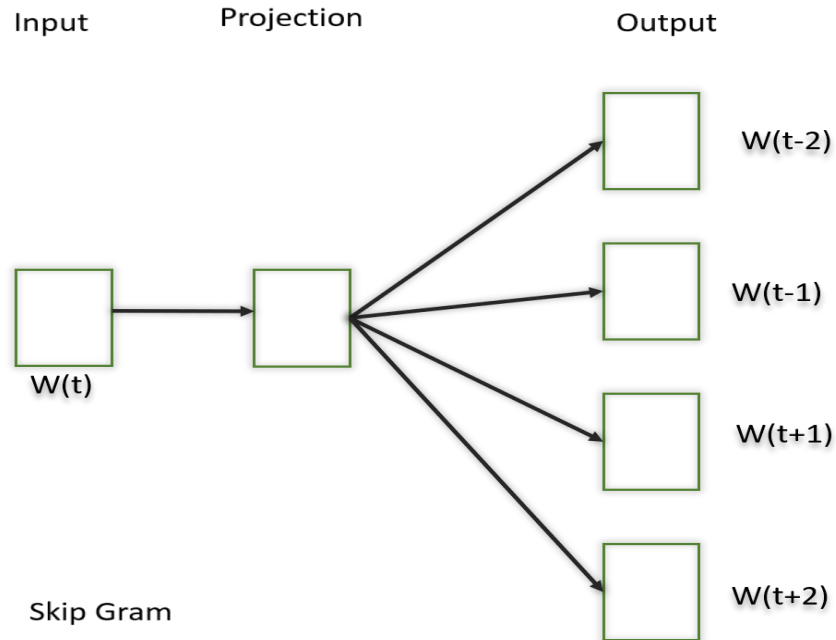
The user must configure the window size. The word pairings would appear like this if the context word's window **were 2: ([it, a], is), ([is, nice], a) ([a, day], pleasant).**

The model tries to predict the target term using these word pairings while considering the context words.



# Skip Gram

In this model, we try to make the central word closer to the neighboring words. It is the complete opposite of the CBOW model. It is shown that this method produces more meaningful embeddings.



After applying the above neural embedding methods we get trained vectors of each word after many iterations through the corpus. These trained vectors preserve syntactical or semantic information and are converted to lower dimensions. The vectors with similar meaning or semantic information are placed close to each other in space.

# Why we need Word2Vec?

- **Semantic Representations:** Word2Vec records the connections between words semantically. Words are represented in the vector space so that similar words are near to one another. This enables the model to interpret words according to their context within a particular corpus.
- **Distributional Semantics:** The foundation of Word2Vec is the distributional hypothesis, which holds that words with similar meanings are more likely to occur in similar contexts. Word2Vec generates vector representations that reflect semantic similarities by learning from the distributional patterns of words in a large corpus.
- **Vector Arithmetic:** Word2Vec generates vector representations that have intriguing algebraic characteristics. Vector arithmetic, for instance, can be used to record word relationships. One well-known example is that the vector representation of “queen” could resemble the vector representation of “king” less “man” plus “woman.”
- **Efficiency:** Word2Vec’s high computational efficiency makes training on big datasets possible. Learning high-dimensional vector representations for a large vocabulary requires this efficiency.
- **Transfer Learning:** A variety of natural language processing tasks can be initiated with pre-trained Word2Vec models. Time and resources can be saved by [fine-tuning](#) the embeddings discovered on a sizable dataset for particular uses.
- **Applications:** Word2Vec embeddings have shown promise in a number of natural language processing (NLP) applications, such as [machine translation](#), text classification, [sentiment analysis](#), and information retrieval. These applications are successful in part because of their capacity to capture semantic relationships.
- **Scalability:** Word2Vec can handle big corpora with ease and is scalable. Scalability like this is essential for training on large text datasets.

# Applications of Word Embedding

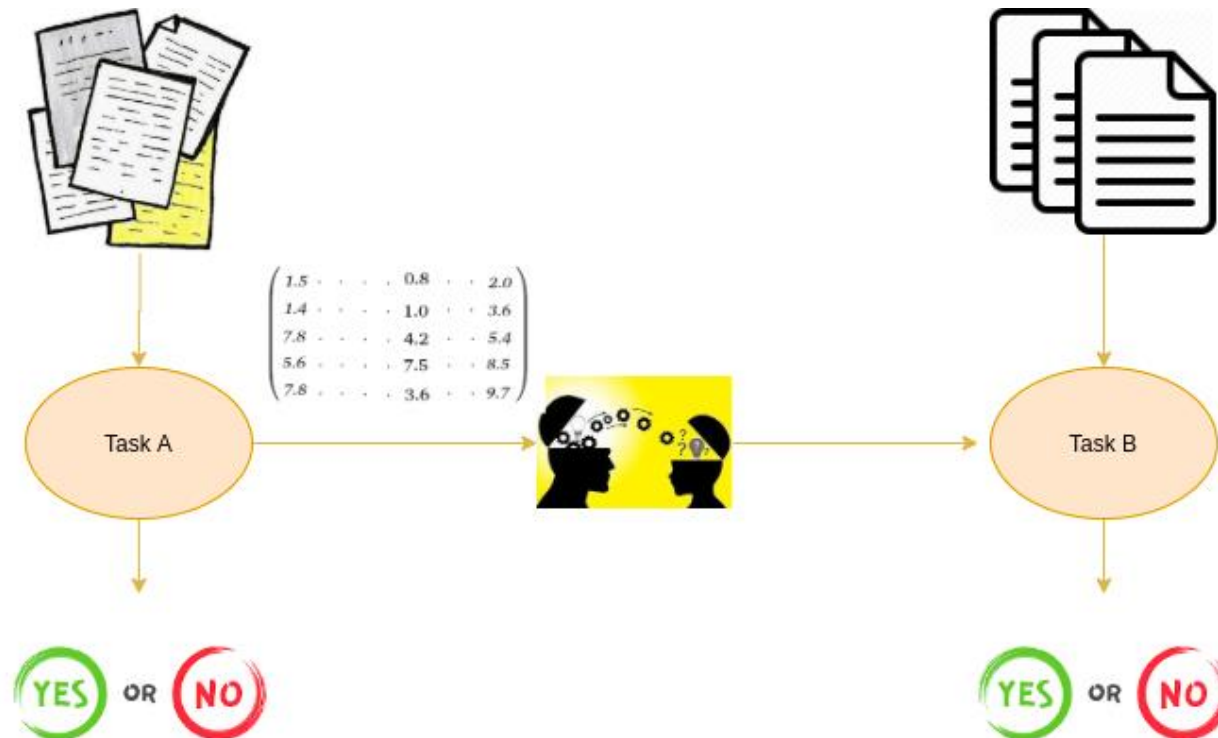
- **Text classification:** Using word embeddings to increase the precision of tasks such as topic categorization and sentiment analysis.
- **Named Entity Recognition (NER):** Using word embeddings semantic context to improve the identification of entities (such as names and locations).
- **Information Retrieval:** To provide more precise search results, embeddings are used to index and retrieve documents based on semantic similarity.
- **Machine Translation:** The process of comprehending and translating the semantic relationships between words in various languages by using word embeddings.
- **Question Answering:** Increasing response accuracy and understanding of semantic context in Q&A systems.



# **Sentence Embeddings**

# What are Pretrained Word Embeddings?

- Pretrained Word Embeddings are the embeddings learned in one task that are used for solving another similar task. These embeddings are trained on large datasets, saved, and then used for solving other tasks. That's why pretrained word embeddings are a form of **Transfer Learning**.



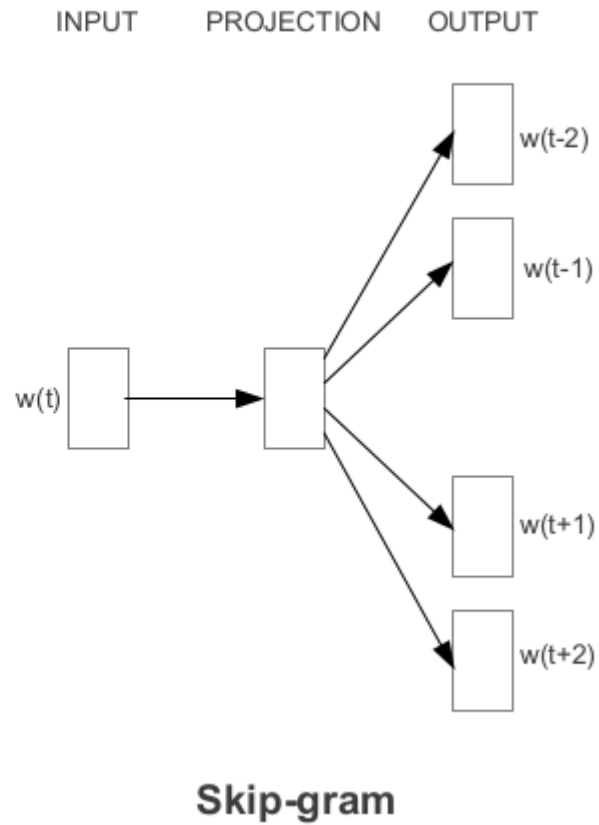
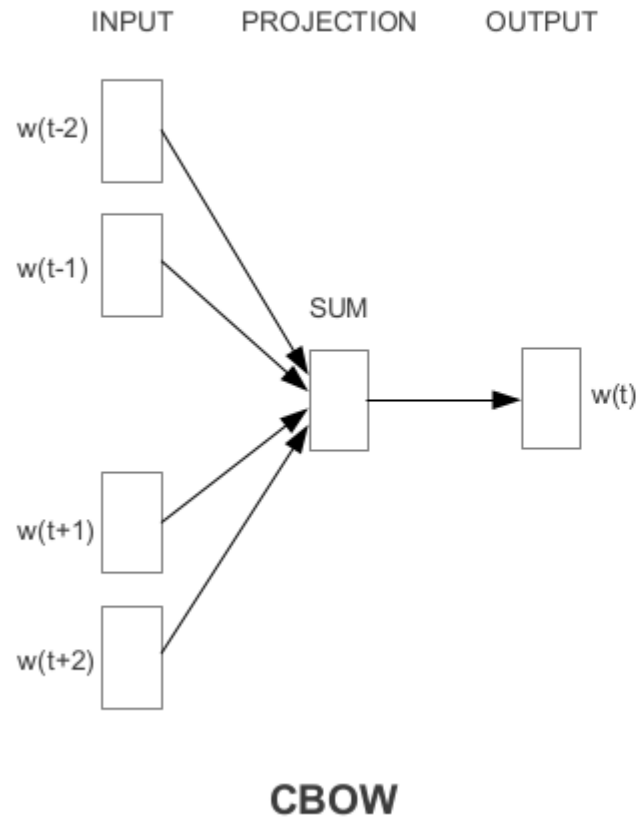
# Google's Word2vec Pretrained Word Embedding

- Word2Vec is one of the most popular pretrained word embeddings developed by Google.
- Word2Vec is trained on the Google News dataset (about 100 billion words).
- It has several use cases such as [Recommendation Engines](#), Knowledge Discovery, and also applied in the different [Text Classification](#) problems.
- Depending on the way the embeddings are learned, Word2Vec is classified into two approaches:
- **Continuous Bag-of-Words (CBOW)**
- **Skip-gram model**

# Example

- Exp: “I have failed at times but I never stopped trying”.
- Let’s say we want to learn the embedding of the word “failed”. So, here the focus word is “failed”.
- **The first step is to define a context window.** A context window refers to the number of words appearing on the left and right of a focus word.
- The words appearing in the context window are known as neighboring words (or context).
- Let’s fix the context window to 2 and then input and output pairs for both approaches:
- **Continuous Bag-of-Words:** Input = [ I, have, at, times ], Output = failed
- **Skip-gram:** Input = failed, Output = [I, have, at, times ]

# CBOW and Skip Gram



# Stanford's GloVe Pretrained Word Embedding

- The basic idea behind the GloVe word embedding is to derive the relationship between the words from Global Statistics
- **A co-occurrence matrix tells us how often a particular pair of words occur together.**
- **Each value in a co-occurrence matrix is a count of a pair of words occurring together.**

# Stanford's GloVe Pretrained Word Embedding

- For example, consider a corpus: “I play cricket, I love cricket and I love football”. The co-occurrence matrix for the corpus looks like this:

	play	love	football	I	cricket
play	0.0	0.0	0.0	1.0	1.0
love	0.0	0.0	1.0	2.0	1.0
football	0.0	1.0	0.0	0.0	0.0
I	1.0	2.0	0.0	0.0	0.0
cricket	1.0	1.0	0.0	0.0	0.0

$$p(\text{cricket}/\text{play})=1$$

$$p(\text{cricket}/\text{love})=0.5$$

Next, let's compute the ratio of probabilities:

$$\frac{p(\text{cricket}/\text{play})}{p(\text{cricket}/\text{love})} = 2$$

As the ratio  $> 1$ , we can infer that the most relevant word to cricket is “play” as compared to “love”.

Similarly, if the ratio is close to 1, then both words are relevant to cricket.

# What is Sentence Embedding?

- In NLP, sentence embedding refers to a **numeric representation of a sentence in the form of a vector of real numbers, which encodes meaningful semantic information.**
- It enables comparisons of sentence similarity by measuring the distance or similarity between these vectors.
- Techniques like **Universal Sentence Encoder (USE)** use deep learning models trained on large corpora to generate these embeddings
- Applications in tasks like **text classification, clustering, and similarity matching** this embedding is used



# What is Sentence Embedding?

- Exp: 1) 'I don't like crowded places',
- 2) 'However, I like one of the world's busiest cities, New York'.
- How can we make the machine draw the inference between 'crowded places' and 'busy cities'?
- Sentence embedding techniques represent entire sentences and their semantic information as vectors.
- This helps the machine in understanding the context, intention, and other nuances in the entire text.

# Sentence Embedding Models

- Sentence embedding models are designed to encapsulate the semantic essence of a sentence within a fixed-length vector.
- Unlike traditional Bag-of-Words (BoW) representations or one-hot encoding, sentence embeddings capture context, meaning, and relationships between words.
- This transformation is crucial for enabling machines to grasp the subtleties of human language.

# Methods of Sentence Embedding

Several methods are employed to generate sentence embeddings:

- 1.Averaging Word Embeddings:** This approach involves taking the average of word embeddings within a sentence. While simple, it may not capture complex contextual nuances.
- 2.Pre-trained Models like BERT:** Models like BERT (Bidirectional Encoder Representations from Transformers) have revolutionized sentence embeddings. BERT-based models consider the context of each word in a sentence, resulting in rich and contextually aware embeddings.
- 3.Neural Network-Based Approaches:** Skip-Thought vectors and InferSent are examples of neural network-based sentence embedding models. They are trained to predict the surrounding sentences, which encourages them to understand sentence semantics.

# References

- [Embedding Models in NLP: A Comprehensive Guide to Word Embeddings and Contextualized Embeddings | Medium](#)

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.

- 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 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.**



# Example

- 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.

# 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.

# 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.