MODULE I

MODULE II

Module III

MODULE IV

Lexical Semantics, Attachment for fragment of English- sentences, noun phrases, Verb phrases, prepositional phrases, Relations among lexemes & their senses –Homonymy, Polysemy, Synonymy, Hyponymy, WordNet, Robust Word Sense Disambiguation (WSD), Dictionary based approach

MODULE V

Discourse –reference resolution, reference phenomenon , syntactic & semantic constraints on co reference

UNIT VI

Machine translation, Information retrieval, Question answers system, categorization, summarization, sentiment analysis, Named Entity Recognition.

□□□

Module - 1

1 Introduction

Syllabus

History of NLP, Generic NLP system, levels of NLP , Knowledge in language processing , Ambiguity in Natural language , stages in NLP, challenges of NLP ,Applications of NLP

TOPICS

1.1 What is Natural Language Processing ?

- Humans are using language as a primary means of communication and by using this tool they are expressing their emotions and ideas.
- Language is used to shape the thoughts; it has structure and also it carries a meaning. By using our language, we naturally learn the new concepts and hardly realise how we process this natural language.
- **Natural Language Processing** is the process of computer analysis of input provided in a human language, and conversion of this input into a useful form of representation.
- Natural language processing is concerned with the development of computational models of aspects of human language processing. The following are the two main reasons for such developments
  - Develop automatic tool for natural language processing
  - Gain better understanding of human communication
- When we build computational models by using human language, we need processing abilities where this processing abilities and incorporates how human collects, Store and process the language. It also needed a knowledge of the world and of language.
- The input and output of the NLP is text or speech.

1.2 History of NLP

Machine Translation (1940-1960)

In year 1940 Natural language processing has started. In 1948,in Birkbeck College, London, the first NLP application was developed. Later, in 1950, there were contradictory opinions between linguistics and computer science.Chomsky came up with his first book called syntactic structures and claimed that language is reproductive in nature.Chomsky then in 1957 introduced idea of Generative Grammar that was a rule-based description of syntactic structures.

Flavoured with Artificial Intelligence (1960-1980)

- The year 1960 to 1980 witnessed the developments like Augmented Transition Networks which was a finite state machine that is capable of recognizing regular languages. Then in 1968, Linguist Charles J. FillmoredevelopedCase Grammar and this case grammar utilizes English language to express the association between nouns and verbs by using the preposition. Here, Case role is to link certain types of verbs and objects. For example, "Meena broke the table with the stone". In this example case grammar identify Meena as an agent, table as a theme, and stone as an instrument. SHRDLU and LUNAR were the key systems in the year 1960 to 1980.

- Terry Winograd in 1968-70 wrote a program named SHRDLU. This program helps users to communicate with the computer and moving objects. It can handle orders such as *pick up the green ball* and likewise answer the questions like *What is inside the black box*. The SHRDLU's key importance is it shows those syntax, semantics, and reasoning about the world that can be combined to produce a system that understands a natural language.
- Another system is LUNAR. It is the typical example of a Natural Language database interface system. LUNAR used ATNs and Woods' Procedural Semantics. It was proficient of translating extravagant natural language expressions into database queries and handle 78% of requests without mistakes.

1980 - Current

NLP was based on complex sets of hand-written rules till the year 1980. machine learning algorithms were introduced after 1980 for language processing. NLP started growing faster in the beginning of the year 1990s and accomplished good process accuracy, especially in English Grammar. Electronic text was introduced in 1990 thatgiven a decent resource for training and examining natural language programs. Other factors may include the availability of computers with fast CPUs and more memory. The keyfeature behind the progress of natural language processing was the Internet. In the year1990 Probabilistic and data-driven models had become quite standard. After that, in the year 2000, ahuge amount of spoken and textual data become available.

1.3 Generic NLP System

The Generic NLP systems are ELIZA, SysTran, TAUM METEO, SHRDLU, and LUNAR.

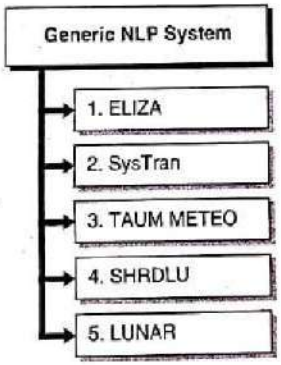


Fig. 1.3.1 : Generic NLP system

1. **ELIZA :** ELIZA an early natural language understanding program created by Joseph Weizenbaum.The human conversation with the user is mimicked by using syntactic patterns. This system demonstrates communication between humans and machines.
2. **SysTran (System Translation) :** In 1969, the SysTran machine translation system was developed. This system was developed for Russian- English translation. This system provides the first online machine translation service named Babel Fish. This Babel Fish was used by Alta Vista Search engine to handle translation request from users.
3. **TAUM METEO :** TAUM METEO, is a natural language generation system. This system was used in Canada for generating weather reports. This system accepts daily weather reports in English and French.
4. **SHRDLU :** Terry Winograd in 1968-70 wrote a program named SHRDLU. This program helps users to communicate with the computer and moving objects. It can handle orders such as pick up the green ball and likewise answer the questions like What is inside the black box. The SHRDLU's key importance is it shows those syntax, semantics, and reasoning about the world that can be combined to produce a system that understands a natural language.
5. **LUNAR :** LUNAR is the typical example of a Natural Language database interface system. It was an early question answer system that answers questions related to moon rock. It was proficient of translating extravagant natural language expressions into database queries and handle 78% of requests without mistakes.

1.4 Levels of NLP

There are seven interdependent levels to understand and extract meaning from a text or spoken words. In order to understand natural languages, it's important to differentiate amongst them:

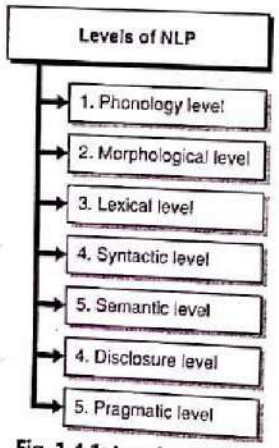


Fig. 1.4.1: Levels of NLP

1. **Phonology level**

   This level basically deals with the pronunciation. As English spelling is especially only partially phonemic, John *inputs the data* does not show these very clearly; for example, the *h* in *John* is silent and the two as in *data* resemble to very unlike sounds

2. **Morphological level**

   Morphology deals with the smallest parts of words that convey meaning, and suffixes and prefixes. Morphemes means studying how the words are built from smaller meaning. For example, the word 'dog' has single morpheme while the word 'rats' have two morphemes 'rat' and morpheme 's' denotes singular and plural concepts.

3. **Lexical level**

   The lexical level deals with the study at the level of words with respect to their lexical meaning and Part-Of-Speech (POS). This level uses lexicon that is a collection of individual lexemes. A lexeme is a basic unit of lexical meaning; which is an abstract unit of morphological analysis that represents the set of forms or *"senses"* taken by a single morpheme. For example,*"Duck"*, can take the form of a noun or a verb but its POS and lexical meaning can only be derived in context with other words used in the phrase/sentence.

4. **Syntactic level**

   Syntactic level deals with grammar and structure of sentences.It studies the proper relationships between words. The POS tagging output of the lexical analysis can be used at the syntactic level of two group words into the phrase and clause brackets. Syntactic Analysis also referred to as *"parsing"*, allows the extraction of phrases which convey more meaning than just the individual words by themselves, such as in a noun phrase.

5. **Semantics level**

   - This level deals with the meaning of words and sentences. There are two approaches of semantic level :
     1) syntax-driven semantic analysis,
     2) Semantic grammar
   - It is a study of the meaning of words that are associated with grammatical structure. For example, *John inputs the data* from this statement we can understand that John is an *Agent.*

6. **Discourse level**

- This level deals with the structure of different kinds of text. There are two types of discourse:
  1) Anaphora resolution,
  2) discourse/text structure recognition.
- The words are replaced in Anaphora resolution, for example pronouns. Discourse structure recognition determines the purpose of sentences in the text which enhances meaningful illustration of the text.

7. **Pragmatic level**

This level deals with the use of real world knowledge and understanding of how this influences the meaning of what is being communicated. By analysing the appropriate dimension of the documents and queries, a more detailed representation is derived.

1.5 Knowledge in Language Processing

- Language is used for communication and knowledge is interpreted in it. Here, we consider text as language and content as knowledge.
- Language is a medium used for expression. It is the outer form of the content it expresses, the same content can be expressed in various languages.
- The question over years can language be separated from its content? if yes then how can the content itself be represented? Usually, the meaning of one language is written in the same language but by using a different set of words. Therefore, to process a language means to process the content of it.
- Computers are not able to understand natural language, methods are developed for mapping its content in the formal language. The knowledge representation tool represents the whole body of knowledge and that has been modified maybe through generation of new words, to include new ideas and situations.
- The language processing has different levels and each level of processing contains different types of knowledge. the various levels of processing and the type of knowledge it in walls is explained below :

1. **Phonetic and Phonological Knowledge**
   - Phonetics deals sounds while phonology is the study of combination of sounds into organized units of speech, the formation of syllables and larger units.
   - Phonetic and phonological knowledge are essential for speech based systems as they deal with how words are related to the sounds that realize them.

2. **Morphological Knowledge**
   - Morphology concerns word formation.
   - It is a study of the patterns of formation of words by the combination of sounds into minimal distinctive units of meaning called mophemes.
   - **Morphological knowledge** concerns how words are constructed from morphemes.
3. **Syntactic Knowledge**
   - Syntax deals with how words combine to form phrases, phrases combine to form clauses and clauses join to make sentences.
   - Syntactic analysis concerns sentence formation.
   - It deals with how words can be put together to form correct sentences.
   - It also determines what structural role each word plays in the sentence and what phrases are subparts of what other phrases.
4. **Semantic Knowledge**
   - It concerns meanings of the words and sentences.
   - This is the study of context independent meaning that is the meaning a sentence has, no matter in which context it is used.
   - Defining the meaning of a sentence is very difficult due to the ambiguities involved.
5. **Pragmatic Knowledge**
   - Pragmatics is the extension of the meaning or semantics.
   - Pragmatics deals with the contextual aspects of meaning in particular situations.
   - It concerns how sentences are used in different situations and how use affects the interpretation of the sentence.
6. **Discourse Knowledge**
   - Discourse concerns connected sentences.
   - It is a study of chunks of language which are bigger than a single sentence.
   - Discourse language concerns inter-sentential links that is how the immediately preceding sentences affect the interpretation of the next sentence.
   - Discourse knowledge is important for interpreting pronouns and temporal aspects of the information conveyed.

7. World Knowledge

- Word knowledge is nothing but everyday knowledge that all speakers share about the world.
- It includes the general knowledge about the structure of the world and what each language user must know about the other user's beliefs and goals.
- This essential to make the language understanding much better.

1.6 Ambiguity in Natural Language

- Natural language has a very rich form and structure. It is very ambiguous. Ambiguity means not having well defined solution. Any sentence in a language with a large-enough grammar can have another interpretation.
- There are various forms of ambiguity related to natural language and they are:

1. Lexical Ambiguity
2. Syntactic Ambiguity
3. Semantic Ambiguity
4. Metonymy Ambiguity

1. Lexical Ambiguity

- When words have multiple assertion then it is known as lexical ambiguity.
- For example:

  the word *back* can be a noun or an adjective.

  Noun: back stage

  adjective: back door

2. Syntactic Ambiguity

- Syntactic ambiguity means sentences are parsed in multiple syntactical forms or A sentence can be parsed in different ways
- For example:

  I saw the girl on the beach with my binoculars.

  In this sentence, confusion in meaning is created. The phrase with my binoculars could modify the verb, saw or the noun, girl.

3. Semantic Ambiguity

- Semantic ambiguity is related to the sentence interpretation.
- For example:

  I saw the girl on the beach with my binoculars.

  The sentence means that I saw a girl through my binoculars or the girl had my binoculars with her

4. Metonymy Ambiguity

- Metonymy is the most difficult ambiguity. It deals with phrases in which the literal meaning is different from the figurative assertion.
- For example:

  "Nokia us screaming for new management",

  Here it really doesn't mean that the company is literally screaming.

1.7 Stages in NLP

There are five stages in natural language processing. The Fig. 1.7.1 shows the stages lexicalanalysis, syntactic analysis, semantic analysis, disclosureintegration, and pragmatic analysis.

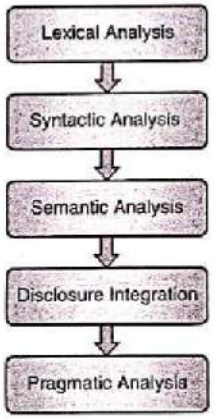


Fig. 1.7.1: Stages of NLP

1. Lexical Analysis

- Lexical Analysis is the first stage in NLP. It is also known as morphological analysis.
- At this stage the structure of the words is identified and analysed.

- Lexicon of a language means the collection of words and phrases in a language.
- Lexical analysis is dividing the whole portion of text into paragraphs, sentences, and words.

2. Syntactic Analysis (Parsing)

- It involves analysis of words in the sentence for grammar and ordering words in a way that shows the relationship among the words.
- The sentence such as *The school goes to girl* is rejected by English syntactic analyser.

3. Semantic Analysis

- Semantic analysis draws the exact meaning or the dictionary meaning from the text.
- The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain.
- The semantic analyserneglects sentence such as "*hot ice-cream*".

4. Discourse Integration

- The meaning of any sentence depends upon the meaning of the sentence just before it. Furthermore, it also brings about the meaning of immediately following sentence.
- For example: *Meena is a girl, she goes to school* here "she" is a dependency pointing to Meena.

5. Pragmatic Analysis

- During this, what was said is re-interpreted on what it truly meant. It contains deriving those aspects of language which necessitate real world knowledge.
- For example, *John saw Mary in a garden with a cat*, here we can't say that John is with cat or mary is with cat

1.8 Challenges of NLP

NLP is a powerful tool with enormous benefits; nonetheless there are still numerous Natural Language Processing challenges :

1. Contextual words and phrases and homonyms

- The same words and phrases can have diverse meanings according the context of a sentence and many words have the exact same pronunciation but completely different meanings.

- For example :

  I ran to the store because we ran out of milk.

  Can I run something past you really quick?

  The house is looking really run down.
- In the above three sentences the meaning of the *run* is different according to the context
- Homonyms means the pronunciation of two or more words is same but have different meaning. For example, their and there, right and write. This will create problem in question answering and speech-to-text applications.

2. Synonyms

- Synonyms can cause issues like contextual understanding since we use many different words to express the identical idea.
- Additionally, some of these words may convey exactly the same meaning, while some may be levels of complexity and different people use synonyms to denote slightly different meanings within their personal vocabulary.
- For example, small, little, tiny, minute have same meaning.

3. Irony and sarcasm

- Irony and sarcasm present problems for machine learning models since they usually use words and phrases that, strictly by definition, may be positive or negative, but truly mean the opposite.
- Models can be trained with certain indications that frequently accompany ironic or sarcastic phrases, like *yeah right, whatever*, etc., and word embeddings (where words that have the same meaning have a similar representation), but it's still a complicated process.

4. Ambiguity

- Ambiguity in NLP refers to sentences and phrases that potentially have two or more possible interpretations.
- There is lexical, syntactic and sematic ambiguity.

5. Errors in text or speech

- Misspelled or misused words can generate problems for text analysis. Autocorrect and grammar correction applications can handle common mistakes, but do not at all times understand the writer's intention.

- With spoken language it is difficult for the machine to understand mispronunciations, different accents, stammers, etc..

6. Idioms and slang

- Informal phrases, expressions, idioms, and culture-specific lingo present a number of problems for NLP, especially for models intended for comprehensive use.
- Because as formal language, idioms may have no dictionary definition at all, and these expressions may even have different meanings in different geographic areas.
- Furthermore, cultural slang is continuously morphing and increasing, so new words arise every day.

7. Domain specific language

- Different businesses and industries often use very different language.
- An NLP processing model needed for healthcare would be very different than one used to process legal documents.

8. Low-resource languages

- Artificial Intelligence, machine learning NLP applications have been mostly built for the most common, widely used languages.
- It is absolutely incredible at how precise translation systems have become. However, many languages, especially those spoken by people with less access to technology often go overlooked and under processed.
- For example, there are over 3,000 languages in Africa, alone. There simply isn't ample data on many of these languages.

1.9 Applications of NLP

The first application area of Natural Language Processing is machine translation. Machine translate considered complete linguistic analysis of the natural language sentences as well as linguistic generation of and output sentence. There is vast progress happened in NLP field. Some applications of NLP are as follows :

1.	Machine translation	2.	Speech recognition
3.	Speech synthesis	4.	Information retrieval
5.	Information extraction	6	Question answering
7.	Text summarization.	8.	Sentiment Analysis

1. Machine translation

- In machine translation, the translation of the text in one human language to another human language is performed automatically.
- For performing the translation, it is important to have the knowledge of the words and phrases, grammar of two languages that are involved in Translation, semantics of the languages and the Knowledge of the word.

2. Speech recognition

- Speech recognition is the process where the acoustic speech signals are mapped to the set of words.
- As there is wide variation in the pronunciation of the word, homonym for example, sea and see, acoustic ambiguities like in the rest and interest.

3. Speech synthesis

- Automatic production of speech is known as speech synthesis. It means speaking a sentence in natural language.
- The speech synthesis system reads mails on your telephone or reads storybooks for you. For generating the utterances text processing is required, so, NLP is an important component in speech synthesis system.

4. Information retrieval

- In Information retrieval the relevant documents related to the user's queries are identified. In Information retrieval indexing, query modification, word sense disambiguation, and knowledge bases are used for enhancing the performance.
- For example, wordnet, and Longman Dictionary of Contemporary English (LDOCE) Are some useful lexical resources for Information retrieval research.

5. Information extraction

- Information extraction is one of the most significant applications of NLP. It is used for extracting structured information from unstructured or semi-structured machine-readable documents. Information extraction system captures and outputs factual information contained within a document.
- Like Information retrieval system information extraction system also response to user's information need. Unlike the Information retrieval system, the information required is not expressed as a keyword query. Instead, it is stated as redefine database schemas all templates.
- In the Information retrieval system, it identifies a subset of documents in a large repository of text database for example consider a library scenario a subset of resources in a library. Information extraction system identifies a subset of information within a document that fits the predefined template.

6. Question answering

- The question answers system tries to find out the correct answer or part of the text where the answer appears for the given question and a set of documents.
- The question answers system returns a full document that is relevant to the user's query. The question answer system uses an information extraction system for identifying the entities in the text.
- A question answering system needed more NLP than an Information retrieval system or an information extraction system. It needed process analysis of question and portions of text and also semantic as well as background knowledge to answer certain type of questions.

7. Text summarization

- Text summarization means creating short, correct summary of longer text documents. Automatic text summarization will assist us with appropriate information in less time. NLP has an important role in developing an automatic text summarization.
- Text summarisation involves syntactic, semantics, and discourse level processing of text.

8. Sentiment Analysis

- Sentiment Analysis is also referred as opinion mining. It is used on the web to analyse the attitude, behaviour, and emotional state of the sender.
- This application is implemented through a combination of NLP and statistics by assigning the values to the text such as positive, negative, or natural and then recognize the mood of the context for example, happy, sad, angry, etc.

Review Questions

Q.1 What is NLP? What are the applications of NLP ?
Q.2 Explain generic NLP system.
Q.3 what are the levels of NLP ?
Q.4 Explain the stages of NLP ?
Q.5 Explain the challenges in NLP ?
Q.6 List and explain applications of NLP.
Q.7 Explain the knowledge level ?

□□□

Module - 2

2 Word Level Analysis

Syllabus

Morphology analysis – survey of English Morphology, Inflectional morphology & Derivational morphology. Lemmatization, Regular expression, finite automata, finite state transducers (FST), Morphological parsing with FST , Lexicon free FST Porter stemmer. N –Grams- N-gram language model, N-gram for spelling correction.

TOPICS

2.1 Morphology Analysis

- NLP has various levels and complexity of Processing. One technique to analyse natural language text is breakdown the text into constituent units like words phrases sentences and paragraphs and after that analyses these units.
- It is important to analyse the words before analysing the syntax because the words are the fundamental unit. This chapter is focusing on NLP performed at word level, that incorporates characterizing word sequences, recognising morphological variants, detecting the misspelled words and correcting them and recognising the precise part of speech of a word.

2.2 Survey of English Morphology

- Morphology is nothing but studying the way of words formation from minor meaning-bearing units, morphemes. There are two classes of morphemes:
  1. stem
  2. affixes
- Let take the example of a word unhappiness. The word unhappiness consists of 3 parts as shown in the Fig.2.2.1.

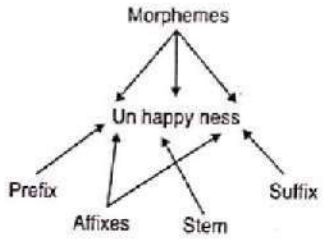


Fig. 2.2.1

- The main morpheme of the word is stem. It also states that stem are the morphemes with the central meaning
- Once the affixes are applied to the stem it modifies its meaning. There are 4 types of affixes:
  - **prefix** –These are the morphemes that come before the stem.
  - **suffix**-These are the morphemes that comes at the end of the stem
  - **infix** - These are the morphemes that come to the any end of the stem
  - **circumfix** –These are the morphemes that come inside the stem

- For Example:

Affixes	Original word	Affix applied	Word after applying affix
suffix	rat	s	rats
prefix	happy	un	unhappy
circumfix	sag	Ge and t	Ge-sag-t (in German it means said)
inflix	hingi	um	Humingi (in Philippine language)

2.3 Inflectional Morphology and Derivational Morphology

There are three types of word formation

1. Inflection
2. Derivation
3. Compounding

1. Inflection

- When the word stem is fused with the grammatical morpheme then it generally results in a word of the similar class as the original stem.
- If we consider the English language then the things that get inflected are nouns, verbs, and sometimes adjectives. So, the affixes are quite small in number.
- Nouns have simple inflectional morphology. Examples of the Inflection of noun in English are given below, here an affix is marking plural.
  - mat (-s), cat(-s)
  - ibis(-es)
  - thrush(-es)
  - waltz(-es), finch(-es), box(-es)
  - butterfly(-lies)
  - ox (oxen), mouse (mice) [irregular nouns]
- A possessive affix is a suffix or prefix attached to a noun to indicate its possessor. the following are some affixes marking possessive
  - Regular singular noun- llama's

- Plural noun not ending in 's '-children's
- Regular plural noun –llamas'
- Names ending in 's' or 'z' - Euripides' comedies

- Verbs have slightly more complex inflectional, but still relatively, simple inflectional morphology. There are three types of verbs in English
  - Main verbs- eat, sleep, impeach
  - Modal verbs- can will, should
  - Primary verbs-be, have, do
- Regular verb is a large class are regular. In this class all the verbs have the same endings marking the same functions. Regular verbs have four morphological form. Just by knowing the stem we can predict the other forms. Just add one of the three predictable endings and make some regular spelling changes. These regular verbs and forms are significant in the morphology of English because of their majority and being productive.
- The Table 2.3.1 shows the morphological forms for regular verbs.

Table 2.3.1

Stem	Talk	Urge	Cry	tap
-s form	Talks	urges	cries	Taps
-ing form	Talking	urging	Crying	Tapping
Past form or -ed participle	Talked	urged	Cried	tapped

- The morphological form for the irregular verbs is shown in the Table 2.3.2

Table : 2.3.2

Stem	Eat	Think	put
-s form	Eats	Thinks	puts
-ing form	eating	Thinking	putting
Past form	Ate	Thought	put
-ed participle	Eaten	Thought	put

The simple from: be
The -ing participle form: being
The past participle: been
The first person singular present tense from : am
The third person present tense (-s) from: is
The plural present tense from: are
The singular past tense form: was
The plural past tense from:were

2. Derivation

- The combination of a word stem with a grammatical morpheme usually resulting in a word of a different class, often with a meaning hard to predict exactly.
- Nominalizations are nothing but the nouns which aregenerated from adjectives in English. The Table 2.3.3 shows the formation of new nouns, often from verbs or adjectives.

Table 2.3.3

Suffix	Base Verb/Adjective	Derived Noun
-ation	Computerize (V)	Computerization
-ee	Appoint (V)	Appointee
-er	Kill (V)	Killer
-ness	Fuzzy (A)	Fuzziness

- Adjectives can also be derived from nouns or verbs. The Table 2.3.4 shows the objectives derived from the verbs/noun.

Table 2.3.4

Suffix	Base Verb/Adjective	Derived Noun
-al	Computation (N)	Computational
-able	embrace (V)	Embraceable
-less	clue (A)	Clueless
-ness	Fuzzy (A)	Fuzziness

3. Compounding

- A word is composed of a number of morphemes concatenated together.
- In English, compounds can be written together, for example, notebook, desktop, overlook, bookstore, fireman etc.
- Separately the compounds are written as Living room, dinner table, full moon etc.

2.4 Lemmatization

- Lemmatization is performing the task properly by using the vocabulary and morphological analysis of words, and it generally aims to eliminate inflectional endings and to return the base form of the word/ dictionary form of a word, that is called lemma.
- If we try the word saw, stemming might return just s, but lemmatization would try to return either see or saw depending on whether the use of the token was as a verb or a noun.
- In other words, Lemmatization is a technique responsible for grouping diverse inflected forms of words into the root form, having the similar meaning. It is similar to stemming, in order, it provides the stripped word that has some dictionary meaning. The Morphological analysis would need the mining of the accurate lemma of each word.
- For example: troubled' -> Lemmatization -> 'troubled', and error
- The applications of lemmatizations are information retrieval, sentiment analysis, and document clustering.

2.4.1 Difference between Stemming and Lemmatization

Table 2.4.1 : Difference between Stemming and Lemmatization

Sr. No	Stemming	Lemmatization
1	Stemming is faster because it chops words without knowing the context the word is given sentences	Lemmatization is slower as compared to stemming but is knows the context of the word before proceeding.
2	It is a rule based approach	It is a dictionary based approach.
3	Accuracy is less,	Accuracy is more as compared to stemming.
4	When we convert any word into root from then stemming may create the non-existence meaning of word.	Lemmatization always gives the dictionary meaning word while converting into root form

Sr. No	Stemming	Lemmatization
5	Stemming is preferred when the meaning of the word is not important of analysis Example : Spam detection	Lemmatization would be recommended when the meaning of the word is important for analysis. Example : Question answer
6.	For example : "Studies"=> "Studi"	For example : "Studies"=> "Study"

2.5 Regular Expression

- Regular expressions are also called as regexes. It is used for pattern matching standards for string passing and replacement.
- Regular expressions are powerful way to find and replace string that take a defined format. For example, regular expressions are used to parse email addresses, url's, dates, log files, configuration file, command line, programming script or switches.
- Regular expression is a useful tool to design language compilers as well as they are used in natural language processing for tokenization, describing lexicons, morphological analysis, etc. Many of us have used simple form of regular expression for searching file patterns in MS DOS for example dir*.txt.
- The Unix based editor Ed regular expression for popular in computer science. Perl is the first language that provided integrated support for regular expression. It used slash (/) around each regular expression here we are also following the same notation. however, slashes are not a part of regular expression.
- Regular expressions introduced in 1956 by Kleene. It was originally studied as part of theory of computation. Regular expression is an algebraic formula whose value is a pattern consisting of a set of Strings known as the language of expression.
- The simple type of regular expression contains a single symbol.
- For example the expression : /a/ - The expression denotes a set containing a string 'a'.
- It also pacifies sequence of characters.
- For example : /avengers/ - It denotes Indian notes that contain screen avengers and nothing else.
- **Brackets:** Characters are group by putting them between square brackets. This way any character in the class will match One character in the input.

- **For example,**

  /[abcd]/ will match any of a, b, c, and d.

  / [0123456789]/ specifies any single digit.

- **Range:**Sometimes regular expression led to cumbersome notation.

  For example:

  /[abcdefghijklmnopqrstuvwxyz] - It specifies any lowercase letter.

- In such cases a dash is used to specify a range.
- **For Example:**

  / [3-6]/ specifies any one of the digits 3,4,5, or 6.

  T /[c-f]/ specifies any one of the letter c,d,e, or f.

- **Caret^**: The caret is used at the beginning of the regular expression to specify what a single character cannot be.
- **For example:**

  /[^x] - matches any single character except x

  /[^A-Z]/ --> not an upper-case letter

  /[^Tt]/ --> neither 'T' nor 't'

  /[^\.]/ --> not a period

  /[p^]/ --> either 'p' or '^'

  /x^y/ --> the pattern 'x^y'

- Regular expressions are case sensitive. the pattern /t/ matches *t* but not *T*. It means pattern /Tree/ will not match pattern/tree/. To solve this issue the regular expression for this is written as / [Tt]ree/.
- *** or +** : The use of * or + allows you to add 1 or more of a preceding character.
- **For example:**

  oo*h! → 0 or more of a previous character (e.g. ooh!,ooooh!)

  o+h! → 1 or more of a previous character (e.g. ooh!,oooooh!) .

  paa+ →paa, paaaa, paaaaaa, paaaaaaa

- **?** : The question mark ? letters optionality of the previous expression.

- **For example:**

  /woodchucks?/ --> woodchuck or woodchucks

  /colou?r/ --> color or colour

- **Anchor:** These are special characters to accomplish string operations at the start and end of a text input.

  '^' — specifies the start of the string. The character followed by the '^' in the pattern should be the first character of the string in order for a string to match the pattern.

  '$' — specifies the end of the string. The character that precedes the '$' in the pattern should be the last character in the string in order for the string to match the pattern.

- The following are some special characters:

  . → any character except a new line

  \\w → any word character

  \\W → anything but a word character

  \\d → any digit character

  \\D → anything but a digit character

  \\b → a word boundary

  \\B → anything but a word boundary

  \\s → any space character

  \\S → anything but a space character

2.6 Finite Automata

- An automaton having a finite number of states is named a Finite Automaton (FA) or Finite State automata (FSA). Finite automata are used to identify patterns. It takes the string of symbol as input and changes its state accordingly. When the required symbol is found, then the transition happens.
- When transition takes place, the automata can either move to the succeeding state or stay in the similar state. there are two states in Finite automata: Accept state or Reject state. When the input string is processed successfully, and the automata reached its final state, then it will accept.
- Mathematically, an automaton can be represented by a 5-tuple (Q, Σ, δ, q0, F), where –

  Q -is a finite set of states.

Σ -is a finite set of symbols, called the alphabet of the automaton.

δ -is the transition function

q0 -is the initial state from where any input is processed (q0 ∈ Q).

F- is a set of final state/states of Q (F ⊆ Q).

- There are two types of Finite Automata
  - Deterministic Finite automation (DFA)
  - Non-deterministic Finite Automation (NDFA)

Deterministic Finite automation (DFA)

It may be defined as the type of finite automation wherein, for every input symbol we can determine the state to which the machine will move. It has a finite number of states that is why the machine is called Deterministic Finite Automaton (DFA).

- Mathematically, a DFA can be represented by a 5-tuple (Q, Σ, δ, q0, F), where –

  Q- is a finite set of states.

  Σ- is a finite set of symbols, called the alphabet of the automaton.

  δ -is the transition function where δ: $Q \times \Sigma \rightarrow Q$.

  q0 -is the initial state from where any input is processed (q0 ∈ Q).

  F- is a set of final state/states of Q (F ⊆ Q).

- However, graphically a DFA can be represented by diagraphs called state diagramis where -

Vertices	It represents states
Labeled arcs	It represents transitions
Empty incoming arc	It represents initial state
Double circle	It represents final state

- **Example:**

$\Sigma = \{0, 1\}$,

$Q = \{q_0, q_1, q_2, q_3, q_4\}$,

start state is q_0

and final state is q_4

- The following are the rules for transition.
  1. from state q_0 and with input a go to state q_1
  2. from state q_1 and with input b go to state q_2
  3. from state q_1 and with input c go to state q_3
  4. from state q_2 and with input b go to state q_4
  5. from state q_3 and with input b go to state q_4
- State transition diagram is as shown in Fig. 2.6.1.

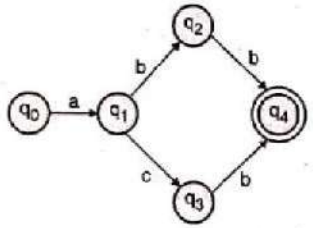


Fig. 2.6.1

Non-deterministic Finite Automation (NDFA)

Non-deterministic Finite Automation is defined as the type of finite automation where for each input symbol we cannot determine the state to which the machine will move i.e., the machine can move to any combination of the states. It means for each state there can be more than one transition on a given symbol, each lead to a different state. It has a finite number of states due to this the machine is called Non-deterministic Finite Automation.

Mathematically, NDFA can be given by a 5-tuple (Q, Σ, δ, q0, F), where –

- Q - is a finite set of states.
- Σ- is a finite set of symbols, called the alphabet of the automaton.
- δ -is the transition function where δ: $Q \times \Sigma \rightarrow 2^Q$.
- q0 - is the initial state from where any input is processed (q0 ∈ Q).
- F - is a set of final state/states of Q (F ⊆ Q).

Graphically the NDFA is represented same as DFA. Consider the same example of DFA. There are two possible transitions from state q_0 input symbol a. The transition diagram for NDFA is as shown in Fig.2.6.2

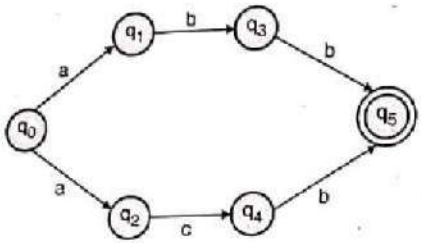
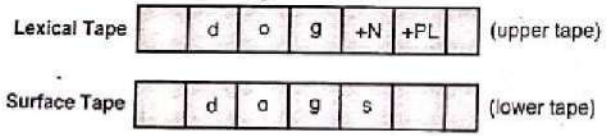


Fig. 2.6.2

2.7 Finite State Transducers (FST)

- Two-level morphology represents the correspondence between lexical and surface levels. The finite-state transducer is used to find mapping between these two levels.
- A FST is a two-tape automaton: Reads from one tape, and writes to other one.FST defines relations between sets of strings.
- For morphological processing, one tape holds lexical representation, the second one holds the surface form of a word.



- An FSA represents a set of strings. e.g.{walk, walks, walked, love loves, loved}, while FST represents a set of pairs of strings (think of asinput, output pairs),

  For example: {(walk, walk+V+PL), (walk, walk+N+SG), (walked, walk+V+PAST) ...}

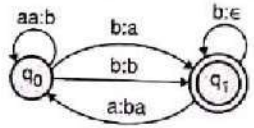


Fig. 2.7.1: A finite-state transducer

- The FST is a multi-function device, and can be viewed in the following ways:
  - Translator: It reads one string on one tape and outputs another string,
  - Recognizer: It takes a pair of strings as two tapes and accepts/rejects based on their matching.

  - Generator: It outputs a pair of strings on two tapes along with yes/no result based on whether theyare matching or not.
  - Relater: It computes the relation between two sets of strings available on two tapes
- Finite State Transducer definition is

  FST is $Q \times \Sigma \times q_0 \times F \times \delta$
  - **Q** : a finite set of N states $q_0, q_1, \ldots q_N$
  - **Σ**: a finite input alphabet of complex symbols.
  - Each complex symbol is a pair of an input and an output symbol i:o
  - where **i** is a member of I (an input alphabet),
  - and **o** is a member of O (an output alphabet).
  - I and O may contain empty string.
  - So, Σ is a subset of IxO.
  - **q_0** : the start state
  - **F** : the set of final states -- F is a subset of Q
  - **δ(q,i:o)** : transition function
- Σ may not contain all possible pairs from IxO.
- For example:
  - I = {a, b, c} O={a,b,c, ϵ}
  - Σ = {a:a, b:b, c:c, a:ϵ, b: ϵ, c: ϵ}
- **feasible pairs** – In two-level morphology terminology, the pairs in Σ are called as feasible pairs.
- **default pair** – Instead of a:a we can use a single character for this default pair.
- FSAs are isomorphic to regular languages, and FSTs are isomorphic to regular relations (pair of strings of regular languages).
- **Properties of FST :** the properties of FST are union, inversion, and composition.
  - **Union :** The union of two regular relations is also a regular relation.
  - **Inversion :** The inversion of a FST simply switches the input and output labels. This means that the same FST can be used for both directions of a morphological processor. The inversion of a transducer T (T^{-1}) simply switches the input and output labels. Thus, if T maps from the input alphabet I to the output alphabet O,T^{-1} maps from O to I.

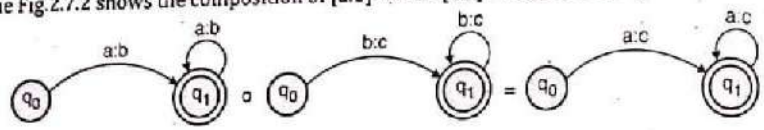
$T = \{(a,a1)\ (a,a2)\ (b,b1),\ (c,c1),\ (c,a)\}$

$T^{-1} = \{(a1,a),\ (a2,a),\ (b1,a),\ (c1,c),\ (a,c)\}$

- **Composition :** If T1 is a FST from I1 to O1 and T2 is a FST from O1 to O2, then composition of T1 and T2 (T1oT2) maps from I1 to O2.If T1 is a transducer from I1 to O1 and T2 a transducer from O1 to O2, then $T1 \circ T2$ maps from I1 to O2.

  So the transducer function is : $(T1 \circ T2)\ (x) = T1(T2(x))$

The Fig.2.7.2 shows the composition of [a:b]+ with [b:c]+ to produce[a:c]+



- he Fig.2.7.3 shows the a FST for simple english nominals that has regular nouns, singular nouns and irregular noun

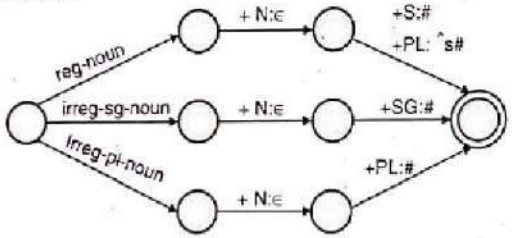


Fig. 2.7.3

- **A FST for stems–** FST maps the s roots to their root-class

reg-noun	irreg-pl-noun	irret-sg-noun
fox	g o :e o:e se	goose
cat	sheep	sheep
dog	m o:l U:ε s:c e	mouse

- For example: fox stands for f: f o:o x:x
- When these two transducers are composed, we have a FST which maps lexical forms to intermediate forms of words for simple English noun inflections.
- Next thing that we should handle is to design the FSTs for orthographic rules, and combine all these transducers.

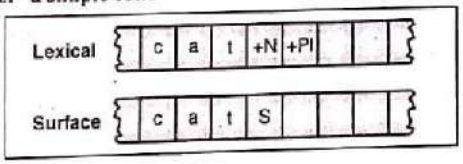
We use these properties of FSTs in the creation of the FST for a morphological processor.

2.8 Morphological Parsing With FST

- The objective of the morphological parsing is to produce output lexicons for a single input lexicon. let's consider, parsing just the productive nominal plural (-s) and the verbal progressive (-ing). Our aim is to take input forms like those in the first column below and produce output forms like those in the second column. The second column contains the stem of each word as well as mixed morphological features. These features specify additional information about the stem. For example: +SG - singular, +PL – plural.

Cats	cat +N +PLU
Cat	cat +N +SG
Goose	goose +N +SG or goose +V
Geese	goose +N +PLU
Gooses	goose +V +3SG
Catch	catch +V
Caught	catch +V +PAST or catch +V +PP

- The column contains the stem of the corresponding word (lexicon) in first column, along with its morphological features, like, +N means word is noun, +SG means it is singular, +PL means it is plural, +V for verb, and pres-part for present participle. There can be more than one lexical level representation for a given word
- We achieve it through two level morphology, which represents a word as a correspondence between lexical level - a simple concatenation of lexicons, as shown in column 2 of table



For a morphological processor, we need at least followings:

- **Lexicon** : The list of stems and affixes together with basic information about them such as their main categories (noun, verb, adjective, ...) and their sub-categories (regular noun, irregular noun, ...).
- **Morphotactics** : The model of morpheme ordering that explains which classes of morphemes can follow other classes of morphemes inside a word.

- An FSA for another fragment of English derivational morphology

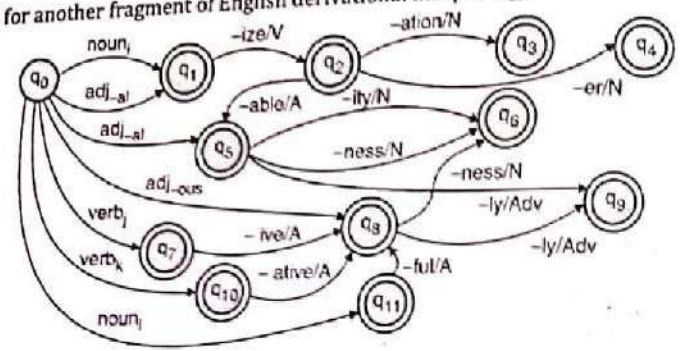


Fig. 2.8.4

- **Compose :** the figure shows the fleshed-out English nominal inflection FST Tlex, expanded from Tnum by replacing the three arcs with individual stem work the rules are demonstrated in the table.

Name	Description of Rule	Example
Consonant doubling	1 letter consonant doubled before-ing/-ed	beg/begging
E deletion	Silent e dropped before -ing and -ed	Make/making
E insertion	e added after -s,-z,-x,-ch, -sh before -s	Watch/watches
Y replacement	-y changes to -ie before -s, -I before -ed	Try/tries
K insertion	verbs ending with vowel + -c add-k	Panic/paniked

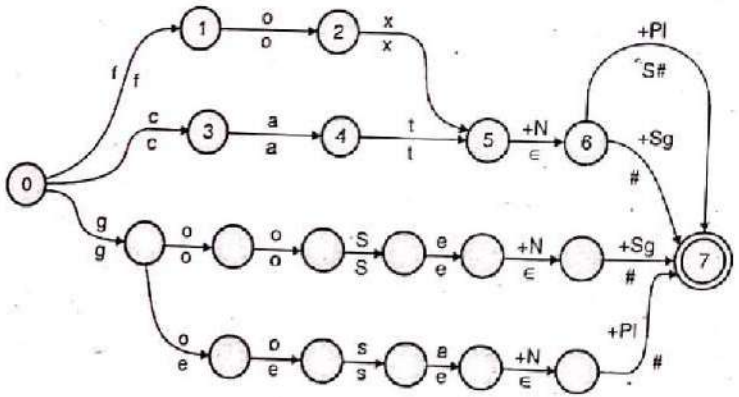


Fig. 2.8.5

2.9 Lexicon Free FST Porter Stemmer

- There are some informational retrieval applications that do not perform the whole morphological processor. They only need the stem of the word. It is just a cascaded rewrite rule.
- In this the Output of one stage is the input for the next
- It is based on a series of simple cascade rules-

  ATIONAL → ATE (relational → relate)

  ING →ε (motoring → motor)

  SSES → SS (grasses → grass)
- A stemming algorithm (Port Stemming algorithm) is a lexicon-free FST.
- Stemming algorithms are efficient but they may introduce errors because they do not use a lexicon.
- Some errors of commission are:

  ORGANIZATION - ORGAN

  DOING – DOE

  GENERALIZATION- GENERIC

  NUMERICAL- NUMEROUS

  POLICY-POLICE
- Some errors of omission are :

  European -Europe

  analysis- analyzes

  matrices- matrix

  noise-noisy

  spare- sparsity

2.10 N –Grams- N-Gram Language Model

- An N-gram is a sequence of N words. Let's understand N-gram with an example. Consider the following sentence: "I love reading books about data science on Analytics Swati."
- A 1-gram/ unigram is a one-word sequence. For the given sentence, the unigrams would simply be: "I", "love", "reading", "books", "about", "data", "science", "on", "Analytics", "Swati".

- A 2-gram/bigram is a two-word sequence of words, such as "I love", "love reading", or "Analytics Swati".
- A 3-gram/ trigram is a three-word sequence of words like "I love reading", "about data science" or "on Analytics Swati".
- An N-gram language model predicts the probability of a given N-gram within any sequence of words in the language. If we have a good N-gram model, we can predict p(w | h) – what is the probability of seeing the word w given a history of previous words h – where the history contains n-1 words.
- The estimation of probability of the of a sentence is achieved by decomposing sentence probability into a product of conditional probabilities. We compute this probability in two steps:
  1. Apply the chain rule of probability
  2. Apply a very robust simplification assumption to let us to compute p(w1...ws) in an easy manner.

  apply chain of custody rule as follows:

$$P(s) = P(w_1 w_2 w_3 \dots w_n)$$

$$= P(w_1)\, P\left(\frac{w_2}{w_1}\right) P\left(\frac{w_3}{w_1 w_2}\right) w_1 P\left(\frac{w_3}{w_1 w_2 w_3}\right) P\left(\frac{w_3}{w_1 w_2 w_3 \dots w_n}\right)$$

$$= \pi_{i=1}^{n} P\left(\frac{w_i}{h_i}\right)$$

where h_i is history and w_i is defined as $(w_1 w_2 w_3 \dots w_{i-1})$

so, to calculate the word probability we have to calculate the probability of the word, given the sequence of word preceding it.

- An n-gram model simplifies the task by approximating the probability of the word given all the previous words by the conditional probability given previous n-1 word only.

$$P\left(\frac{w_i}{h_i}\right) = P\left(\frac{w_i}{w_{i-n+1}\, w_{i-1}}\right)$$

Hence, N-gram model calculates $P\left(\frac{w_i}{h_i}\right)$ by modelling language as Markov model of order n-1. It means by looking at the previous n-1 word only.

using the bigram and the trigram model the probability is estimated as:

$$P(s) = \pi_{i=1}^{n} P\left(\frac{w_i}{w_{i-1}}\right)$$

and trigram model,

$$P(s) = \pi_{i=1}^{n} P\left(\frac{w_i}{w_{i-1}\, w_{i-1}}\right)$$

2.11 N-Gram for Spelling Correction

- In Computer Based information systems, errors of typing and spelling constitutes a very common source of variation between strings. These areas have been widely investigated. it has been observed that a single character insertion, omission, substitution and reversal are the most common typing mistakes.
- As per survey it is reported that over 80% of typing errors were single error spelling:
  1. single letter substitution
  2. single letter On Mission
  3. inserting a single letter
  4. transposition of two adjacent letters.
- Optical character recognition and other automatic reading devices introduce errors of substitution, insertion, and deletion but not of reversal.
- The substitution errors in OCR are occurring due to the visual similarity such as c-e, l-1, r-n and m-rn.
- Unlike typing errors, spelling errors are mainly phonetic, where the Misspell word is pronounced in the same way as the correct word. Phonetic errors are difficult to set right because they distort the word by more than a single insertion, substitution audition.
- The spelling errors belong to two categories named non word errors and real-world errors. When an error results in the world that does not appear in a given lexicon or is not a valid orthographic word form it is known as a non-word error.
- To detect the non-word error to main techniques are used name n gram analysis and dictionary lookup.
- The real-world error result in actual words of the language it occurs because of the typographical mistakes or due to spelling errors, for example, substituting the spelling of a homophones or near- homophone, piece of peace or meat for meet.
- Spelling correction consist of detecting and correcting errors. error detection is the process of finding the misspelled word and error correction is the process of suggesting correct words to a misspelled word.

- The n-gram can be used for the both non word and real-world errors detection because in the English alphabet certain bigram and trigram of letters never occur or rarely do so.
- For example, the trigram qst and bigram qd. this information can be used to handle non word error. strings that contain this unusual n-grams can be identified as possible spelling errors. n-grams Technique generally required a large Corpus or dictionary as training data so that an n gram table of possible combinations of letter can be compiled.
- Special word <s> is used to beginning of the word in bi-gram.
- N gram uses chain of custody rule as follows:

$$P(s) = P(w_1 w_2 w_3 \dots w_n)$$

$$= P(w_1) P\left(\frac{w_2}{w_1}\right) P\left(\frac{w_3}{w_1 w_2}\right) w_1 P\left(\frac{w_3}{w_1 w_2 w_3}\right) P\left(\frac{w_3}{w_1 w_2 w_3 \dots w_{n-1}}\right)$$

$$= \pi_{i=1}^{n} P\left(\frac{w_i}{h_i}\right)$$

for example:

1. Traning set is

- The Arabian Nights
- These are the fairy tales of the east
- The stories of the Arabian Nights are translated in many language

2. Bi-gram Model

P(the.<s>) = 0.67

P (are/these) = 1.0

P(tales/fairy) = 1.0

P(east/the) = 0.2

P(are/knights) = 1.0

P(many/in) = 1.0

P(languages/many) = 1.0

P(Arabian/the) = 0.4

P(the/are) = 0.5

P(of/tales) = 1.0

P(stories/the) = 0.2

P(translated/are) = 0.5

P(Knights/Arabian) = 1.0

P(fairy/the) = 0.2

P(the/of) = 1.0

P(of/stories) = 1.0

P(in/translated) = 1.0

3. Test sentence(s)

The Arabian Nights are the fairy tales of the east

P(The/<s>) × P(Arabian/the) ×P(Knights/Arabian) × P(are/knights) × P(the/are) × P(fairy/the)P(tales/fairy) × P(of/tales) × P(the/of) × P(east/the)

= 0.67 × 0.5 × 1.0 × 1.0 × 0.5 × 0.2 × 1.0 × 1.0 × 1.0 × 0.2

= 0.0067

- As each probability is necessarily less than 1, if you multiply the probability it might cause a numerical and a few, normally in long sentences. to avoid these calculations are made in log space, where calculation corresponds to adding log of individual probabilities and taking antilog of the sum.
- The n-gram model suffers from data sparseness problems. The n-gram that does not occur in the training data is assigned zero probability. Show the large corpus even have many zero entries in its bigram matrix. the smoothing techniques are used to handle the data sparseness problem.
- smoothing is generally referring to the task of re- evaluating zero probability or low probability n-grams and assigning them non zero values.
- **Add one smoothing:**

This Technique adds a value of one to each n-gram frequency before normalising them into probabilities. So, the conditional probability becomes

$$P\left(\frac{w_i}{w_{n+1} w_{ni-1}}\right) = \frac{C(W_{i-n+1}, W_{i-1}, W_i)}{C(W_{i-n+1} \dots W_{i-1}) + V}$$

V is vocabulary size.

Review Questions

Q.1 What is morphology parsing ?

Q.2 write short note on finite automata ?

Q.3 Explain Finite transducer ?

Q.4 Explain N-gram model ?

Q.5 How n-gram model is used for spelling correction ?

Q.6 Explain Lexicon Free FST Porter Stemmer.

□□□

3

Module - 3

Syntax Analysis

Syllabus

Part-Of-Speech tagging(POS)- Tag set for English (Penn Treebank) , Rule based POS tagging, Stochastic POS tagging, Issues –Multiple tags & words, Unknown words. Introduction to CFG, Sequence labeling: Hidden Markov Model (HMM), Maximum Entropy, and Conditional Random Field (CRF).

TOPICS

If words are the foundation of speech and language processing, syntax is the skeleton. Syntax is the study of formal relationships between words. These six chapters study how words are clustered into classes called parts-of-speech, how they group with their neighbors into phrases, and the way words depends on other words in a sentence. The section explores computational models of all of these kinds of knowledge, including context-free grammars, lexicalized grammars, feature structures, and meta theoretical issues like the Chomsky hierarchy.

It introduces fundamental algorithms for dealing with this knowledge, like the Earley and CYK algorithms for parsing and the unification algorithm for feature combination. It also includes probabilistic models of this syntactic knowledge, including HMM part-of speech taggers, and probabilistic context-free grammars. Finally, this section will explore psychological models of human syntactic processing.

3.1 Part-Of-Speech Tagging (POS)

- When Words are grouped into similar classes which can be called as Parts of speech (POS), word classes, morphological classes, or lexical tags, these classes give information about a word and its neighbours.
- The Greeks or traditional grammars has 8 basic Part Of Speech like Noun, verb, pronoun, preposition, adverb, conjunction, adjective, and article. Modern models has much larger numbers of extended list of POS, 45 for the Penn Treebank, 87 for the Brown corpus, and 146 for the C7 tagset.
- Tag sets for example distinguish between possessive pronouns like *my, your, his, her, its* and personal pronouns like *I, you, he, me*. Knowledge about words arranged or occurred in sentences for example possessive pronouns are likely to be followed by a noun, personal pronouns by a verb, can be useful in a language model for speech recognition.
- How word is pronounced? *CONtent* and *conTENT* based on their pronunciation considered as noun and adjective respectively. Speech synthesis system and speech recognition system can be understand by knowing part of speech like again *OBject* (noun) and obJECT(verb).
- So, now we can define Part-Of-Speech (POS) as it is process of assigning a tag to a word in a corpus.

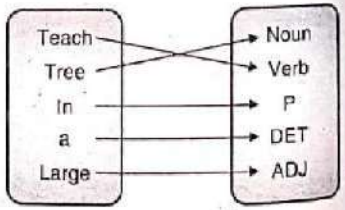


Fig. 3.1.1 : Part-Of-Speech (POS)

3.1.1 Part of Speech Categories

- Parts of speech can be divided into two broad super categories: **closed class** types and **open class** types. Closed classes are those that have relatively fixed membership. For example, prepositions are a closed class because there is a fixed set of them in English; new prepositions are rarely coined.

- By contrast nouns and verbs are open classes because new nouns and verbs are continually coined or borrowed from other languages. Closed class words are generally also **function words**; function words are grammatical words like *of, it, and,* or *you*, which tend to be very short, occur frequently, and play an important role in grammar.
- **Closed classes** which are function words include prepositions, pronouns, determiners, conjunctions, numerals, auxiliary verbs and particles (preposition or adverbs in phrasal verbs).
- The closed classes differ more from language to language than do the open classes. Here's a quick overview of some of the more important closed classes in English, with a few examples of each:
  - **Prepositions :** on, under, over, near, by, at, from, to, with
  - **Determiners :** a, an, the
  - **Pronouns :** she, who, I, others
  - **Conjunctions :** and, but, or, as, if, when
  - **auxiliary verbs :** can, may, should, are
  - **particles :** up, down, on, off, in, out, at, by
  - **numerals :** one, two, three, first, second, third

There are four major open classes that occur in the languages of the world: **nouns, verbs, adjectives,** and **adverbs.**

1. **Nouns :** people, place and things proper nouns, common nouns, count nouns and mass nouns
2. **Verbs :** actions and processes. Main verbs, not auxiliaries
3. **Adjectives :** Most languages have adjectives for the concepts of color (*white, black*), age (*old, young*),and value (*good, bad*)
4. **Adverbs :** The final open class form, adverbs, is rather a mixture, both semantically and formally. For example in a given sentence like the following, all the italic words are adverbs :

   Unfortunately, John walked home extremely slowly yesterday

Tagsets for English

- There are a small number of popular tagsets for English, many of which evolved from the 87-tag tagset used for the Brown corpus.
- Three of the most commonly used are the small 45-tag Penn Treebank tagset, the medium-sized 61 tag C5 tagset and the larger 146-tag C7 tagset.
- The Penn Treebank tagset, shown in Fig. 3.1.1 , has been applied to the Brown corpus and a number of other corpora.

- Here is an example of a tagged sentence from the Penn Treebank version of the Brown corpus (in a flat ASCII file, tags are often represented after each word, following a slash, but tags can also be represented in various other ways), for e.g. consider the sentence "The grand jury commented on a number of other topics." This can be represented with tagset as follows:

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Tag	Description	Example
CC	Coordin Conjuction	and, but, or
CD	Cardinal number	One, two, three
DT	Determine	a, the
EX	Existential 'there'	there
FW	Foregin work	mea culpa
IN	Preposition/sub-conj	Of, in, by
JJ	Adjective	Yellow
JJR	Adj., comparative	Bigger
LS	List item maker	1, 2, one
MD	Modal	Can should
NN	Noun, sing. Or mass	llama
NNS	Noun, plural	llamas
NNP	Proper noun, singular	IBM
NNPS	Proper noun, plural	Carolinas
PDT	Possessive ending	All both
POS	Possessive ending	'S
PP	Personal pronoun	I, You, he
PP$	Possessive pronoun	your; one's
RB	Adverb	Quickly, never
RBR	Adverb, comparative	faster
RBS	Adverb, superlative	Fastest
RP	Particle	Up, off

Tag	Description	Example
SYM	Symbol	+, %, &
TO	"to"	To
UH	Interjection	ah, opps
VB	Verb, base form	Eat
VBD	Verb, past tense	ate
VBG	Verb, gerund	Eating
VBN	Verb, past participle	Eaten
VBZ	Verb, 3sg pres	Eats
WDT	Wh-determine	Which, that
WP	Wh-pronoun	What, who
WP$	Possessive wh-	Whose
WRB	Possessive wh-	Whose
$	Dollar sign	$
#	Pound sign	#
"	Left quote	('or ")
"	Right quote	('or")
(	Left parenthesis	([, (, {, <)
)	Right parenthesis	(],), }, >)
,	Comma	,
.	Sentence-final punc	(. ! ?)
:	Mid-sentence punc	(: ; ... – -)

Fig. 3.1.1: Penn Treebank Part-Of-Speech Tagset

3.1.2 Part-Of-Speech Tagging

- Part-of-speech tagging (or just **tagging** for short) is the process of assigning a part-of-speech or other lexical class marker to each word in a corpus. Tags are also usually applied to punctuation markers; thus tagging for natural language is the same process as **tokenization** for computer languages, although tags for natural languages are much more ambiguous.
- The input to a tagging algorithm is a string of words and a specified **tagset**.

VB	DT	NN
Book	that	flight.

- Tagging algorithms automatically choose multiple tags for single word and select only one best appropriate tag for that word. Although, tagging can be hard as it face lot of disambiguation. E.g. word *book* can be consider as verb for *book that flight* and can be noun for *please give me that book*.

3.1.3 Methods for Part-Of-Speech(POS) Tagging

- Most tagging algorithms fall into one of two classes: **rule-based** taggers and **stochastic** taggers. Rule-based taggers generally involve a large database of hand-written disambiguation rule which specify, for example, that an ambiguous word is a noun rather than a verb if it follows a determiner.
- Stochastic taggers generally resolve tagging ambiguities by using a training corpus to compute the probability of a given word having a given tag in a given context.

Rule-Based Part-Of-Speech Tagging

- Rule-based POS tagging uses ENGTWOL tagger which based on a two-stage architecture. The first stage used a dictionary to assign each word a list of potential parts of speech. The second stage used large lists of hand-written disambiguation rules to winnow down this list to a single part-of-speech for each word.
- The ENGTWOL lexicon is based on the two-level morphology, and has about 56,000 entries for English word stems, counting a word with multiple parts of speech (e.g. nominal and verbal senses of *hit*) as separate entries, and inflected and many derived forms not counted. Each entry is annotated with a set of morphological and syntactic features. Fig. 3.1.2 shows some selected words, together with a slightly simplified listing of their features.

Word	POS	Additional POS features
Smaller	ADJ	COMPARATIVE
Entire	ADJ	ABSOLUTE ATTRIBUTIVE
Fast	ADV	SUPERLATIVE
That	DET	CENTRAL DEMONSTRATIVE SG
All	DET	PREDETERMINER SG/PL QUANTIFIER
Dog's	N	GENITIVE SG
Furniture	N	NOMINATIVE SG NONINDEFETERMINER
One-third	NUM	SG
She	PRON	PERSONAL FEMININE NOMINATIVE SG3
Show	V	IMPERATIVE VFIN
Show	V	PRESENT -SG3 VFIN
Shown	PCP2	SVOO SVO SV
Occurred	PCP2	SV
Occurred	V	PAST VFIN SV

Fig. 3.3 Sample lexical entries from the ENGTWOL lexicon.

1. In the first stage of the tagger, each word is run through the two-level lexicon transducer and the entries for all possible parts of speech are returned.

Stage 1 of ENGTWOL Tagging

First Stage : Run words through FST morphological analyzer to get all parts of speech.

- Example sentence : Pavlov had shown that salivation ...

PAVLOV	**N SG PROPER**
HAVE	**V PAST VFIN SVO (verb with subject and object)**
HAVE	**PCP2(past participle) SVO**
SHOWN	**SHOW PCP2 SVO SV SVOO (verb with subject and two complements)**
THAT	**ADV**
	PRON DEM SG
	DET CENTRAL DEM SG
	CS (subordinating conjunction)
SALIVATION	**N SG**

- A set of about 1,100 constraints are then applied to the input sentence to rule out incorrect parts of speech; the boldfaced entries in the given above show the desired result, in which the preterite (not participle) tag is applied to *had*, and the complementizer (CS) tag is applied the *that*. The constraints are used in a negative way, to eliminate tags that are inconsistent with the context. In stage 2 of ENGTWOL tagging we can write rule for word "that" to eliminate unwanted tags which initially assigned to it in stage one.

Stage 2 of ENGTWOL Tagging

Second Stage : Apply NEGATIVE constraints.

- Example :

```
ADVERBIAL-THAT RULE
Given input: "that"
if
(+1 A/ADV/QUANT); /*if next word is adj/adv/quantifier */
(+2 SENT-LIM)     ; /*following which is E-O-S */
(NOT -1 SVOC/A)   ; /*and the previous word is not averb like "consider" */
                    /* which allows adjective complements in "I consider that odd"*/
then eliminate non-ADV tags
else eliminate ADV tag
```

- The first two clauses of this rule check to see that the *that* directly precedes a sentence-final adjective, adverb, or quantifier. In all other cases the adverb reading is eliminated.

Stochastic Part-Of-Speech Tagging

Stochastic taggers generally resolve tagging ambiguities by using a training corpus to compute the probability of a given word having a given tag in a given context. Stochastic tagger called also **HMM tagger** or a **Maximum Likelihood Tagger**, or a **Markov model HMM TAGGER tagger**, based on the Hidden Markov Model.

The simplest stochastic tagger applies the following approaches for POS tagging –

1. Approach 1 : Word Frequency Approach

In this approach, the stochastic taggers disambiguate the words based on the probability that a word occurs with a particular tag. We can also say that the tag encountered most frequently with the word in the training set is the one assigned to an ambiguous instance of that word. The main issue with this approach is that it may yield inadmissible sequence of tags.

- Assign each word its most likely POS tag
  - If w has tags t1, ..., tk, then can use

$$P(ti \mid w) = c(w,ti)/(c(w,t1) + \ldots + c(w,tk)), \text{ where}$$

$$c(w,ti) = \text{number of times w/ti appears in the corpus}$$

- Success: 91% for English

- Example heat :: noun/89, verb/5

2. Apprach 2 : Tag Sequence Probabilities

It is another approach of stochastic tagging, where the tagger calculates the probability of a given sequence of tags occurring. It is also called n-gram approach. It is called so because the best tag for a given word is determined by the probability at which it occurs with the n previous tags.

1. Given : sequence of words W

$$W = w_1, w_2, \ldots, w_n \text{ (a sentence)}$$

– e.g., W = heat water in a large vessel

2. Assign sequence of tags T :

$$T = t_1, t_2, \ldots, t_n$$

3. Find T that maximizes $P(T \mid W)$

3.2 Other Issues

3.2.1 Multiple Tags and Multiple Words

- Two issues that arise in tagging are tag indeterminacy and multi-part words.
- Tag indeterminacy arises when a word is ambiguous between multiple tags and it is impossible or very difficult to disambiguate. In this case, some taggers allow the use of multiple tags. This is the case in the Penn Treebank and in the British National Corpus. Common tag indeterminacies include adjective versus preterit versus past participle (JJ/VBD/VBN), and adjective versus noun as prenominal modifier (JJ/NN).
- The second issue concerns multi-part words. The C5 and C7 tagsets, for example, allow prepositions like *'in terms of'* to be treated as a single word by adding numbers to each tag.

3.2.2 Unknown Words

- All the tagging algorithms require a dictionary that lists the possible parts of speech of every word. But the largest dictionary will still not contain every possible word. Proper names and acronyms are created very often, and even new common nouns and verbs are added in the language at a surprising rate. Therefore in order to build a complete tagger we need some method for guessing the tag of an unknown word.

- The simplest possible unknown-word algorithm is to pretend that each unknown word is ambiguous among all possible tags, with equal probability. Then the tagger must rely solely on the contextual POS-trigrams to suggest the proper tag. A slightly more complex algorithm is based on the idea that the probability distribution of tags over unknown words is very similar to the distribution of tags over words that occurred only once in a training set.

3.3 Introduction to CFG

The word **syntax** meaning 'setting out together or arrangement', and refers to the way words are arranged together. This chapter and the following ones introduce a number of more complex notions of syntax and grammar. There are three main new ideas: **constituency, grammatical relations,** and **subcategorization and dependencies.**

3.3.1 Constituency

- Groups of words may behave as a single unit or phrase, called a constituent; also Sentences have parts, some of which appear to have subparts. These groupings of words that go together we will call constituents.
- Based on how words are formed, the standard grouping is usually called the noun phrase. This is a sequence of words surrounding by at least one noun. Let's understand with some examples,
  - *Kermit the frog*
  - *They*
  - *December twenty-sixth*
  - *the reason he is running for president*
- How do we know they go together? Consider the following sentences with grouping of words to form constituents,
  - *I hit the man with a cleaver*
  - *I hit [the man with a cleaver]*
  - *I hit [the man] with a cleaver*
  - *You could not go to her party*
  - *You [could not] go to her party*
  - *You could [not go] to her party*

Constituent Phrases

For constituents, we usually name them as phrases based on the word that heads the constituent :

the man from Amherst	is a Noun Phrase (NP) because the head man is a noun
extremely clever	is an Adjective Phrase (AP) because the head clever is an adjective
down the river	is a Prepositional Phrase (PP) because the head down is a preposition
killed the rabbit	is a Verb Phrase (VP) because the head killed is a verb

- Note that a word is a constituent (a little one). Sometimes words also act as phrases.
- Consider a sentence *Joe grew potatoes.* In this sentence *Joe* and *potatoes* are both nouns and noun phrases.
- Now consider another sentence which can be comparing with *The man from Amherst grew beautiful russet potatoes.*
- We say Joe counts as a noun phrase because it appears in a place that a larger noun phrase could have been.

Evidence constituency exists

There are cases when constituents appear in similar environment and constituents can be placed in number of different locations

Case 1 : They appear in similar environments (before a verb)

- Kermit the frog comes on stage
- They come to Massachusetts every summer
- December twenty-sixth comes after Christmas
- The reason he is running for president comes out only now.
- Exceptions are, not each individual word in the constituent

  **The comes our... *is comes out... *for comes out...*

Case 2 : The constituent can be placed in a number of different locations

Constituent = Prepositional phrase: *On December twenty-sixth*

- On December twenty-sixth. I'd like to fly to Florida.
- I'd like to fly on December twenty-sixth to Florida.

- I'd like to fly to Florida on December twenty-sixth.
- Exceptions are, phrase cannot split apart

  **On December I'd like to fly twenty-sixth to Florida.*

  **On I'd like to fly December twenty-sixth to Florida.*

3.3.2 Context-Free Grammars

- The most commonly used mathematical system for modeling constituent structure in English and other natural languages is the Context-Free Grammar, or CFG. Context-free grammars are also called Phrase-Structure Grammars, and the formalism is equivalent to what is also called Backus-Naur Form or BNF.
- Grammars (and parsing) are key components in many applications

  1. Grammar checkers
  2. Dialogue management
  3. Question answering
  4. Information extraction
  5. Machine translation

- A context-free grammar consists of a set of rules or productions, each of which expresses the ways that symbols of the language can be grouped and ordered together, and a lexicon of words and symbols. For example, the following productions expresses that a NP (or noun phrase), can be composed of either a *ProperNoun* or of a determiner (*Det*) followed by a *Nominal*; a *Nominal* can be one or more *Nouns.*

$$NP \rightarrow \text{Det Nominal} \quad (3.1)$$

$$NP \rightarrow \text{ProperNoun} \quad (3.2)$$

$$\text{Nominal}' \rightarrow \text{Noun} \mid \text{Noun Nominal} \quad (3.3)$$

- Context free rules can be hierarchically embedded, so we could combine the previous rule with others like these which express facts about the lexicon :

$$\text{Det} \rightarrow \text{a} \quad (3.4)$$

$$\text{Det} \rightarrow \text{the} \quad (3.5)$$

$$\text{Noun} \rightarrow \text{flight} \quad (3.6)$$

- The symbols that are used in a CFG are divided into two classes. The symbols that correspond to words in the language ('the', 'nightclub') are called **terminal** symbols; the lexicon is the set of rules that introduce these terminal symbols.

- The symbols that express clusters or generalizations of these are called **nonterminal**. In each context-free rule, the item to the right of the arrow (→) is an ordered list of one or more terminals and nonterminal, while to the left of the arrow is a single nonterminal symbol expressing some cluster or generalization.
- Notice that in the lexicon, the nonterminal associated with each word is its lexical category, or part-of-speech.
- A CFG is usually thought of in two ways: as a device for generating sentences, or as a device for assigning a structure to a given sentence. As a generator, we could read the → arrow as 'rewrite the symbol on the left with the string of symbols on the right'. So starting from the symbol *NP*,

  we can use rule 3.1 to rewrite NP as,

  Det Nominal,

  and then rule 3.3,

  Det Noun,
- and finally via rules 3.4 and 3.6 as

  a flight,
- Together, these rules describe two kinds of NPs.
  1. One that consists of a determiner followed by a nominal
  2. And another that says that proper names are NPs.
  3. The third rule illustrates two things:
     - An explicit disjunction
     - A recursive definition
- We say the string a flight can be **derived** from the nonterminal NP. Thus, a CFG can be used to randomly generate a series of strings. This sequence of rule expansions is called a **derivation** of the string of words. It is common to represent a derivation by a **parse tree** (commonly shown inverted with the root at the top). Here is the tree representation of this derivation:

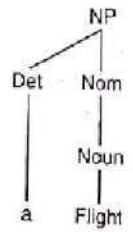


Fig. 3.3.1 A parse tree for 'a flight'

- The formal language defined by a CFG is the set of strings that are derivable from the designated **start symbol**. Each grammar must have one designated start symbol, which is often called *S*. Since context-free grammars are often used to define sentences, *S* is usually interpreted as the 'sentence' node, and the set of strings that are derivable from S is the set of sentences in some simplified version of English.
- Let's add to our sample grammar a couple of higher-level rules that expand *S*, and a couple others. One will express the fact that a sentence can consist of a noun phrase and **a verb phrase :**

  S →NP VP

  I prefer a morning flight
- A verb phrase in English consists of a verb followed by assorted other things; for example, one kind of verb phrase consists of a verb followed by a noun phrase:

  VP →Verb NP

  prefer a morning flight

  Or the verb phrase may have a noun phrase and a prepositional phrase:

  VP →Verb NP PP

  leave Boston in the morning

  Or the verb may be followed just by a preposition-phrase:

  VP →Verb PP

  leaving on Thursday

A prepositional phrase generally has a preposition followed by a noun phrase. For example, a very common type of prepositional phrase in the ATIS corpus is used to indicate location or direction:

PP →Preposition NP

from Los Angeles

- Fig. 3.3.2 gives a sample lexicon and Fig. 3.3.3 summarizes the grammar rules we have seen so far, which we'll call *L0*. Note that we can use the or-symbol | to indicate that a non-terminal has alternate possible expansions.
- It is sometimes convenient to represent a parse tree in a more compact format called **bracketed notation**, essentially the same as LISP tree representation; here is the bracketed representation of the parse tree of Fig. 3.3.4:

[S [NP [Pro I]] [VP [V **prefer**] [NP [Det a] [Nom [N **morning**] [N **flight**]]]]]

Noun	→	*flights / breeze / trip / morning / ...*
Verb	→	*is / prefer / like / need / want / fly*
Adjective	→	*cheapest / non-stop / first / latest / other / direct / ...*
Pronoun	→	*Alaska / Baltimore / Los Angles / Chicago / United / American / ...*
Determine	→	*the / a / an / this / these / that / ...*
Preposition	→	*from / to / on / near / ...*
Conjuction	→	*and / or / but / ...*

Fig. 3.3.2 : The possible lexicon for *Lo*

S	→	*NP VP*	*I + want a morning flight*
N	→	*Pronoun*	*I*
		/ Proper-Noun	*Los Angles*
		/ Det nominal	*a + flight*
Nominal	→	*Noun Nominal*	*morning + flight*
		/ Noun	*Flights*
VP	→	*Verb*	*do*
		/ Verb NP	*want + a flight*
		/ Verb NP Pp	*leave + Boston + in the morning*
		/ Verb PP	*leaving + on Thursday*
PP	→	*Preposition Np*	*from + Los Angeles*

Fig. 3.3.3: Each rule for example phrase(The grammar for *L0*)

- A CFG like L0 defines a formal language. A formal language is a set of strings. Sentences (strings of words) that can be derived by a grammar are in the formal language defined by that grammar, and are called grammatical sentences.
- Sentences that cannot be derived by a given formal grammar are not in the language defined by that grammar, and are referred to as ungrammatical. In linguistics, the use of formal languages to model natural languages is called generative grammar, since the language is defined by the set of possible sentences 'generated' by the grammar.

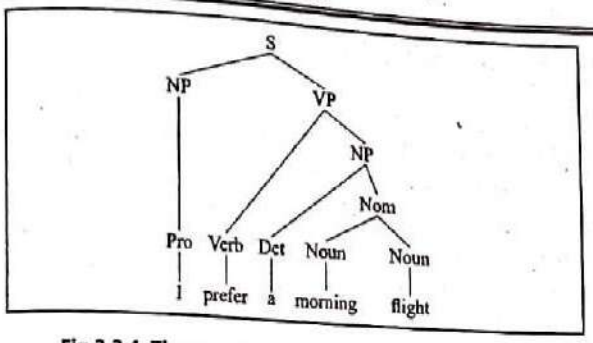


Fig.3.3.4: The parse tree for 'I prefer a morning flight'

Context free grammars consist of

- **Terminals** -words
- **Non-terminals** - constituents in a language such as noun phrases, verb phrases and sentences.
- **Rules** - are equations that consist of a single non-terminal on the left and any number of terminals and non-terminals on the right.

$$G = \{T, N, S, R\}$$

  - *T* is set of terminals (lexicon)
  - *N* is set of non-terminals For NLP, we usually distinguish out a set $P \subset N$ of *preterminals* which always rewrite as terminals.
  - *S* is start symbol (one of the nonterminals)
  - *R* is rules/productions of the form $X \rightarrow \gamma$, where *X* is a nonterminal and γ is a sequence of terminals and nonterminals (may be empty).
  - A grammar *G* generates a language *L*.

3.3.3 Parsing

- Parsing is the process of taking a string and a grammar and returning a (or multiple) parse tree(s) for that string. It is completely analogous to running a finite-state transducer with a tape.
- Parsing can be viewed as a search problem. So, what we can do with parsing? We search through the legal rewritings of the grammar. We want to find all structures matching an input string of words (for the moment). We can do parsing in bottom-up or top-down way. This distinction is independent of depth-first versus breadth-first; we can do either both ways.

3.3.3(A)Top-down parsing

A top-down is goal oriented. Parser starts root and then with a list of constituents to be built. It rewrites the goals in the goal list by matching one against the LHS of the grammar rules, and expanding it with the RHS in attempting to match the sentence to be derived. If a goal can be rewritten in several ways, then there is a choice of which rule to apply (search problem). For Top-down parsing we can use depth-first or breadth-first search, and goal ordering.

Top-down parsing example (Breadth-first) :

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a \mid the$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid man$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid read$
$NP \rightarrow Det\ NOM$	$Aux \rightarrow does$
$NOM \rightarrow Noun$	
$NOM \rightarrow Noun\ NOM$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ VP$	

Fig. 3.3.5: *L0* Grammar rule

Example sentence: Book that flight.

From Fig. 3.3.5 we can understand that *S* is start symbol of tree, which will be expand for *NP VP*, *Aux NP VP* and *VP*. Further as we expand using breadth first search shown in Fig. 3.3.6 for finding sequence for "book that flight" as *Verb Determiner Nominal Noun*.

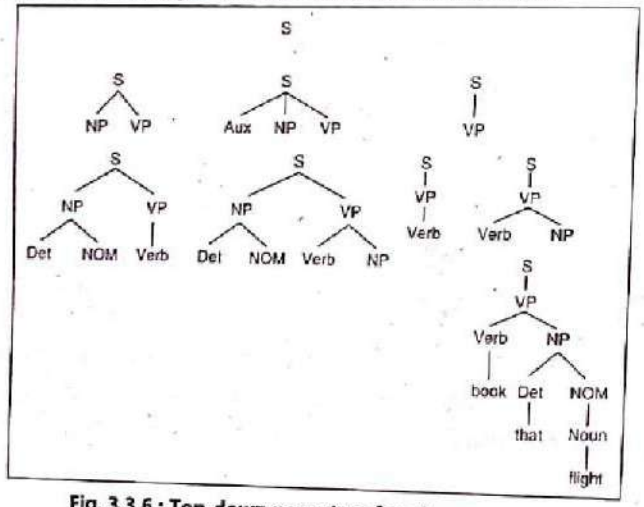


Fig. 3.3.6 : Top-down parse tree for given sentence

There are some problems with top-down parsing. In case of left recursive rules. e.g.*NP → NP PP* can be lead to infinite recursion. Top-down is sometimes waste of time for rewriting parts of speech (preterminals) with words (terminals). In practice that is always done bottom-up as lexical lookup.

3.3.3(B) Bottom-up parsing

Bottom-up parsing is data-directed. The initial goal list of a bottom-up parser is the string to be parsed. If a sequence in the goal list matches the RHS of a rule, then this sequence may be replaced by the LHS of the rule. Parsing is finished when the goal list contains just the start symbol. If the RHS of several rules match the goal list, then there is a choice of which rule to apply (search problem). Can use depth-first or breadth-first search, and goal ordering. The standard presentation is as shift-reduce parsing

- Bottom-up parsing example: "Book that flight"
- For rule we will consider same grammar rule which given in Fig.3.3.5.
- We will solve this problem with shift-reduce parsing.

Sr. No.	Tag	Description	Example
1	()	Book that flight	Shift
2	(Book)	That flight	Reduce, Verb →book,(choice #1 of 2)
3	(Verb)	That flight	Shift
4	(Verb that)	Flight	Reduce, Det → that
5	(Verb Det)	Flight	Shift
6	(Verb Det flight)		Reduce, Noun → flight
7	(Verb Det NOM)		Reduce, NP → Det, NOM
8	(Verb NP)		Reduce, VP → Verb Np
9	(Verb)		Reduce → S → V
10	(S)		SUCCESS!

Fig. 3.3.7: Bottom-up parsing using shift-reduce parser

Problems with bottom-up parsing are, parse unable to deal with empty categories like termination problem, unless rewriting empties as constituents is somehow restricted (but then it's generally incomplete). Inefficient when there is great lexical ambiguity (grammar-driven control might help here). Conversely, it is data-directed so it attempts to parse the words that are there.

The Noun Phrase

- We can view the noun phrase as revolving around a head, the central noun in the noun phrase. The syntax of English allows for both prenominal (pre-head) modifiers and post-nominal (post-head) modifiers.
- Let's understand noun phrase with parsing following example,
- Consider a sentence "All the morning flights from Denver to Tampa leaving before 10" for NP structure.

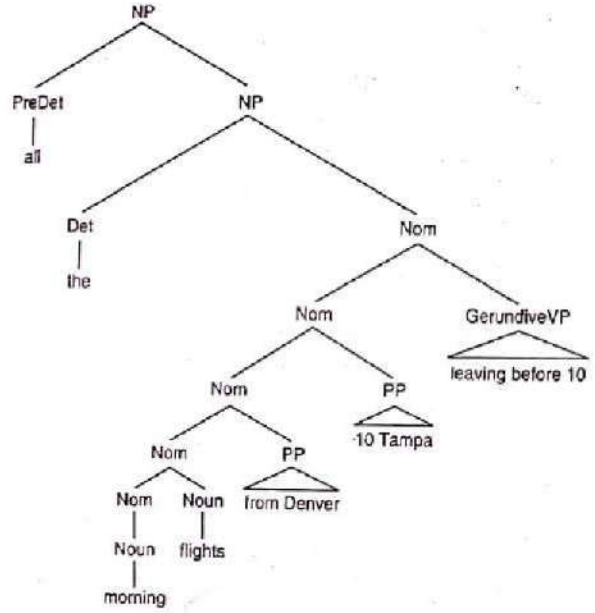


Fig. 3.3.8 : Parse tree for given example.

From the given example if we able to identify central word in the sentence then applying rule to generate parse tree will be easier. Clearly *NP* is about the word *flights*. That's the central critical noun in this *NP*. Such word is called as the **head**. We can dissect this kind of *NP* into part that comes before the head called pre-modifiers, and part that comes after it called post-modifiers.

Before the Head Noun

After discussion of some of the parts of the noun phrase; the determiner, and the use of the Nominal constituent for representing double noun phrases.

We have seen that noun phrases can begin with a determiner, as follows:

Determiners

- Noun phrases can start with determiners. Determiners can be simple lexical items like *the, this, a, an*, etc. For example, *A car* or simple possessives like *John's car* or complex recursive versions of that like *John's sister's husband's son's car.*
- There are certain circumstances under which determiners are optional in English. For example, determiners may be omitted if the noun they modify is plural :

  Show me *flights* from San Francisco to Denver on weekdays
- Word classes that appear in the NP before the determiner are called pre determiners. Many of these have to do with number or amount; a common pre determiner is all:
  - all the flights
  - all flights

Nominals

- It contains the head and any pre- and post- modifiers of the head.
- For instance, Pre-modifiers can be
- Quantifiers, cardinals, ordinals. E.g. *Three cars.*
- Adjectives and Aps. E.g. *large cars.*
- And ordering constraints. E.g. *Three large cars.*

After the Noun

A head noun can be followed by **post modifiers.** Three kinds of nominal post modifiers are very common in English:

prepositional phrases	all flights *from Cleveland.*
non-finite clauses	any flights *arriving after eleven a.m.*
relative clauses	a flight *that serves breakfast*

- Prepositional phrase post modifiers are particularly common in the ATIS corpus, since they are used to mark the origin and destination of flights.

  all flights [from Cleveland] [to Newark]

- The three most common kinds of non-finite post modifiers are the gerundive (-ing), -ed, and infinitive forms. Gerundive post modifiers are so-called because they consist of a verb phrase that begins with the gerundive (-ing) form of the verb.

  any flights (arriving after eleven a.m)

- A post nominal relative clause (more correctly a restrictive relative clause), is a clause that often begins with a relative pronoun (that and who are the most common).
  - a flight that serves breakfast
  - flights that leave in the morning

3.3.4 Coordination

- Noun phrases and other units can be **conjoined** with **conjunctions** like *and, or,* and *but.*
- For example a **coordinate** noun phrase can consist of two other noun phrases separated by a conjunction (we used brackets to mark the constituents) :
  - Please repeat [*NP* [*NP* the flights] *and* [*NP* the costs]]
  - I need to know [*NP* [*NP* the aircraft] *and* [*NP* flight number]]
  - I would like to fly from Denver stopping in [*NP* [*NP* Pittsburgh] *and* [*NP* Atlanta]]
- Here's a new rule for this:

  NP → NP and NP
- Similar conjunction rules can be built for VP and S conjunction:

  VP →VP and VP

  S → S and S

3.3.5 Agreement

- Most verbs in English can appear in two forms in the present tense: the form used for third-person, singular subjects (*the flight does*), and the form used for all other kinds of subjects (*all the flights do, I do*).The third-person-singular (*3sg* form usually has a final *-s* where the non-3sg form does not).
  - You [*VP* [*V* said [*S* there were two flights that were the cheapest]]]
  - Do [*NP* any flights] stop in Chicago?
  - Do [*NP* all of these flights] offer first class service?
  - Do [*NP* I] get dinner on this flight?

- Agreement phenomenon occurs whenever there is a verb that has some noun acting as its subject.
- Note that sentences in which the subject does not agree with the verb are ungrammatical:
  - What flights *leave* in the morning ?
  - What flight *leaves* from Pittsburgh ?
- To handle these agreement phenomena, we can modify grammar with multiple set of rules, one rule set for *3sg* subjects, and one for non-*3sg* subjects.
- For example, the rule that handled these yes-no-questions
  1. Does [*NP* you] have a flight from Boston to Forth Worth?
  2. Do [*NP* this flight] stop in Dallas?)
- Used to look like this:

  S → Aux NP VP
- We could replace this with two rules of the following form:

  S → 3sgAux 3sgNP VP

  S → Non3sgAux Non3sgNP VP
- We could then add rules for the lexicon like these:

  3sgAux → does | has| can |...

  Non3sgAux → do | have | can |...

3.3.6 The Verb Phrase and Subcategorization

- English Verb Phrase consist of a head verb along with 0 or more following constituents which we'll call ***arguments***. In the simple rules we have built so far, these other constituents include NP's and PP's and combinations of the two:

VP → Verb	disappear
VP → VerbNP	prefer a morning flight
VP → Verb NP PP	leave Boston in the morning
VP → Verb PP	leaving on Thursday

- But even though there are many valid VP rules in English, not all verbs are allowed to participate in all those VP rules. We can subcategorize the verbs in a language according to the sets of VP rules that they participate in. This is a modern take on the traditional notion of transitive/intransitive.

Modern grammars may have 100s or such classes. Some subcategorization examples are,

- Sneeze: John sneezed
- Find: Please find [a flight to NY]$_{NP}$
- Give: Give [me]$_{NP}$[a cheaper fare]$_{NP}$
- Help: Can you help [me]$_{NP}$[with a flight]$_{PP}$
- Prefer: I prefer [to leave earlier]$_{TO-VP}$
- Told: I was told [United has a flight]$_{S}$

3.4 Sequence Labeling

- The goal of sequence labeling is to assign tags to words, or more generally, to assign discrete labels to discrete elements in a sequence. The task of assigning label sequences to a set of observation sequences arises in many fields, including bioinformatics, computational linguistics and speech recognition. For example, consider the natural language processing task of labeling the words in a sentence with their corresponding part-of-speech (POS) tags. In this task, each word is labeled with a tag indicating its appropriate part of speech, resulting in annotated text, such as :

*[PRP **He**] [VBZ **reckons**] [DT **the**] [J] **current**] [NN **account**] [NN **deficit**] [MD **will**] [VB **narrow**] [TO **to**] [RB **only**] [# #] [CD 1.8] [CD billion] [IN in] [NNP September] [. .]*

we'll focus on the classic problem of **part-of-speech tagging**, which requires tagging each word by its grammatical category. Coarse-grained grammatical categories include NOUNs, which describe things, properties, or ideas, and VERBs, which describe actions and events.

- Consider a simple input :

  They can fish.

- A dictionary of coarse-grained part-of-speech tags might include NOUN as the only valid tag for they, but both NOUN and VERB as potential tags for can and fish. A accurate sequence labeling algorithm should select the verb tag for both can and fish, but it should select noun for the same two words in the phrase can of fish.

3.4.1 Sequence Labeling as Classification

- One way to solve a tagging problem is to turn it into a classification problem. Let f((w,m), y) indicate the feature function for tag y at position m in the sequence w = (w1,w2, ... ;wM).

- A simple tagging model would have a single base feature, the word itself :
  - f((w = they can fish, m = 1),N) =(they, N)
  - f((w = they can fish, m = 2), V) =(can, V)
  - f((w = they can fish, m = 3), V) =(fish, V)
- Here the feature function takes three arguments as input: the sentence to be tagged (e.g., they can fish), the proposed tag (e.g., N or V), and the index (m) of the token to which this tag is applied.
- This simple feature function then returns a single feature: a tuple including the word to be tagged and the tag that has been proposed. If the vocabulary size is V and the number of tags is K, then there are V x K features. Each of these features must be assigned a weight. These weights can be learned from a labeled dataset using a classification algorithm such as perceptron, but this isn't necessary in this case: it would be equivalent to define the classification weights directly, with $\Theta_{w,y} = 1$ for the tag y most frequently associated with word w, and $\Theta_{w,y} = 0$ for all other tags.
- However, it is easy to see that this simple classification approach cannot correctly tag both they can fish and can of fish, because can and fish are grammatically ambiguous. To handle both of these cases, the tagger must rely on context, such as the surrounding words.
- We can build context into the feature set by incorporating the surrounding words as additional features :

$$f((w = \textit{they can fish}, 1), N) = \{(w_m = \text{they}, y_m = N); (w_{m-1} = \_, y_m = N), (w_{m+1} = \text{can}, y_m = N)\}$$

$$f((w = \textit{they can fish}, 2), V) = \{(w_m = \text{can}, y_m = V); (w_{m-1} = \text{they}, y_m = V), (w_{m+1} = \text{fish}, y_m = V)\}$$

$$f((w = \textit{they can fish}, 3), V) = \{(w_m = \text{fish}, y_m = V); (w_{m-1} = \text{can}, y_m = V), (w_{m+1} = \_, y_m = V)\}.$$

- These features contain enough information that a tagger should be able to choose the right tag for the word fish: words that come after can are likely to be verbs, so the feature (wm-1 = can; ym = V) should have a large positive weight.

3.4.2 Hidden Markov Model

- As with the naïve Bayes classifier, instead of thinking directly about finding the most probable tagging of the sequence, we think about how the sequence might have come to be. Let

  w = w1 wn be a sequence of words and let t = t1 ···tn be a sequence of tags.

$$\arg\max P(t \mid w) = \arg\max P(t,w)$$

$$P(t,w) = P(t)\,P(w|t).$$

- The first term, P(t), can be described using a weighted FSA, just like a language model except over tags instead of words. For example, a bigram model :

$$P(t) = p(t_1 \mid <s>) \times \left(\prod_{i=2}^{n} p(t_i \mid t_{i-1}) \right) \times p(</s> \mid t_n)$$

The second term is even easier :

$$P(w \mid t) = \prod_{i=1}^{a} p(w_i \mid t_i)$$

- This is a hidden Markov model (HMM). If we're given labeled data, like

I	saw	her	duck
PRP	VBD	PRP$	NN

then the HMM is easy to train. But what we don't know how to do is classify (or decode): given w (word), what's the most probable t (tag)? Below, we'll see how to do that, not just for HMMs but for a much broader class of models.

Decoding

Suppose that our HMM has the following parameters:

P(t' | t)

t'	<s>	NN	PRP	PRP$	VB/VBD/VBP
NN	0	0.1	0	1	0.1
PRP	0.8	0	0	0	0.4
PRP$	0.2	0	0	0	0.3
VB	0	0.1	0.2	0	0
VBD	0	0.5	0.3	0	0
VBP	0	0	0.3	0	0
<S>	0	0.3	0.2	0	0.2

p(I | PRP) = 0.5

p(her| PRP) = 0.5

p(her| PRP$) = 1

p(saw | VBD) = 1

p(saw | VBP) = 1

p(duck | NN) = 1

p(duck | VB) = 1

3.4.3 Conditional Random Fields

- The **conditional random field** (CRF) is a conditional probabilistic model for sequence labeling; just as structured perceptron is built on the perceptron classifier, conditional random fields are built on the logistic regression classifier.
- Conditional random fields (CRFs) are a probabilistic framework for labeling and segmenting sequential data, based on the conditional approach described in hidden markov model.
- A CRF is a form of undirected graphical model that defines a single log-linear distribution over label sequences given a particular observation sequence.
- The primary advantage of CRFs over hidden Markov models is their conditional nature, resulting in the relaxation of the independence assumptions required by HMMs in order to ensure tractable inference.
- Additionally, CRFs avoid the label bias problem, a weakness exhibited by maximum entropy Markov models (MEMMs) and other conditional Markov models based on directed graphical models.
- CRFs outperform both MEMMs and HMMs on a number of real-world sequence labeling tasks.
- The graphical structure of a conditional random field may be used to factorize the joint distribution over elements Yv of Y into a normalized product of strictly positive, real-valued potential functions, derived from the notion of conditional independence.
- The probability of a particular label sequence *y* given observation sequence *x* to be a normalized product of potential functions, each of the form

$$\exp\left(\sum_j \lambda_j t_j(y_{i-1}, x, i) + \sum_k \mu_k s_k(y_i, x, i) \right)$$

where ***tj (yi – 1, yi, x, i)*** is a transition feature function of the entire observation sequence and the labels at positions i and i-1 in the label sequence;

sk(yl, x, l) is a state feature function of the label at position i and the observation sequence;

And λJ and μk are parameters to be estimated from training data.

- When defining feature functions, we construct a set of real-valued features b(x, i) of the observation to expresses some characteristic of the empirical distribution of the training data that should also hold of the model distribution.

An example of such a feature is,

$$b(x, i) = \begin{cases} 1 & \text{if the observation at position i is the word "September"} \\ 0 & \text{otherwise} \end{cases}$$

- Each feature function takes on the value of one of these real-valued observation features b(x, i) if the current state (in the case of a state function) or previous and current states (in the case of a transition function) take on particular values.

3.4.4 Maximum Entropy

- The form of a CRF, as given in, is heavily motivated by the principle of maximum entropy - a framework for estimating probability distributions from a set of training data.
- Entropy of a probability distribution is a measure of uncertainty and is maximized when the distribution in question is as uniform as possible.
- The principle of maximum entropy asserts that the only probability distribution that can justifiably be constructed from incomplete information, such as finite training data, is that which has maximum entropy subject to a set of constraints representing the information available.

Review Questions

Q.1 What is syntax analysis?

Q.2 What are the types of Part Of Speech? Explain.

Q.3 Explain methods for Part Of Sppech(POS) tagging.

Q.4 Write a note on CFG.

Q.5 Explain the concept of Parsing.

Q.6 What is sequence labeling.

□□□

4 Module - 4 SEMANTIC ANALYSIS

Syllabus

Lexical Semantics, Attachment for fragment of English- sentences, noun phrases, Verb phrases, prepositional phrases, Relations among lexemes & their senses –Homonymy, Polysemy, Synonymy, Hyponymy, WordNet, Robust Word Sense Disambiguation (WSD), Dictionary based approach

TOPICS

4.1 Lexical Semantics

4.1.1 Semantic Analysis

1. Semantic analysis is the process of extracting meaning from text.
2. It permits computers to know and interpret sentences, paragraphs, or whole documents.
3. It is done by analysing their grammatical structure, and distinguishing relationships between individual words in a specific context.
4. It's an important sub-task of Natural Language Processing (NLP)
5. It drives machine learning tools like chatbots, search engines, and text analysis.
6. Semantic analysis-driven tools can facilitate industries to automatically extract meaningful information from unstructured data, like emails, comments, and feedback.
7. Semantic analysis can be divided into two parts. They are:
   a) **Studying meaning of individual word:** It is the primary part of the semantic analysis during which the study of the meaning of single words is done. This part is named lexical semantics.
   b) **Studying the combination of individual words:** In the secondary part, the single words are combined to produce meaning in sentences.
8. Example:"Your customer service is a joke! I've been on hold for 30 minutes and counting!"
   a) Any human can understand that a customer is frustrated because a customer service agent is taking too long to respond.
   b) However, machines first need to be trained to make sense of human language and understand the context in which words are used
   c) Otherwise, they might misinterpret the word "joke" as positive.

4.1.2 Lexical Semantics

1. Lexical semantics plays a crucial role in semantic analysis, allowing computers to know relationships between words, phrasal verbs, etc.
2. In Lexical Semantics words, sub-words, etc. are called lexical items.
3. In simple terms, lexical semantics is the relationship between lexical items, meaning of sentences and syntax of sentence.

4. Lexical semantics includes the following two mainpoints :
   a) Classification and Decomposition of lexical items
   b) Differences and similarities between numerous lexical semantic structures

4.1.3 Elements of Lexical Semantic Analysis

Followings are some important elements of lexical semantic analysis –

1. **Hyponyms :** They are specific lexical items of a general lexical item (hypernym) e.g., apple is a hyponym of fruit (hypernym).
2. **Meronomy :** It is a logical arrangement of text and words that represent a part of or member of something e.g., a segment of an apple
3. **Polysemy :** It a relationship between the meanings of words or phrases, although slightly different, they share a common basic meaning e.g. I read a book, and I wrote a book
4. **Synonyms :** They are words that have the same sense or mostly the same meaning as another, e.g., sad, unhappy, depressed, etc.
5. **Antonyms :** They words that have close to opposite meanings e.g., good, bad
6. **Homonyms :** They are two words that sound the same and are spelled alike but have a different meaning e.g., right (correct), right (turn)

4.2 Attachment for Fragment of English

4.2.1 Semantic Attachment

1. Semantic attachment is the process of making semantics of a sentence by attaching pieces of semantics to the syntax tree.
2. It helps in creating semantic representation of a sentence.
3. Such a representation consists of a set of predications: relations between objects and events.
4. Example,

   "Ramruns"

   It can be represented by this set of predications:

   ∃e1, o1 isa(e1, Runs) ∧ subject(e1, o1) ∧ name(o1, "Ram")
5. The task is to get from the syntax tree to the semantic representation
6. There are 3 ways that to get from the syntax tree to the semantic representation. They are:

a) Semantic specialists

- Their job is to create complex semantic structures that represent the sentence's semantics.
- The specialist for the NP node uses the node of the proper noun below it to create its semantics.
- It may also inspect other nodes of the tree and check if their meaning is compatible.
- Specialists may use domain-specific procedures to provide the correct meaning.
- Their form is procedural rather than declarative and this makes them hard to extend.
- For example,

```
S
+--NP
|  +-- proper noun: Ram
|
+--VP
   +-- verb: runs
```

b) Lambda Calculus

- It extends First Order Predicate Calculus (FOPC) with function application.
- The predicate calculus includes unary, binary, and n-ary predicates.
- For example,

1. **"Ram runs"**

```
S {Sem = VP.Sem(NP.sem) = λe isa(e, Run) ∧ subject(e, <name(y, "Ram")>)}
+--NP {Sem = propernoun.Sem = λy name(y, "Ram")}
|  +-- propernoun: Ram {Sem = λy name(y, "Ram")}
|
+--VP {Sem = verb.Sem = λx,e isa(e, Run) ∧ subject(e, x)}
   +-- verb: Runs {Sem = λx,e isa(e, Run) ∧ subject(e, x)}
```

2. The verb node gets its semantics from the dictionary entry of run.
3. The proper noun gets its semantics from a procedure that creates a predication name (y, "Ram"), for the word Ram.

4. Both the VP and the NP node gets their semantics from their child node.
5. Only the S node performs some actual composition by applying the semantics of VP, like a function, to the semantics of the NP.
6. Sem (the semantics of S) applies the function VP.Sem to the argument NP.Sem. This results in a new function with only one unbound argument.

- In this technique every node exports a lambda function that has zero or more (unbound) arguments.
- This lambda function is then used as an argument for the lambda function of the above node.
- The order of the exposed arguments is very important.
- It is a very simple technique for the given example.
- But as soon as more complex sentences are used, the technique requires special constructs and becomes less intuitive quickly.
- This is mostly due to the tight handling of variables.

c) Feature unification

- Unification is a (partial) operation on feature structures.
- Intuitively, it is the operation of combining two feature structures such that the new feature structure contains all the information of the original two, and nothing more.
- For example,

1. let F1 be the feature structure

$$\begin{bmatrix} \text{CAT} & \text{np} \\ \text{AGREE} & [\text{NUMBER singular}] \end{bmatrix}$$

2. and let F2 be the feature structure

$$\begin{bmatrix} \text{CAT} & \text{np} \\ \text{AGREE} & [\text{PERSON third}] \end{bmatrix}$$

3. Then, F1L1F2, the unification of these two feature structures is:

$$\begin{bmatrix} \text{CAT} & \text{np} \\ \text{AGREE} & \begin{bmatrix} \text{NUMBER} & \text{singular} \\ \text{PERSON} & \text{third} \end{bmatrix} \end{bmatrix}$$

4. Clearly F1L1F2 contains all the information that is in F1 and F2 and it doesn't contain any other information.

5. The unification is not guaranteed to return a result. For example, let F3 be the feature structure

[CAT NP]

6. and let F4 be the feature structure

[CAT vp]

7. Then F3 ⊔ F4 does not exist.
8. There is no feature structure that contains all the information in F3 and F4, because the information in these two feature structures is contradictory.
9. So, the value of this unification is undefined.

4.2.2 Strategy for Semantic Attachments

1. Create complex, lambda expressions with lexical items
2. Introduce quantifiers, predicates, terms
3. Percolate up semantics from child if non-branching
4. Apply semantics of one child to other through lambda
5. Combine elements, but don't introduce new
6. For example,

a) Every flight arrived.

(S (NP (Det every) (Nom (Noun flight)))(VP (V arrived)))

b) Target representation:

$\forall x Flight(x) \Rightarrow \exists e Arrived(e) \wedge ArrivedThing(e, x)$

c) Defining Representation

i) Idea: Every flight = $\forall x Flight(x)$

Good enough ?

No : roughly 'everything is a flight'

Saying something about all flights – nuclear scope

ii) Solution : Dummy predicate = $\forall x Flight(x) \Rightarrow Q(x)$

Good enough ?

No : no way to get Q(x) from elsewhere in sentence

iii) Solution: Lambda

$\lambda Q.\forall x Flight(x) \Rightarrow Q(x)$

iv) Creating Attachments

Noun → flight - {λx.Flight(x)}

Nom → Noun - { Noun.sem }

Det → Every - { $\lambda P.\lambda Q.\forall x P(x) \Rightarrow Q(x)$ }

NP → Det Nom - { Det.sem(Nom.sem) }

$\lambda P.\lambda Q.\forall x P(x) \Rightarrow Q(x)(\lambda x.Flight(x))$

$\lambda P.\lambda Q.\forall x P(x) \Rightarrow Q(x)(\lambda y.Flight(y))$

$\lambda Q.\forall x \lambda y.Flight(y)(x) \Rightarrow Q(x)$

$\lambda Q.\forall x Flight(x) \Rightarrow Q(x)$

v) Full Representation

Verb → arrived { }

VP → Verb { Verb.sem }

S → NP VP { NP.sem(VP.sem) }

$\lambda x.\exists e Arrived(e) \wedge ArrivedThing(e, x)$

$\lambda Q.\forall x Flight(x) \Rightarrow Q(x)(\lambda y.\exists e Arrived(e) \wedge ArrivedThing(e, y))$

$\forall x Flight(x) \Rightarrow \lambda y.\exists e Arrived(e) \wedge ArrivedThing(e, y)(x)$

$\forall x Flight(x) \Rightarrow \exists e Arrived(e) \wedge ArrivedThing(e, x)$

4.2.3 Attachments for a Fragment of English

1. Sentences

a) Flight 707 serves dinner.

S → NP VP {DCL(VP.sem(NP.sem))}

Serve dinner.

S → VP {IMP(VP.sem(DummyYou))}

IMP($\exists e Serving(e) \wedge Server(e, DummyYou) \wedge Served(e, Dinner)$)

b) Imperatives can be viewed as a kind of speech act.

Does Flight 313 serve dinner ?

S → Aux NP VP {YNQ(VP.sem(NP.sem))}

YNQ(∃eServing(e) ∧Server(e, Flt313) ∧Served(e, Dinner)

c) Which flights serve dinner?

S → WhWord NP VP {WHQ(NP.sem.var,VP.sem(NP.sem))}

WHQ(x, ∃e, x Isa(e, Serving) ∧Server(e, x) ∧Served(e, Dinner) ∧ Isa(x, Flight))

2. Compound Nominals

a) The meaning representations for NPs can be either normal FOPC terms or complex terms.

b) E.g., Flight schedule

c) E.g., Summer flight schedule

Nominal → Noun

Nominal → Nominal Noun {λx Nominal.sem(x) ∧ NN(Noun.sem,x)}

λx Isa(x, Schedule)∧NN(x, Flight)

λx Isa(x, Schedule)∧NN(x, Flight)∧NN(x, Summer)

3. Adjective Phrases

a) E.g., I don't mind a cheap restaurant.

b) E.g., This restaurant is cheap.

c) For pre-nominal case, an obvious and often incorrect proposal is:

Nominal → Adj Nominal

{λx Nominal.sem(x)∧Isa(x, Adj.sem)}

Adj → cheap {Cheap}

λx Isa(x, Restaurant)∧Isa(x, Cheap) intersective semantics

d) The best approach is to simply note the status of a special kind of modification relation

e) Assume that some further procedure with access to additional relevant knowledge can replace this vague relation with an appropriate representation.

f) Nominal → Adj Nominal

{λx Nominal.sem(x)∧AM(x, Adj.sem)}

Adj → cheap {Cheap}

λx Isa(x, Restaurant)∧AM(x, Cheap)

4. VPs: Infinite VPs(Verbal Noun)

a) Averbphraseis a syntactic unit consisting of an auxiliary (helping) verb preceding the main verb.

b) It often contains a head verb, complements, objects, and modifiers as its dependents.

c) In general, the λ-expression attached to the verb is simply applied to the semantic attachments of the verb's arguments.

d) However, some special cases, for example, infinite VP

"I told Harry to go to Maharani."

e) The meaning representation could be:

∃e, f, x Isa(e, Telling)∧Isa(f, Going)

∧Teller(e, Speaker)∧Tellee(e, Harry)∧ToldThing(e, f)

∧Goer(f, Harry)∧Destination(f, x)

f) Two things to note:

i) It consists of two events, and

ii) One of the participants, Harry, plays a role in both of the two events.

g) A way to represent the semantic attachment of the verb, tell

λx, yλz∃e Isa(e, Telling)∧Teller(e, z)∧Tell(e, x)∧ToldThing(e,y)

h) Providing three semantic roles:

i) a person doing the telling,

ii) a recipient of the telling, and

iii) the proposition being conveyed

i) Problem :

Harry is not available when the Going event is created within the infinite verb phrase.

j) Solution :

VP → Verb NP VPto {Verb.sem(NP.sem, VPto.sem)}

VPto → to VP Verb NP {VP.sem}

Verb → tell {λx, y λz

∃e, y.variable Isa(e, Telling)∧ Teller(e, z)∧Tellee(e,x)

∧ToldThing(e, y.variable)∧y(x)}

k) The λ -variable x plays the role of the Teller of the telling and the argument to the semantics of the infinitive, which is now contained as a λ -expression in the variable y.

l) The expression y(x) represents a λ-reduction that inserts Harry into the Going event as the Goer.

m) The notation y.variable is analogous to the notation used for complex-terms variables, and gives us access to the event variable representing Going event within the infinitive's meaning representation.

5. Noun Phrases

1. A noun phrase is a group of two or more words accompanied by a noun that includes modifiers e.g., 'the,' 'a,' 'of them,' 'with him'.
2. A noun phrase plays the role of a noun. In a noun phrase, the modifiers can come before or after the noun.

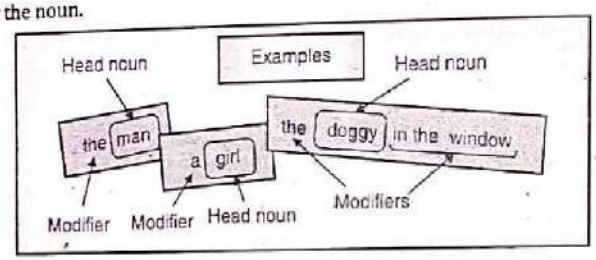


Fig. 4.2.1 : Noun Phrases

Examples of Noun Phrases

1. Generally, nouns mostly always feature in noun phrases. It is rare to find a noun functioning by its ownthat is without any modifiers in a sentence.
   a) e.g., Love is a beautiful feeling.
2. Some examples of common traditional noun phrases.
   a) **People**: the **captain**, my **sister**, funny **Sham**, the **judge** with the big nose
   b) **Animals**: that **lion**, one **cat**, a **whale**
   c) **Places**: the **building** in the corner, inner **Delhi**, dirty **office**
3. Here, every noun phrase consists of a head noun and at least one modifier.

The Function of Noun Phrases

1. A noun phrase can function as a subject, an object, or a complement within a sentence.

2. For example,
   a) **Running** in the morning relaxes me.
   Here, the noun phrase is the subject of the verb "relaxes."
   b) I know the back **alleys**.
   Here, the noun phrase is the direct object of the verb "know."
   c) He was the **devil** in disguise.
   d) Here, the noun phrase is a subject complement following the linking verb "was."

NP: Genitive NPs(NP-NOUN Phrase)

a) (Ex.) Atlanta's airport

b) (Ex.) Maharani's menu

NP → ComplexDet Nominal

{<∃xNominal.sem(x)∧GN(x, ComplexDet.sem)>}

ComplexDet → NP's {NP.sem}

<∃x Isa(x, Airport) ∧GN(x, Atlanta)>

6. Prepositional Phrases

a) At an abstract level, PPs serve two functions:
   i) They assert binary relations between their heads and the constituents to which they attached, and
   ii) They signal arguments to constituents that have an argument structure.

b) We will consider three places in the grammar where PP serve these roles:

i) Modifiers of NPs

1) E.g., A restaurant on Pearl

$\exists x\ Isa\ (x, Restaurant) \wedge On(x, Pearl)$

$NP \rightarrow Det\ Nominal$

$Nominal \rightarrow Nominal\ PP\ \{\lambda z\ Nominal.sem(z) \wedge PP.sem(z)\}$

$PP \rightarrow P\ NP\ \{P.sem(NP.sem)\}$

$P \rightarrow on\ \{\lambda y\ \lambda x\ On(x,y)\}$

ii) PP: VP Modifier

1) (Ex.) ate dinner in a hurry

$VP \rightarrow VP\ PP$

2) The meaning representation of ate dinner

$\lambda x \exists e\ Isa(e, Eating) \wedge Eater(e, x) \wedge Eaten(e, Dinner)$

3) The representation of the PP

$\lambda x\ In(x, <\exists h\ Hurry(h)>)$

4) The correct representation of the modified VP should contain the conjunction of the two

5) With the Eating event variable filling the first argument slot of the In expression.

$VP \rightarrow VP\ PP\ \{\lambda y\ VP.sem(y) \wedge PP.sem(VP.sem.variable)\}$

6) The result of application

$\lambda y \exists e\ Isa(e, Eating) \wedge Eater(e, y) \wedge Eaten(e, Dinner) \wedge In(e, <\exists h\ Hurry(h)>)$

4.3 Relations Among Lexemes and Their Senses

4.3.1 Senses

1. One word can have multiple meanings.
2. The unit of meaning is a sense.
3. A word sense is a discrete representation of oneaspect of the meaning of a word.
4. For example,
   a) Instead, a bank can hold the investments in a custodial account inthe client's name.
   b) But as agriculture burgeons on the east bank, the river will shrinkeven more.
   c) Here bank here has two senses one is bank a place where money is kept and second bank is the side of the river.

4.3.2 Relations Between Words/Senses

1. Homonymy

- They are lexemes that share a form, but unrelatedmeanings.
- Homonyms are words that have a same syntax or same spelling or same form but their meaning are different and unrelated to each other.
- Two or more words become homonyms if they either sound the same (homophones), have the same spelling (homographs), or if they both homophones and homographs, but do not have related meanings.

- Examples:
  i) bat (wooden stick thing) vs bat (flying scary mammal)
  ii) bank (financial institution) vs bank (riverside)

2. Polysemy

- Polysemy refers to words or phrases with different, but related meanings.
- A word becomes polysemous if it can be used to express different meanings.
- The difference between these meanings can be obvious or subtle.
- It is sometimes difficult to determine whether a word is polysemous or not because the relations between words can be vague and unclear.
- But, examining the origins of the words can help to decide whether a word is polysemic or homonymous.
- The following sentences contain some examples of polysemy.
  i) He drank a glass of milk.
  i) He forgot to milk the cow.
  iii) The enraged actor sued the newspaper.
  iv) He read the newspaper.

Difference between Polysemy and Homonymy

Sr. No.	Polysemy	Homonymy
1.	In Polysemy two words or phrase may havemany possible meanings.	In Homonymy two unrelated words that look or sound the same and have different meanings.
2.	They have different, but related meanings	They have completely different meanings
3.	They have related word origins	They have different origins
4.	Polysemous words are entered under one entry in dictionaries	Homonyms are entered separately in dictionaries
5.	Polysemous words can be understood if one knows the meaning of one word	Homonyms words meaning cannot be guessed as the words have unrelated different meanings

3. Synonymy

- A synonym is a word or phrase that means exactly the same as another word or phrase, in the same language.
- In other words, synonyms are words with similar meanings.
- For instance, words like delicious, yummy, succulent are synonyms of the adjective tasty.
- Similarly, verbs like commence, initiate, and begin are synonyms of the verb start.
- However, some synonyms do not have exactly the same meaning – there may be minute differences.
- Sometimes a word can be synonymous with another in one context or usage, but not in another.
- Examples
  - i) Beautiful-Gorgeous
  - ii) Purchase-Buy
  - iii) Use-Employ
  - iv) Rich-Wealthy
  - v) Mistake - Error

4. Antonymy

- Antonyms are words that have opposite or contrasting meanings.
- For example, the antonym of hot is cold; similarly, the antonym of day is night.
- Antonyms are actually the opposite of synonyms.
- Furthermore, there are three types of antonyms as gradable, complementary, and relational antonyms.
- Gradable antonyms are pairs of words with opposite meanings that lie on a continuous spectrum.
- For example, if we take age as a continuous spectrum, young and old are two ends of the spectrum.
- Complementary antonyms are pairs of words with opposite meanings that do not lie on a continuous spectrum.
- For example, interior: exterior, true: false, and inhale: exhale
- Relational antonyms are pairs of words that refer to a relationship from opposite points of view.
- For example, doctor: patient, husband: wife, teacher: student, sister: brother.

5. Hypernymy and Hyponymy

- Hyponym is the sense which is a subclass of another sense
  - i) car is a hyponym of vehicle
  - ii) dog is a hyponym of animal
  - iii) mango is a hyponym of fruit
- Hypernym is the sense which is a superclass
  - i) vehicle is a hypernym of car
  - ii) animal is a hypernym of dog
  - iii) fruit is a hypernym of mango
- In simpler terms, a hyponym is in a type-of relationship with its hypernym.
- Hypernyms and hyponyms are asymmetric.
- Hyponymy can be tested by substituting X and Y in the sentence "X is a kind of Y" and determining if it makes sense.
- For example, "A screwdriver is a kind of tool" makes sense, but not "A tool is a kind of screwdriver".
- Strictly speaking, the meaning relation between hyponyms and hypernyms applies to lexical items of the same word class (or parts of speech), and holds between senses rather than words.
- Hyponymy is a transitive relation, if X is a hyponym of Y, and Y is a hyponym of Z, then X is a hyponym of Z.
- For example, violet is a hyponym of purple and purple is a hyponym of color; therefore, violet is a hyponym of color.
- A word can be both a hypernym and a hyponym: for example, purple is a hyponym of color but itself is a hypernym of the broad spectrum of shades of purple between the range of crimson and violet.
- The hierarchical structure of semantic fields can be mostly seen in hyponymy.
- They could be observed from top to bottom, where the higher level is more general and the lower level is more specific.
- Hyponymy is the most frequently encoded relation among synsets used in lexical databases such as WordNet.
- These semantic relations can also be used to compare semantic similarity by judging the distance between two synsets and to analyse anaphora.

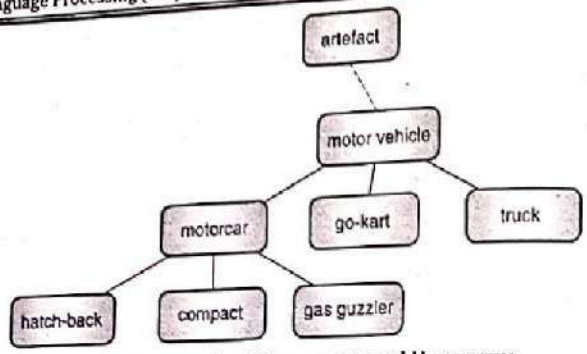


Fig 4.3.1 : Example of Hypernymy and Hyponymy

6. Meronymy

- A meronym is a word that represents a constituent part or a member of something.
- For example, Guava is a meronym of Guava tree (sometimes written as guava<guava tree).
- This part-to-whole relationship is named as meronymy.
- Meronymy is not only a single relation but a bunch of different part-to-whole relationships.
- It is also expressed in terms of first-order logic.
- It can also be considered as a partial order.
- In knowledge representation languages, meronymy is often represented as "part-of".

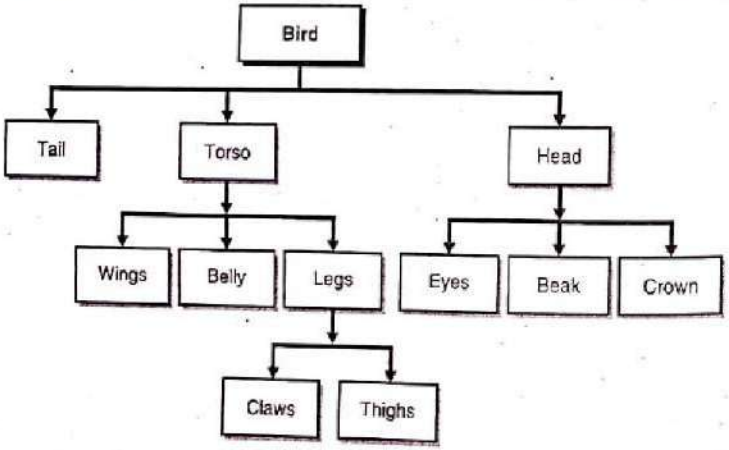


Fig. 4.3.2: Example of meronymy

4.4 WordNet

4.4.1 WordNet and Synsets

- Wordnet is a big collection of words from the English language that are related to each other and are grouped in some way.
- It is also called as a lexical database.
- In other words, WordNet is a database of English words that are connected together by their semantic relationships.
- It is like a superset dictionary with a graph structure.
- WordNet groups nouns, verbs, adjectives, etc. which are similar and the groups are called synsets or synonyms.
- In a wordnet a group of synsets may belong to some other synset.
- For example, the synsets "Stones" and "cement" belong to the synset "Building Materials" or the synset "Stones" also belongs to another synset called "stonework".
- In the example given, stones and cement are called hyponyms of synset building materials and also the synsets building materials and stonework are called synonyms.
- Every member of a synset denotes the same concept but not all synset members are interchangeable in context.
- The membership of words in multiple synsets or concepts mirrors polysemy or multiplicity of meaning.
- There are three principles the synset construction process must adhere to:

a) Minimality

i) This principle determines on capturing those minimal set of the words in the synset which especially identifies the concept.

ii) For example, {family, house} uniquely identifies a concept e.g., "she is from the house of the Classical Singers of Hyderabad".

b) Coverage

i) This principle'smain aim is completion of the synset, that is capturing all those words that represents the concept expressed by the synset.

ii) In the synset, the words should be ordered according to their frequency in the collection.

c) Replaceability

i) Replaceability dictates that the most common words in the synset, that is words towards the beginning of the synset should be able to replace one another.

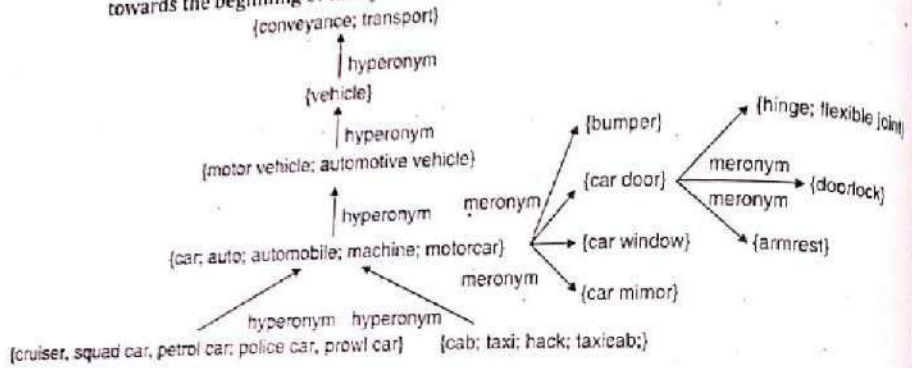


Fig. 4.4.1 : Wordnet

- WordNet seeminglyis similar to a dictionary, in that it groups words based on their meanings. However, there are some important differences.
  a) WordNet interlinks not only word forms or strings of letters but particular senses of words. As a result, words that are found near one another in the network are semantically clarified.
  b) WordNet marks the semantic relations among words, whereas the groupings of words in a dictionary does not follow any particular pattern other than meaning similarity.

4.5 Word-sense Disambiguation(WSD)

4.5.1 Word-sense Disambiguation(WSD)

- Word-sense disambiguation (WSD) is a well-known problem in NLP.
- WSD is used in identifying what the sense of a word means in a sentence when the word has multiple meanings.
- When a single word has multiple meaning, then for the machine it is difficult to identify the correct meaning and to solve this challenging issue we can use the rule-based system or machine learning techniques.
- WSD is a natural classification problem: Given a word and its possible senses, as defined by a dictionary, classify an occurrence of the word in context into one or more of its sense classes.

- The features of the context such as neighbouring words provide the evidence for classification.
- A famous example is to determine the sense of pen in the following passage.

"Little John was looking for his toy box. Finally, he found it. The box was in the pen. John was very happy."

WordNet lists five senses for the word pen:

- pen — a writing implement with a point from which ink flows.
- pen — an enclosure for confining livestock.
- playpen, pen — a portable enclosure in which babies may be left to play.
- penitentiary, pen — a correctional institution for those convicted of major crimes.
- pen — female swan.

- There are four conventional approaches to WSD :
  1. **Dictionary- and knowledge-based methods :** These rely primarily on dictionaries, thesauri, and lexical knowledge bases, without using any corpus evidence.
  2. **Supervised methods :** These make use of sense-annotated corpora to train from.
  3. **Semi-supervised or minimally-supervised methods :** These make use of a secondary source of knowledge such as a small annotated corpus as seed data in a bootstrapping process, or a word-aligned bilingual corpus.
  4. **Unsupervised methods :** These eschew (almost) completely external information and work directly from raw unannotated corpora. These methods are also known under the name of word sense discrimination.

4.5.2 WSD Methods

1. Dictionary- and knowledge-based methods

a. The Lesk method is dictionary-based method. It is based on the hypothesis that words used together in text are associated with one another which the relation are often observed within the definitions of the words and their senses.

b Two or more words are disambiguated by finding the pair of dictionary senses with the best word overlap in their dictionary definitions.

c. For example, when disambiguating the words in pine cone, the definitions of the acceptable senses both include the words evergreen and tree a minimum of in one dictionary.

d. An alternative to the utilization of the definitions is to think about general word-sense relatedness and to compute the semantic similarity of every pair of word senses supported a given lexical knowledgebase like WordNet.

e. Graph-based methods like spreading-activation research of the first days of AI research are applied with some success.

f. The use of selection preferences or selection restrictions also are useful.

g. For example, knowing that one typically cooks food, one can disambiguate the word bass in I am cooking bass i.e., it's not a musical instrument.

2. Supervised methods

a. Supervised methods are supported the idea that the context can provide enough evidence on its own to disambiguate words.

b. Hence, world knowledge and reasoning are deemed unnecessary.

c. Probably every machine learning algorithm going has been applied to WSD, including associated techniques like feature selection, parameter optimization, and ensemble learning.

d. Support vector machines and memory-based learning are shown to be the foremost successful approaches, to date, probably because they will deal with the high-dimensionality of the feature space.

e. However, these supervised methods are subject to a replacement knowledge acquisition bottleneck since they believe substantial amounts of manually sense-tagged corpora for training, which are laborious and expensive to make

3. Semi-supervised methods

a. The bootstrapping approach starts from a little number of seed data for every word: either manually-tagged training examples or a little number of sure-fire decision rules

b. E.g., play within the context of bass nearly always indicates the instrument.

c. The seeds are wont to train an initial classifier, using any supervised method.

d. This classifier is then used on the untagged portion of the corpus to extract a bigger training set, during which only the foremost confident classifications are included.

e. The process repeats, each new classifier being trained on a successively larger training corpus, until the entire corpus is consumed, or until a given maximum number of iterations is reached.

f. Other semi-supervised techniques use large quantities of untagged corpora to supply co-occurrence information that supplements the tagged corpora.

g. These techniques have the potential to assist within the adaptation of supervised models to different domains.

h. Also, an ambiguous word in one language is usually translated into different words during a second language counting on the sense of the word.

i. Word-aligned bilingual corpora are wont to infer cross-lingual sense distinctions, a sort of semi-supervised system.

4. Unsupervised methods

a. Unsupervised learning is that the greatest challenge for WSD researchers.

b. The underlying assumption is that similar senses occur in similar contexts, and thus senses are often induced from text by clustering word occurrences using some measure of similarity of context.

c. Then, new occurrences of the word are often classified into the closest induced clusters/senses.

d. Performance has been less than other methods, above, but comparisons are difficult since senses induced must be mapped to a known dictionary of word senses.

e. Alternatively, if a mapping to a group of dictionary senses isn't desired, cluster-based evaluations (including measures of entropy and purity) are often performed.

f. It is hoped that unsupervised learning will overcome the knowledge acquisition bottleneck because they're not hooked in to manual effort.

4.5.3 WSD Evaluation

- The evaluation of WSD systems requires a test corpus hand-annotated with the target or correct senses, and assumes that such a corpus can be constructed.
- Two main performance measures are used:
- Precision: the fraction of system assignments made that are correct
- Recall: the fraction of total word instances correctly assigned by a system
- If a system makes an assignment for every word, then precision and recall are the same, and can be called accuracy.
- This model has been extended to take into account systems that return a set of senses with weights for each occurrence.
- There are two kinds of test corpora :
  - Lexical sample : the occurrences of a small sample of target words need to be disambiguated, and

- All-words : all the words in a piece of running text need to be disambiguated.

- The latter is deemed a more realistic form of evaluation, but the corpus is more expensive to produce because human annotators have to read the definitions for each word in the sequence every time they need to make a tagging judgement, rather than once for a block of instances for the same target word.
- In order to define common evaluation datasets and procedures, public evaluation campaigns have been organized.

4.5.4 Difficulties in WSD

Followings are some difficulties faced by word sense disambiguation (WSD) :

1. **Differences between dictionaries**
   a) The major problem of WSD is to decide the meaning of the word because different contextual meaning can be very closely related.
   b) Even different dictionaries and thesauruses can provide different parts of words into meanings.
2. **Inter-judge variance**
   a. One problem of WSD is that WSD systems are generally tested by having their results of a task compared against the task of human beings.
   b. This is called the problem of inter-judge variance.
3. **Different algorithms for different applications**
   a. Another problem of WSD is that a different algorithm may be needed for different applications.
   b. For example, in machine translation, it takes the form of target word selection; and in information retrieval, a sense inventory is not required.
4. **Word-sense discreteness**

   One of the difficulty in WSD is that words cannot be easily divided into discrete sub meanings.

4.5.5 Applications of WSD

Word sense disambiguation (WSD) is applied in almost every application of language technology.

Followings are some applications of WSD :

1. **Machine Translation :**
   - Machine translation is the most common application of WSD.
   - In machine translation, Lexical choice for the words that have different translations for different senses, is done by WSD.
   - The senses in machine translation are represented as words in the target language.
2. **Text Mining and Information Extraction (IE)**
   - In most of the text mining and IE, WSD is necessary for accurate analysis of text.
   - WSD helps in flagging of the correct words.
   - For example, email intelligent system might need flagging of "http links" rather than "https links"
3. **Information Retrieval (IR)**
   - Information retrieval (IR) is defined as a software that deals with the organization, storage, retrieval and evaluation of information from document repos particularly text based information.
   - The system mostly assists users in finding the information they require but it does not particularly return the answers of the questions.
   - WSD is used to solve the ambiguities of the queries provided to IR system.
   - Similar to machine translation, current IR systems do not especially use WSD module and they depend on the concept that user would type enough context in the query to only retrieve relevant documents.
4. **Lexicography**
   - WSD and lexicography can work together in loop because modern lexicography is collection based.
   - With lexicography, WSD provides rough actual sense groupings as well as statistically significant contextual measure of sense.

4.6 Dictionary (Knowledge) Based Approach

- A simple approach to segment text is to scan each character one at a time from left to right and look up those characters in a dictionary.
- If the series of characters found in the dictionary, then we have a matched word and segment that sequence as a word.

- But this will match a shorter length word.
- There are several ways to better implement this approach.

a. Maximal Matching

- One way to avoid matching the shortest word is to seek out longest sequence of characters within the dictionary.
- This approach is named as the longest matching algorithm or maximal matching. This is often a greedy algorithm that matches the longest word.
- For example, in English, we have these series of characters: "the settlement phase"
- For the first word, we would find: the, theseand no longer word would match after that.
- Now we just choose the longest which is "these", then start again from 't'. But now we don't have any word in this series.
- When we can't match a word, we just mark the first character as unknown.
- So, in "ttlemen...", we just took 't' out as an unknown word and matching the next word starting with 't'.
- Assume the word "ttle" are not in our dictionary, we would get the series of words as "theset t l ementpahse".
- We get four unknown word here.
- But from example the longest word can make incorrect segmentation.
- This result in overextending the first word "these" into a part of the second word "eset" making the next word unknown "t".

b. Bi-Directional Maximal Matching

- One way to solve this issue is to match backward.
- This approach is named as bi-directional maximal matching.
- It goes from left to right i.e., forward matching first and then from the end of the sentence go from right to left i.e., backward matching.
- Then it chooses the best result.
- As seen earlier, the forward gave incorrect segmentation.
- But the backward would give us the correct result

c. Maximum Matching

- One approach to solve the greedy nature of longest matching is an algorithm called 'maximum matching'.

- This approach segments multiple possible combinations and chooses the one with fewer words in the sentence.
- It also prioritizes fewer unknown words.
- This approach gives the correct segmentation from the example text as shown below.

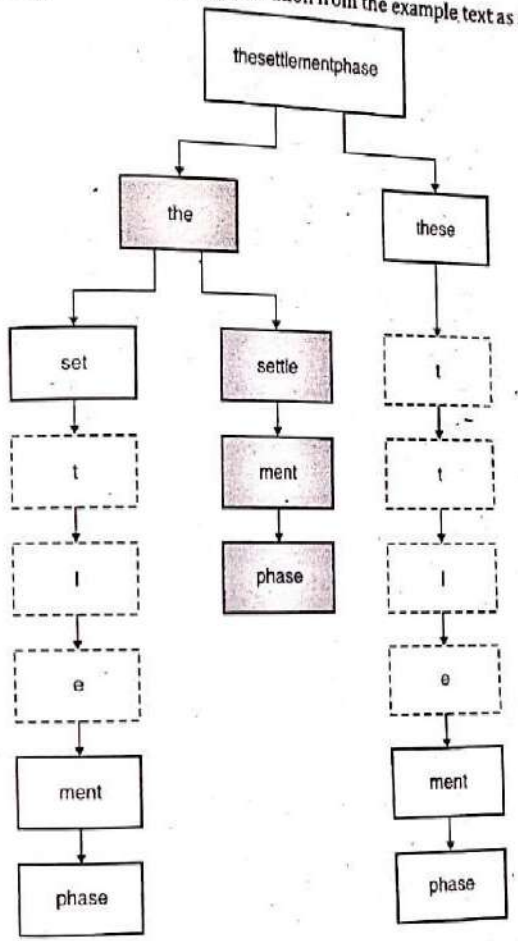


Fig .4.6.1: Maximum Matching

- The first word can be "the" and "these". From each of these nodes, there are multiple choices. Like in "the", the next word can be "set" or "settle".
- The word "t" would result in an incorrect word after "set". Only "settlw" would give use "ment", then "phase" as correct segmentation.

- This approach goes through all the different combinations based on one dictionary.
- So, unknown words are still a problem that needs to be eliminated.
- Also, it can still result in errors with the known words when the correct segment able text requires more words instead of a smaller number of words.

Review Questions

Q.1 Write a short note on lexical ananlysis.

Q.2 Explain the concept of attachments for a fragment of English.

Q.3 Explain the relations among lexemes & their senses.

Q.4 Difference between polysemy and homonymy.

Q.5 Write short note on : wordnet and wsd.

Q.6 Explain the concept of maximum matching.

□□□

5

Module - 5

PRAGMATICS

Syllabus

Discourse –reference resolution, reference phenomenon , syntactic & semantic constraints on co reference

TOPICS

Abstract

Pragmatics studies the processes of language use, while discourse analysis is devoted to its product, i.e. discourse. Pragmatics can be understood in a narrow sense focusing on cognitive-inferential aspects of information processing, and it can be understood in a wider sense in which it also includes social aspects of interaction. In historical pragmatics, the former conceptualization lies behind work on pragmatic explanations in language change, while the latter conceptualization studies earlier language use from a social and interactional perspective, including such aspects as inserts (e.g. interjections and discourse markers), speech acts, and terms of address. Discourse, as the product of language use, can be seen as a stretch of conversation (dialogue) or as a domain of communication. In the former conceptualization, research focuses on the structural properties of the dialogue, and in the latter, it deals with the linguistic practices pertaining to particular fields of knowledge or interaction, e.g. courtroom discourse, the discourse of science, and news discourse.

5.1 Discourse - Reference Resolution

- The most difficult problem of AI is to process the natural language by computers or in other words *natural language processing* is the most difficult problem of artificial intelligence.
- The major problems in NLP, then one of the major problems in NLP is discourse processing - building theories and models of how utterances stick together to form **coherent discourse**.
- Actually, the language always consists of collocated, structured and coherent groups of sentences rather than isolated and unrelated sentences like movies. These coherent groups of sentences are referred to as discourse.

5.1.1 Concept of Coherence

- A sequence of sentences is a "text" when there is some kind of dependence between the sentences. The task of textual analysis is to identify the features that cause this dependence.
- These features have been classified in terms of COHESION and COHERENCE
- COHESION refers to linguistic features which link sentences together and are generally easy to identify.
- The textual world (what the text is about) is made up of concepts and relations. Coherence concerns the way in which concepts and relations are mutually accessible and relevant. In other words, a coherent text is one which is easy for us to understand because it is easy for us to make a mental representation of it. Remember that it is possible for a text to be cohesive but not coherent.

- Coherence refers to a certain characteristic or aspect of writing. Literally, the word means "to stick together." Coherence in writing means that all the ideas in a paragraph flow smoothly from one sentence to the next sentence.
- Coherence in linguistics is what makes a text semantically meaningful. It is especially dealt with in text linguistics

5.1.2 Discourse Structure

- Human discourse often exhibits structures that are intended to indicate common experiences and respond to them
- For example, research abstracts are intended to inform readers in the same community as the authors and who are engaged in similar work.

5.1.3 Discourse Segmentation

- Documents are automatically partitioned into fragments, also known as passages, which are different discourse segments.
- Techniques to separate documents into passages include :
  1. Rule-based systems based on clue words and phrases.
  2. Probabilistic techniques to separate fragments and to identify discourse segments (Oddy)
- TextTiling algorithm uses cohesion to identify segments, assuming that each segment exhibits lexical cohesion within the segment, but is not cohesive across different segments :
  1. Lexical cohesion score - average similarity of words within a segment.
  2. Identify boundaries by the difference of cohesion scores.

Cohesion – Surface Level Ties

- "A piece of text is intended and is perceived as more than a simple sequencing of independent sentences."
- Therefore, a text will exhibit unity / texture on the surface level (cohesion) at the meaning level (coherence).

  Sets forth the linguistic devices that are available in the English language for creating this unity / texture.
- Identifies the features in a text that contribute to an intelligent comprehension of the text.
- Important for language generation, produces natural sounding texts.

Cohesive Relations

- Define dependencies between sentences in text. "He said so."
- "He" and " so " presuppose elements in the preceding text for their understanding
- This presupposition and the presence of information elsewhere in text to resolve this presupposition provide COHESION.
- Part of the discourse-forming component of the linguistic system.
- Provides the means whereby structurally unrelated elements are linked together.
- Types of Cohesive Ties

1.	Grammatical	2.	Reference
3.	Substitution	4.	Ellipsis
5.	Conjunction	6.	Lexical
7.	Reiteration	8.	Collocation

5.2 Coreference Resolution

- The items in a language which, rather than being interpreted in their own right, make reference to something else for their interpretation.
- "Doctor Foster went to Gloucester in a shower of rain. He(preceding) stepped in a puddle right up to *his* (following)middle and never went there again."

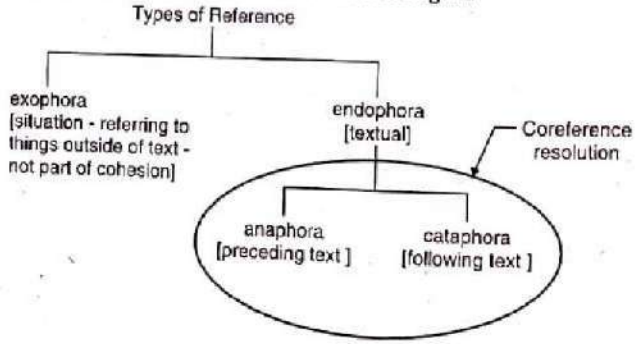


Fig. 5.2.1: Types of Reference

- One of the most important NLP tasks for cohesion at the discourse level
- A linguistic phenomenon of abbreviated subsequent reference

- A cohesive tie of the grammatical and lexical types includes reference, substitution and reiteration.
- A technique for referring back to an entity which has been introduced with more fully descriptive phrasing earlier in the text
- Refers to this same entity but with a lexically and semantically attenuated form

Types of Entity Resolutions

- Entity Resolution is an ability of a system to recognize and unify variant references to a single entity.
- Coreference algorithms usually performed within larger task of entity resolution.
- 2 levels of resolution: – within document (includes co-reference resolution)
- e.g. Bin Ladin = he
- his followers = they
- terrorist attacks = they
- the Federal Bureau of Investigation = FBI = F.B.I – across document (or named entity resolution)
- e.g. Usama Bin Ladin =Osama Bin Ladin = Bin Ladin.
- Event resolution is also possible, but not widely used.

Terminology Examples

- The referent for a referring phrase is found by the resolution algorithm among the candidates, previous noun phrases.

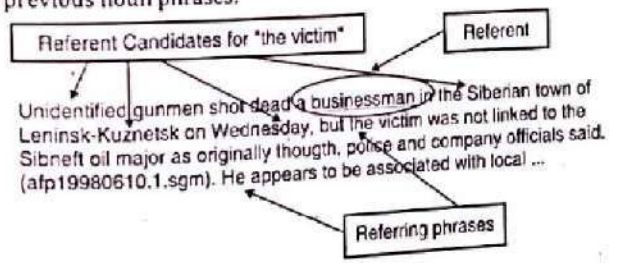


Fig. 5.2.2: Referring Phrase

Reference Types

- An algorithm must first decide which are the referring phrases that must be resolved.

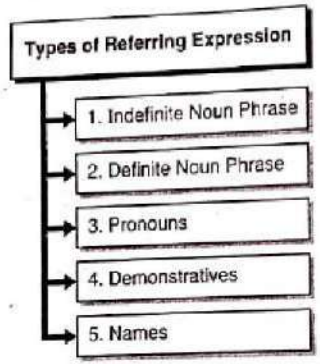


Fig. 5.2.3 : Types of Referring Expressions

1. Indefinite noun phrases

- Typically, an indefinite noun phrase introduces a new entity into the discourse and would not be used as a referring phrase to something else
- The exception is in the case of cataphora: A Soviet pop star was killed at a concert in Moscow last night. Igor Talkov was shot through the heart as he walked on stage.

2. Definite noun phrases

- Definite reference is used to refer to an entity identifiable by the reader because it is either
  a) already mentioned previously (in discourse), or
  b) contained in the reader's set of beliefs about the world (pragmatics), or
  c) the object itself is unique.
- For e.g. Mr. Torres and his companion claimed a hard-shelled black vinyl suitcase. The police rushed the suitcase to the Trans-Uranium Institute

3. Pronouns

- It refer to entities that were introduced fairly recently.
- Nominative (he, she, it, they, etc.)
  - For e.g., The German authorities said a Colombian who had lived for a long time in the Ukraine flew in from Kiev. He had 300 grams of plutonium 239 in his baggage. Oblique (him, her, them, etc.)

- For e.g. Undercover investigators negotiated with three members of a criminal group and arrested them after receiving the first shipment. Possessive (his, her, their, etc. + hers, theirs, etc.)
- For e.g., He3 had 300 grams of plutonium 239 in his baggage. The suspected smuggler denied that the materials were his. (*chain). Reflexive (himself, themselves, etc.)
- For e.g. There appears to be a growing problem of disaffected loners who cut themselves off from all groups.

4. Demonstratives – this and that

- Demonstrative pronouns can either appear alone or as determiners this ingredient, that spice
- These NP phrases with determiners are ambiguous.
- They can be indefinite : *I saw this beautiful car today.*

  Or they can be definite :*I just bought a copy of Thoreau's Walden. I had bought one five years ago. That one had been very tattered; this one was in much better condition.*

5. Names

Names can occur in many forms, sometimes called name variants.

Approach to coreference resolution

- Naively identify all referring phrases for resolution:
  - all Pronouns
  - all definite NPs
  - all Proper Nouns
- Filter things that look referential but, in fact, are not
  - e.g. geographic names, the United State
  - pleonastic "it", e.g. it's 3:45 p.m., it was cold
  - non-referential "it" , "they" , "there"
- For e.g. it was essential, important, is understood, they say, there seems to be a mistake

Identify Referent Candidates

- All noun phrases (both indef. and def.) are considered potential referent candidates.
- A referring phrase can also be a referent for a subsequent referring phrase.

- Example : (omitted sentence with name of suspect) He had 300 grams of plutonium 239 in his baggage. The suspected smuggler denied that the materials were his. (chain of 4 referring phrases)
- All potential candidates are collected in a table collecting feature info on each candidate.

5.3 Syntactic and Semantic Constraints on Coreference

- Having described a variety of reference phenomena that are found in natural language, we can now consider how one might develop algorithms for identifying the referents of referential expressions.
- One step that needs to be taken in any successful reference resolution algorithm is to filter the set of possible referents on the basis of certain relatively hard-and-fast constraints.
- We describe some of these constraints here. Number Agreement Referring expressions and their referents must agree in number; for English, this means distinguishing between singular and plural references. A categorization of pronouns with respect to number is shown in Fig. 5.3.1

Singular	Plural	Unspecified
She, her, he, him, his, it	We, us, they, them	your

Fig.5.3.1.Number Agreement in the English pronominal system.

The following examples illustrate constraints on number agreement.

(5.26) John has a new Acura. It is red.

(5.27) John has three new Acuras. They are red.

(5.28) * John has a new Acura. They are red.

(5.29) * John has three new Acuras. It is red.

Person and Case Agreement

- English distinguishes between three forms of person: first, second, and third. A categorization of pronouns with respect to person is shown in Fig. 5.3.2.
- The following examples illustrate constraints on person agreement.

(5.30) You and I have Acuras. We love them.

	First	Second	Third
Nominative	I, we	You	He, she, they
Accusative	Me, us	You	Him, her, them
Genitive	My, our	Your	His, her, their

Fig.5.3.2:Person and case agreement in the English pronominal system

(5.31) John and Mary have Acuras. They love them.

(5.32) * John and Mary have Acuras. We love them. (where We=John and Mary)

(5.33) * You and I have Acuras. They love them. (where They=You and I)

- In addition, English pronouns are constrained by case agreement; different forms of the pronoun may be required when placed in subject position (nominative case, e.g., he, she, they), object position (accusative case, e.g., him, her, them), and genitive position (genitive case, e.g., his Acura, her Acura, their Acura). This categorization is also shown in Fig. 5.3.2
- Gender Agreement Referents also must agree with the gender specified by the referring expression. English third person pronouns distinguish between male, female, and nonpersonal genders, and unlike some languages, the first two only apply to animate entities. Some examples are shown in Fig. 5.3.3

Masculine	Feminine	Nonpersonal
He, him, his	She, her	it

Fig.5.3.3:Gender agreement in the English pronominal system

- The following examples illustrate constraints on gender agreement.

(5.34) John has an Acura. He is attractive. (he=John, not the Acura)

(5.35) John has an Acura. It is attractive. (it=the Acura, not John)

5.3.1 Syntactic Constraints Reference

- Relations may also be constrained by the syntactic relationships between a referential expression and a possible antecedent noun phrase when both occur in the same sentence.
- For instance, the pronouns in all of the following sentences are subject to the constraints indicated in brackets.

(5.36) John bought himself a new Acura. [himself=John]

(5.37) John bought him a new Acura. [him ≠John]

(5.38) John said that Bill bought him a new Acura. [him ≠ Bill]

(5.39) John said that Bill bought himself a new Acura. [himself=Bill]

(5.40) He said that he bought John a new Acura. [He ≠ John; he ≠John]

- English pronouns such as himself, herself, and themselves are called REFLEXIVES.
- Oversimplifying the situation considerably, a reflexive co-refers with the subject of the most immediate clause that contains it (ex. 5.36), whereas a nonreflexive cannot corefer with this subject (ex. 5.37).
- That this rule applies only for the subject of the most immediate clause is shown by examples (5.38) and (5.39), in which the opposite reference pattern is manifest between the pronoun and the subject of the higher sentence.
- On the other hand, a full noun phrase like John cannot corefer with the subject of the most immediate clause nor with a higher-level subject (ex. 5.40).
- Whereas these syntactic constraints apply to a referring expression and a particular potential antecedent noun phrase, these constraints actually prohibit coreference between the two regardless of any other available antecedents that denote the same entity.
- For instance, normally a nonreflexive pronoun like him can corefer with the subject of the previous sentence as it does in example (5.41), but it cannot in example (5.42) because of its syntactic relationship with the coreferential pronoun he in the second clause.

(5.41) John wanted a new car. Bill bought him a new Acura. [him=John]

(5.42) John wanted a new car. He bought him a new Acura. [he=John; him ≠John]

- These rules oversimplify the situation in a number of ways, and there are many cases that they do not cover. Indeed, upon further inspection the facts actually get quite complicated.
- In fact, it is unlikely that all of the data can be explained using only syntactic relations (Kuno, 1987). For instance, the reflexive himself and the nonreflexive him in sentences (5.43) and (5.44) respectively can both refer to the subject John, even though they occur in identical syntactic configurations.

(5.43) John set the pamphlets about Acuras next to himself. [himself=John]

(5.44) John set the pamphlets about Acuras next to him. [him=John]

- For the algorithms discussed later in this chapter, however, we will assume a syntactic account of restrictions on intrasentential coreference.

5.3.2 Selectional Restrictions

- The selectional restrictions that a verb places on its arguments may be responsible for eliminating referents, as in example (5.45)

(5.45) John parked his Acura in the garage.

- He had driven it around for hours. There are two possible referents for it, the Acura and the garage. The verb drive, however, requires that its direct object denote something that can be driven, such as a car, truck, or bus, but not a garage.
- Thus, the fact that the pronoun appears as the object of drive restricts the set of possible referents to the Acura.
- It is conceivable that a practical NLP system would include a reasonably comprehensive set of selectional constraints for the verbs in its lexicon.
- Selectional restrictions can be violated in the case of metaphor consider example (5.46).

(5.46) John bought a new Acura. It drinks gasoline like you would not believe.

- While the verb drink does not usually take an inanimate subject, its metaphorical use here allows it to refer to a new Acura.
- Of course, there are more general semantic constraints that may come into play, but these are much more difficult to encode in a comprehensive manner. Consider passage (5.47).

(5.47) John parked his Acura in the garage. It is incredibly messy, with old bike and car parts lying around everywhere.

- Here the referent of it is almost certainly the garage, due in part to the fact that a car is probably too small to have bike and car parts laying around "everywhere".
- Resolving this reference requires that a system have knowledge about how large cars typically are, how large garages typically are, and the typical types of objects one might find in each.
- On the other hand, one's knowledge about Beverly Hills might lead one to assume that the Acura is indeed the referent of it in passage (5.48).

(5.48) John parked his Acura in downtown Beverly Hills.

- It is incredibly messy, with old bike and car parts lying around everywhere.
- In the end, just about any knowledge shared by the discourse participants might be necessary to resolve a pronoun reference. However, due in part to the vastness of such knowledge, practical algorithms typically do not rely on it heavily.

Preferences in Pronoun Interpretation

- We discussed relatively strict constraints that algorithms should apply when determining possible referents for referring expressions.
- We now discuss some more readily violated preferences that algorithms can be made to account for.
- These preferences have been posited to apply to pronoun interpretation in particular.
- Since the majority of work on reference resolution algorithms has focused on pronoun interpretation, we will similarly focus on this problem in the remainder of this section.

Recency

- Most theories of reference incorporate the notion that entities introduced in recent utterances are more salient than those introduced from utterances further back.
- Thus, in example (5.49), the pronoun it is more likely to refer to the Legend than the Integra.

(5.49) John has an Integra. Bill has a Legend. Mary likes to drive it.

Grammatical Role

- Many theories specify a salience hierarchy of entities that is ordered by the grammatical position of the expressions which denote them. These typically treat entities mentioned in subject position as more salient than those in object position, which are in turn more salient than those mentioned in subsequent positions.
- Passages such as (5.50) and (5.51) lend support for such a hierarchy.
- Although the first sentence in each case expresses roughly the same propositional content, the preferred referent for the pronoun him varies with the subject in each case – John in (5.50) and Bill in (5.51).
- In example (5.52), the references to John and Bill are conjoined within the subject position. Since both seemingly have the same degree of salience, it is unclear to which the pronoun refers.

(5.50) John went to the Acura dealership with Bill. He bought an Integra. [he = John]

(5.51) Bill went to the Acura dealership with John. He bought an Integra. [he = Bill]

(5.52) John and Bill went to the Acura dealership. He bought an Integra. [he = ??].

Repeated Mention

- Some theories incorporate the idea that entities that have been focused on in the prior discourse are more likely to continue to be focused on in subsequent discourse, and hence references to them are more likely to be pronominalized.

- For instance, whereas the pronoun in example (5.51) has Bill as its preferred interpretation, the pronoun in the final sentence of example (5.53) is more likely to refer to John.

(5.53) John needed a car to get to his new job.

He decided that he wanted something sporty. Bill went to the Acura dealership with him. He bought an Integra. [he = John].

Parallelism

- There are also strong preferences that appear to be induced by parallelism effects, as in example (5.54).

(5.54) Mary went with Sue to the Acura dealership. Sally went with her to the Mazda dealership. [her = Sue]

- The grammatical role hierarchy described above ranks Mary as more salient than Sue, and thus should be the preferred referent of her.
- Furthermore, there is no semantic reason that Mary cannot be the referent. Nonetheless, her is instead understood to refer to Sue. This suggests that we might want a heuristic which says that nonsubject pronouns prefer non-subject referents.
- However, such a heuristic may not work for cases that lack the structural parallelism of example (5.54), such as example (5.55), in which Mary is the preferred referent of the pronoun instead of Sue.

(5.55) Mary went with Sue to the Acura dealership. Sally told her not to buy anything. [her = Mary]

Verb Semantics

- Certain verbs appear to place a semantically-oriented emphasis on one of their argument positions, which can have the effect of biasing the manner in which subsequent pronouns are interpreted. Compare sentences (5.56) and (5.57).

(5.56) John telephoned Bill. He lost the pamphlet on Acuras.

(5.57) John criticized Bill. He lost the pamphlet on Acuras.

- These examples differ only in the verb used in the first sentence, yet the subject pronoun in passage (5.56) is typically resolved to John, whereas the pronoun in passage (5.57) is resolved to Bill.
- Some researchers have claimed that this effect results from what has been called the "implicit causality" of a verb: the implicit cause of a "criticizing" event is considered to be its object, whereas the implicit cause of a "telephoning" event is considered to be its subject.
- This emphasis results in a higher degree of salience for the entity in this argument position, which leads to the different preferences for examples (5.56) and (5.57).

- Similar preferences have been articulated in terms of the thematic roles that the potential antecedents occupy. For example, most hearers resolve He to John in example (5.58) and to Bill in example (5.59).
- Although these referents are evoked from different grammatical role positions, they both fill the Goal thematic role of their corresponding verbs, whereas the other potential referent fills the Source role. Likewise, hearersgenerally resolve
- He to John and Bill in examples (5.60) and (5.61) respectively, providing evidence that fillers of the Stimulus role are preferred over fillers of the Experiencer role. (5.58) John seized the Acura pamphlet from Bill. He loves reading about cars. (Goal=John, Source=Bill)

  (5.59) John passed the Acura pamphlet to Bill. He loves reading about cars. (Goal=Bill, Source=John)

  (5.60) The car dealer admired John. He knows Acuras inside and out. (Stimulus=John, Experiencer=the car dealer)

  (5.61) The car dealer impressed John. He knows Acuras inside and out. (Stimulus=the car dealer, Experiencer=John).

Review Questions

Q.1 Write a short note on : syntactic & semantic constraints on co reference.

Q.2 Explain the concept of Discourse –reference resolution.

❑❑❑

Module - 6

6 APPLICATIONS (Preferably For Indian Regional Languages)

Syllabus

Machine translation, Information retrieval, Question answers system, categorization, summarization, sentiment analysis, Named Entity Recognition.

TOPICS

6.1 Machine Translation

- Machine Translation (MT) or automated translation refers to the process when computer software translates a text from the source language into the target language without human intervention.
- Machine Translation (MT) is the task to translate a text from a source language to its counterpart in a target language.
- There are many challenging aspects of MT:
  1) The large variety of languages, alphabets and grammars;
  2) The task to translate a sequence (a sentence for example) to a sequence is harder for a computer than working with numbers only;
  3) There is no *one* correct answer (e.g.: translating from a language without gender-dependent pronouns, *he* and *she* can be the same).
- There are three types of machine translation methods, described here in simple terms:
  1) Rules-based machine translation uses grammar and language rules that have been developed by language experts, as well as customized dictionaries
  2) Statistical machine translation learns how to translate by analyzing a large number of existing human translations
  3) Neural machine translation teaches itself how to translate by using a large neural network. This method is becoming increasingly popular as it often provides the best results.

6.1.1 Modern Machine Translation

- Software, such as that produced by SYSTRAN or IBM, allows for customization by domain or profession (such as weather reports) - improving output by limiting the scope of allowable substitutions.
- This technique is particularly effective in domains where formal or formulaic language is used.
- Improved output quality can also be achieved by human intervention: for example, some systems are able to translate more accurately if the user has unambiguously identified which words in the text are names.
- With the assistance of these techniques, MT has proven useful as a tool to assist human translators, and in some cases can even produce output that can be used "as is".
- Machine translation (MT) is the application of computers to the task of translating texts from one natural language to another.

- One of the very earliest pursuits in computer science, MT has proved to be an elusive goal, but today a number of systems are available which produce output which, if not perfect, is of sufficient quality to be useful in a number of specific domains.

6.1.2 Approaches

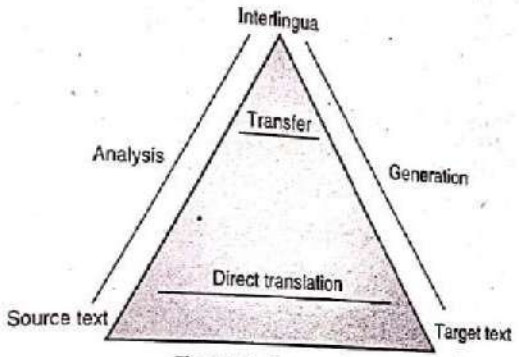


Fig. 6.1.1 : Approaches

- Pyramid showing comparative depths of intermediary representation, interlingual machine translation at the peak, followed by transfer-based, then direct translation.
- Machine translation can use a method based on linguistic rules, which means that words will be translated in a linguistic way , the most suitable (orally speaking) words of the target language will replace the ones in the source language.
- It is often argued that the success of machine translation requires the problem of natural language understanding to be solved first.
- Generally, rule-based methods parse a text, usually creating an intermediary, symbolic representation, from which the text in the target language is generated.
- According to the nature of the intermediary representation, an approach is described as interlingual machine translation or transfer-based machine translation.
- These methods require extensive lexicons with morphological, syntactic, and semantic information, and large sets of rules.
- Given enough data, machine translation programs often work well enough for a native speaker of one language to get the approximate meaning of what is written by the other native speaker.
- The difficulty is getting enough data of the right kind to support the particular method.
- For example, the large multilingual corpus of data needed for statistical methods to work is not necessary for the grammar-based methods. But then, the grammar methods need a skilled linguist to carefully design the grammar that they use.

6.1.3 Different Types of Machine Translation

- There are four types of machine translation :

1. **Statistical Machine Translation (SMT)**
2. **Rule-based Machine Translation (RBMT)**
3. **Hybrid Machine Translation, and**
4. **Neural Machine Translation.**

- Machine translation is primarily a tool that helps marketers/translators achieve a goal. It is not a replacement for the older systems of translation.
- Rather, it's an enhancement. For instance, in a traditional localization cycle, we encounter what is called the TEP phase.
- TEP here stands for 'translate, edit, and proof. Now, in a TEP cycle, the role of machine translation starts and ends with 'T,' which is 'translation.'
- The rest of the work, which is editing and proofing, still needs to be carried out by professional translators and language experts. But, the goal is still the same, irrespective of which approach you follow.
- The client must be provided with top-notch translation work. Attention to style, terminology, and syntax are significant in the localized results. This is why machine translation engines require them as input.
- To make things even more complicated, the turnover time today is drastically low. Human effort simply will not cut it. Integrating machine translation into the localization strategy is a must now.

1. Statistical Machine Translation (SMT)

- SMT works by referring to statistical models that are based on the analysis of large volumes of bilingual text.
- It aims to determine the correspondence between a word from the source language and a word from the target language.
- A good example of this is Google Translate. Now, SMT is great for basic translation, but its greatest drawback is that it does not factor in context, which means translations can often be erroneous. In other words, don't expect high-quality translations.

2. Rule-Based Machine Translation (RBMT)

- RBMT, on the other hand, translates on the basis of grammatical rules.

- It conducts a grammatical analysis of the source language and the target language to generate the translated sentence. However, RBMT requires extensive proofreading, and its heavy dependence on lexicons means that efficiency is achieved after a long period of time.

3. Hybrid Machine Translation (HMT)

- HMT, as the term indicates, is a blend of RBMT and SMT. It leverages a translation memory, making it far more effective in terms of quality.
- However, even HMT has its share of drawbacks, the greatest of which is the need for extensive editing. Human translators will be required.

4. Neural Machine Translation (NMT)

- NMT is a type of machine translation that depends on neural network models (based on the human brain) to develop statistical models for the purpose of translation.
- The primary benefit of NMT is that it provides a single system that can be trained to decipher the source and target text.

As a result, it does not depend on specialized systems that are common to other machine translation systems, especially SMT.

6.1.4 The benefits and Uses of Machine Translation

- Machine translation system enables you to save your time while translating large texts. If a professional translator translates your text, you have to pay enough money for each page but very often we need just a point of matter, general idea.
- Machine translation gives you a quick and comprehensive understanding of a document.
- Translation enables effective communication between people around the world.
- It is a courier for the transmission of knowledge, a protector of cultural heritage, and essential to the development of a global economy.
- Highly skilled translators are key. Translation Studies helps practitioners develop those skills
- Machine Translation enables global companies to translate content at scale using "machines" such as Google Translate.
- Cost-efficient for large volumes of translations. Reduced time-to-market due to faster translation delivery.
- It is often found as a feature that is integrated into localization platforms and used by companies looking to lower their translation costs.

6.1.5 Difference between Rule-Based MT vs. Statistical MT

Sr. No.	Rule-Based MT	Statistical MT
1.	Consistency between versions	Inconsistency between versions
2.	Rule-based MT provides good out-of-domain quality and is by nature predictable.	Statistical MT provides good quality when large and qualified corpora are available.
3.	Knows grammatical rules	Does not know grammar
4.	High performance and robustness	High CPU and disk space requirements
5.	Lack of fluency	Good fluency
6.	Hard to handle exceptions to rules	Good for catching exceptions to rules
7.	High development and customization costs	Rapid and cost-effective development costs provided the required corpus exists

6.2 Information Retrieval

- Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).
- These days we frequently think first of web search, but there are many other cases:
  - E-mail search
  - Searching your laptop
  - Corporate knowledge bases
  - Legal information retrieval
- With the help of the Fig. 6.2.1, we can understand the process of information retrieval (IR) –

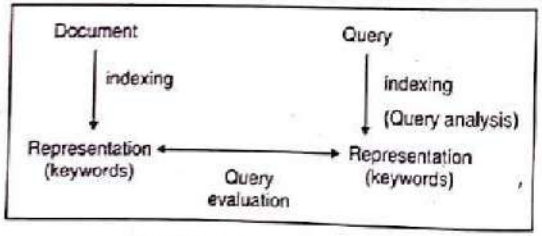


Fig.6.2.1 : Information retrieval

It is clear from the above diagram that a user who needs information will have to formulate a request in the form of query in natural language. Then the IR system will respond by retrieving the relevant output, in the form of documents, about the required information.

6.2.1 Types of IR Models

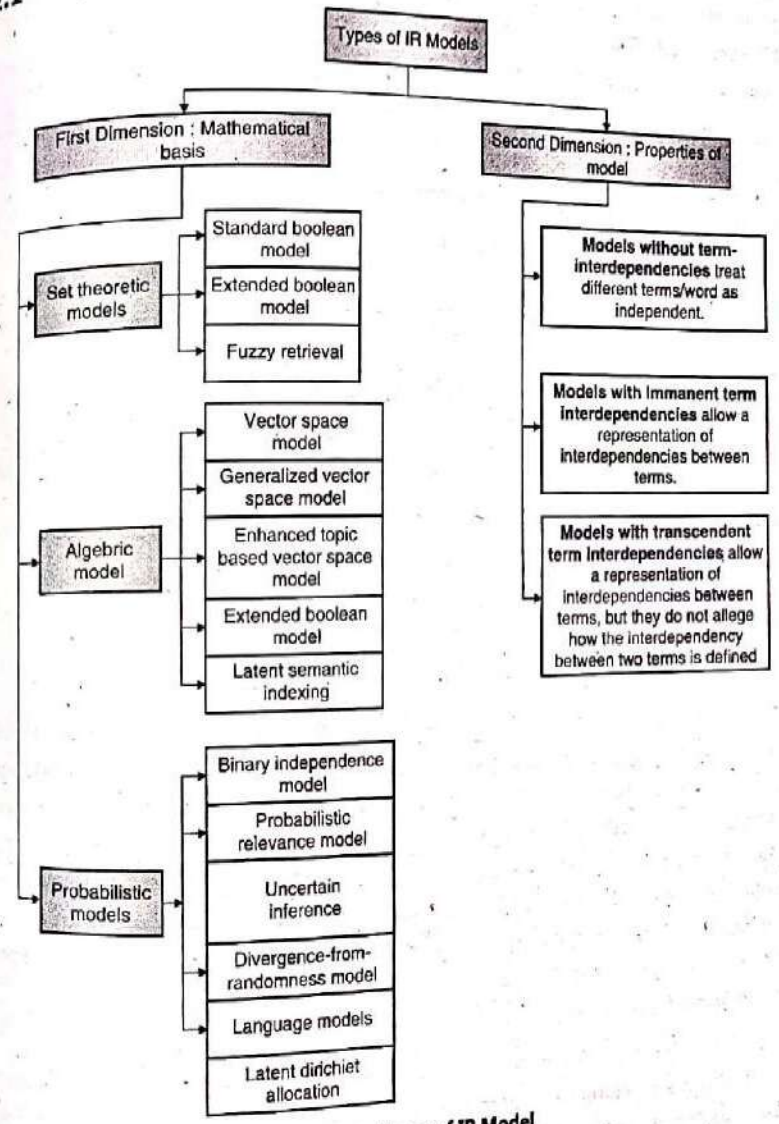


Fig.6.2.2 : Types of IR Model

An information model (IR) model can be classified into the following three models

1. Classical IR Model

It is the simplest and easy to implement IR model. This model is based on mathematical knowledge that was easily recognized and understood as well. Boolean, Vector and Probabilistic are the three classical IR models.

2. Non-Classical IR Model

It is completely opposite to classical IR model. Such kind of IR models are based on principles other than similarity, probability, Boolean operations. Information logic model, situation theory model and interaction models are the examples of non-classical IR model.

3. Alternative IR Model

It is the enhancement of classical IR model making use of some specific techniques from some other fields. Cluster model, fuzzy model and latent semantic indexing (LSI) models are the example of alternative IR model.

Design features of Information retrieval (IR) systems

Let us now learn about the design features of IR systems –

1. Inverted Index

The primary data structure of most of the IR systems is in the form of inverted index. We can define an inverted index as a data structure that list, for every word, all documents that contain it and frequency of the occurrences in document. It makes it easy to search for 'hits' of a query word.

2. Stop Word Elimination

Stop words are those high frequency words that are deemed unlikely to be useful for searching. They have less semantic weights. All such kind of words are in a list called stop list. For example, articles "a", "an", "the" and prepositions like "in", "of", "for", "at" etc. are the examples of stop words. The size of the inverted index can be significantly reduced by stop list. As per Zipf's law, a stop list covering a few dozen words reduces the size of inverted index by almost half. On the other hand, sometimes the elimination of stop word may cause elimination of the term that is useful for searching. For example, if we eliminate the alphabet "A" from "Vitamin A" then it would have no significance.

3. Stemming

Stemming, the simplified form of morphological analysis, is the heuristic process of extracting the base form of words by chopping off the ends of words. For example, the words laughing, laughs, laughed would be stemmed to the root word laugh.

In our subsequent sections, we will discuss about some important and useful IR models.

The Boolean Model

It is the oldest information retrieval (IR) model. The model is based on set theory and the Boolean algebra, where documents are sets of terms and queries are Boolean expressions on terms. The Boolean model can be defined as :

- Document = Logical conjunction of keywords
- Query = Boolean expression of keywords
- $R(D, Q) = D \rightarrow Q$

e.g. $D = t_1 \wedge t_2 \wedge \ldots \wedge t_n$

$Q = (t_1 \wedge t_2) \vee (t_3 \wedge t_4)$

$D \rightarrow Q$, thus $R(D, Q) = 1$.

- **Problems:**
  - R is either 1 or 0 (unordered set of documents)
  - many documents or few documents
  - End-users cannot manipulate Boolean operators correctly

    e.g. documents about *kangaroos* and *koalas*
  - $D = \{\ldots, (t_i, w_i), \ldots\}$: weighted keywords
- **Interpretation :**
  - D is a member of class t_i to degree w_i.
  - In terms of fuzzy sets: $\mu_{ti}(D) = w_i$

A possible Evaluation :

$$R(D, t_i) = \mu_{ti}(D);$$

$$R(D, Q_1 \wedge Q_2) = \min(R(D, Q_1), R(D, Q_2));$$

$$R(D, Q_1 \vee Q_2) = \max(R(D, Q_1), R(D, Q_2));$$

$$R(D, \neg Q_1) = 1 - R(D, Q_1).$$

1. Vector Space Model

- Consider the following important points to understand more about the Vector Space Model –
  - The index representations (documents) and the queries are considered as vectors embedded in a high dimensional Euclidean space.

- The similarity measure of a document vector to a query vector is usually the cosine of the angle between them.

- Vector space = all the keywords encountered

  $<t_1, t_2, t_3, ..., t_n>$

- Document

  $D = <a_1, a_2, a_3, ..., a_n>$

  a_i = weight of t_i in D

- Query

  $Q = <b_1, b_2, b_3, ..., b_n>$

  b_i = weight of t_i in Q

- R(D,Q) = Sim(D,Q)

Document space ↓	t_1	t_2	t_3	...	t_n ← Team Vector space
D_1	a_{11}	a_{12}	a_{13}	...	a_{1n}
D_2	a_{21}	a_{12}	a_{13}	...	a_{1n}
D_3	A_{31}	a_{12}	a_{13}	...	a_{1n}
....					
D_n	a_{m1}	a_{m2}	a_{m3}	...	a_{mn}
Q	b_1	b_2	b_3	...	b_n

- Matrix is very sparse: a few 100s terms for a document, and a few terms for a query, while the term space is large (~100k)

- Stored as :

  D1 → {(t1, a1), (t2,a2), ...}

  t1 → {(D1,a1), ...}

  Document Frequency (df_i)

- It may be defined as the total number of documents in the collection in which w_i occurs. It is an indicator of informativeness. Semantically focused words will occur several times in the document unlike the semantically unfocused words.

- Collection Frequency (cf_i)

- It may be defined as the total number of occurrences of w_i in the collection.

Mathematically,

$$df_i \leq cf_i \text{ and } \sum_j tf_{ij} = cf_i \quad df_i \leq cf_i \text{ and } \sum_j tf_{ij} = cf_i$$

The different forms of document frequency weighting. The forms are described below :

2. Term Frequency Factor

This is also classified as the term frequency factor, which means that if a term *t* appears often in a document then a query containing *t* should retrieve that document. We can combine word's **term frequency (tf_{ij})** and **document frequency (df_i)** into a single weight as follows :

$$\text{weight}(i,j) = \{(1+\log(tf_{ij}))\log N \; df_i \text{ if } tf_{i,j} \geq 1 \text{ 0 if } tf_{i,j}$$

$$= \text{weight}(i,j) = \{(1+\log(tf_{ij}))\log N df_i \text{ if } tf_{i,j} \geq 1 \text{ 0 if } tf_{i,j} = 0$$

Here N is the total number of documents.

3. Inverse Document Frequency (idf)

The important point of idf weighting is that the term's scarcity across the collection is a measure of its importance and importance is inversely proportional to frequency of occurrence.

Mathematically,

$$\text{Idf}_t = \log(1+N/n_t) \quad \text{idf}_t = \log(1+N/n_t)$$

$$\text{Idf}_t = \log(N-n_t/n_t) \quad \text{idf}_t = \log(N-n_t/n_t)$$

N = documents in the collection

n_t = documents containing term *t*

6.2.2 Difference between Data Retrieval and Information Retrieval

Table 6.2.1

Sr. No.	Data Retrieval	Information Retrieval
1	Determines which documents of a collection Contain the keywords in the user query.	Retrieves information about a subject Rather than data which satisfies a given query.

Sr. No.	Data Retrieval	Information Retrieval
2	All objects which satisfy clearly defined conditions are retrieved	IR system somehow 'interprets' the contents of documents in a collection and rank them according to a degree of relevance to the user query.
3	A single erroneous object means total failure	The retrieved objects might be inaccurate and small errors are ignored
4	Data has a well-defined structure and semantics	Data is a natural language text which is not always well structured and could be semantically ambiguous.

6.3 Question Answers System

- Question Answer System is a branch of learning of information retrieval and natural language processing, which focuses on building systems that automatically answer questions posed by users in a natural language.
- An understanding of natural language consists of the ability of a system to decode sentences into a representation so the system generates valid answers to questions asked by an user.
- Effective answers mean answers relevant to the questions posed by the user. As the representation of natural language, sentences must effectively map semantics of the statement.
- To form an answer it is necessary to execute the syntax and semantic analysis of a question.
- The process of the system is as follows :
  1. Query Processing
  2. Document Retrieval
  3. Passage Retrieval
  4. Answer Extraction

Step 1: Query Processing

Classify question into seven categories

- **Who** is/was/are/were...?
- **When** is/did/will/are/were ...?
- **Where** is/are/were ...?

Category-specific transformation rules

a. eg "For Where questions, move 'is' to all possible locations"

"Where is the Louvre Museum located"

→ "is the Louvre Museum located"

→ "the is Louvre Museum located"

→ "the Louvre is Museum located"

→ "the Louvre Museum is located"

→ "the Louvre Museum located is"

b. Expected answer "Datatype" (eg, Date, Person, Location, ...)

- **When** was the French Revolution? → DATE

Hand-crafted classification/rewrite/datatyperules
(Could they be automatically learned?)

Send all rewrites to a Web search engine

- Retrieve top N answers (100?)
- For speed, rely just on search engine's "snippets", not the full text of the actual document
- Some query rewrites are more reliable than others

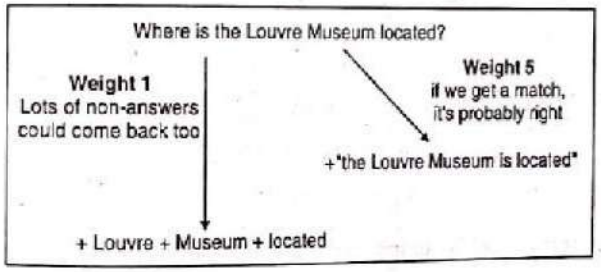


Fig. 6.3.1 : Example of query processing

Step 2: Document Retrieval

- In the document retrieval, we will retrieve relevant documents by using the generated query.
- Users submit queries corresponding to their information need
- System returns (voluminous) list of full-length documents
- Then the users find their original information need, within the returned documents

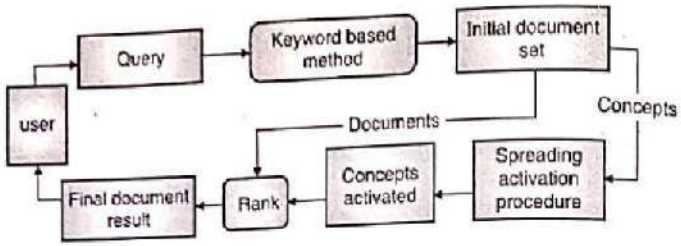


Fig. 6.3.2 : The flow of document retrieval

Step 3: Passage Retrieval

- Passage Retrieval (PR) systems have been proposed for English language. PR systems works on parts of text so that they can limit the relevance of a document to a query, besides detecting document portions that are likely to contain the required answer(rather than the full document).
- The documents are separated into smaller units (passages) such as sentences and paragraphs, and passages that are likely to contain an answer are nominated. If the document is short, it is not needed, but if it is long, passage selection is effective because being uninformed which part of the document the query matched. Generally, the following answer extraction process takes a duration of time to complete, so selecting passages also has the advantage of fast moving up the entire system.

Step 4: Answer Extraction

- Answer Extraction module process the top ranked passages for extracting the final answer to the user's question. Typically, named-entity recognition technique is used to find candidate answers that match the question's named entities.
- For questions that are looking for dates a pattern matching technique is used. Questions that requires descriptive answers like how & why cosine similarity is used to find the answers
- Answer extraction extracts answers from passages. Here, the question and passage are input to the answer extraction model, and the model outputs the answer offset with the score. It then ranks the answers based on the score and presents the answers with the highest scores to the user as the final answer.

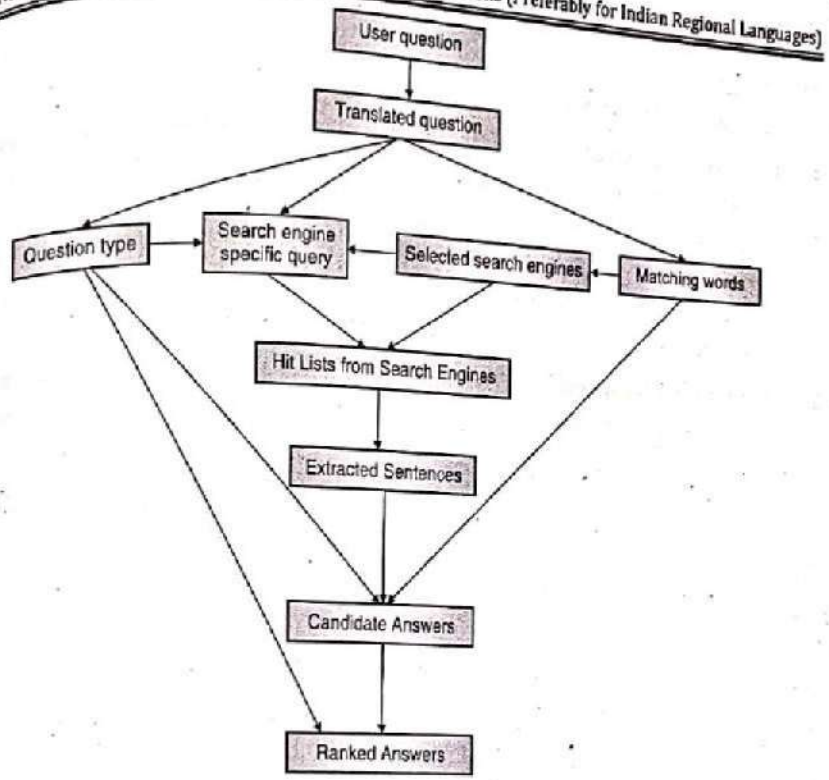


Fig. 6.3.3 : Answer Extraction

6.4 Categorization

- Text classification also known as *text tagging* or *text categorization* is the process of categorizing text into organized groups. By using Natural Language Processing (NLP), text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content.

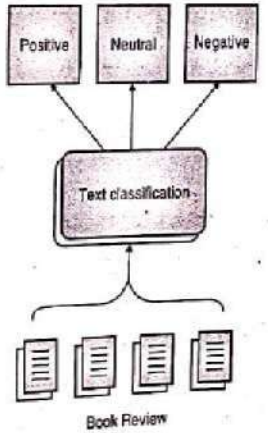


Fig.6.4.1

Approaches

Text Classification can be achieved through three main approaches :

1. **Rule-based approaches :** These approaches make use of handcrafted linguistic rules to classify text. One way to group text is to create a list of words related to a certain column and then judge the text based on the occurrences of these words. For example, words like "fur", "feathers", "claws", and "scales" could help a zoologist identify texts talking about animals online. These approaches require a lot of domain knowledge to be extensive, take a lot of time to compile, and are difficult to scale.
2. **Machine learning approaches :** We can use machine learning to train models on large sets of text data to predict categories of new text. To train models, we need to transform text data into numerical data – this is known as **feature extraction**. Important feature extraction techniques include *bag of words* and *n-grams*. There are several useful machine learning algorithms we can use for text classification. The most popular ones are:
   - Naive Bayes classifiers
   - Support vector machines
   - Deep learning algorithms
3. **Hybrid approaches :**These approaches are a combination of the two algorithms above. They make use of both rule-based and machine learning techniques to model a classifier that can be fine-tuned in certain scenarios.

Some of the most common examples and use cases for automatic text classification include the following :

- **Sentiment Analysis :** the process of understanding if a given text is talking positively or negatively about a given subject (e.g. for brand monitoring purposes).
- **Topic Detection :** the task of identifying the theme or topic of a piece of text (e.g. know if a product review is about *Ease of Use*, *Customer Support*, or *Pricing* when analyzing customer feedback).
- **Language Detection :** the procedure of detecting the language of a given text (e.g. know if an incoming support ticket is written in English or Spanish for automatically routing tickets to the appropriate team).
- **Natural Language Processing (NLP)** is a wide area of research where the worlds of artificial intelligence, computer science, and linguistics collide. It includes a bevy of interesting topics with cool real-world applications, like **named entity recognition, machine translation** or **machine question answering**. Each of these topics has its own way of dealing with textual data.

- But before diving into the deep end and looking at these more complex applications, we need to wade in the shallow end and understand how simpler tasks such as **text classification** are performed.
- Text classification offers a good framework for getting familiar with textual data processing without lacking interest, either. In fact, there are many interesting applications for text classification such as **spam detection** and **sentiment analysis**.
- **Pre-Processing:** A simple approach is to assume that the smallest unit of information in a text is the word (as opposed to the character). Therefore, we will be representing our texts as word sequences. For instance:

  Text: This is a cat. -->**Word Sequence:** [this, is, a, cat]
- In this example, we removed the punctuation and made each word lowercase because we assume that punctuation and letter case don't influence the meaning of words. In fact, we want to avoid making distinctions between similar words such as *This* and *this* or *cat.* and *cat.*
- Moreover, real life text is often "dirty." Because this text is usually automatically scraped from the web, some HTML code can get mixed up with the actual text. So we also need to tidy up these texts a little bit to avoid having HTML code words in our word sequences. For example :

  This is not a sentence.

  <\div> --> [this, is, not, a, sentence]
- Making these changes to our text before turning them into word sequences is called **pre-processing**. Despite being very simple, the pre-processing techniques we have seen so far work very well in practice. Depending on the kind of texts you may encounter, it may be relevant to include more complex pre-processing steps. But keep in mind that the more steps you add, the longer the pre-processing will take.
- A regular expression (or **regex**) is a sequence of characters that represent a search pattern. Each character has a meaning; for example, means any character that isn't the newline character: '\n'. These characters are often combined with quantifiers, such as *, which means zero or more. Combining these two characters, we can make the regex that looks for an expression in the form '<' + 'zero or more' of 'anything but \n' + '>'. This regex is <.*?>. Here the character? indicates a non-greedy search:
- **Input string:** <a>bcd>Difference between greedy and non-greedy search :

  greedy: <.*> --><a>bcd>

  non-greedy: <.*?> --><a>

- Regular expressions are very useful for processing strings. For example, the <.*?> regex we introduced before can be used to detect and remove HTML tags. But we will also be using other regex such as \' to remove the character ' so that words like that's become thats instead of two separate words that and s.
- **Vectorization :** Now that we have a way to extract information from text in the form of word sequences, we need a way to transform these word sequences into numerical features, this is Vectorization.
- The simplest text Vectorization technique is Bag Of Words (BOW). It starts with a list of words called the vocabulary (this is often all the words that occur in the training data). Then, given an input text, it outputs a numerical vector which is simply the vector of word counts for each word of the vocabulary.
- **For example :**

  Training texts: ["This is a good cat", "This is a bad day"] → vocabulary: [this, cat, day, is, good, a, bad].New text: "This day is a good day" → [1, 0, 2, 1, 1, 1, 0]
- As we can see, the values for "cat" and "bad" are 0 because these words don't appear in the original text.
- Using BOW is making the assumption that the more a word appears in a text, the more it is representative of its meaning. Therefore, we assume that given a set of positive and negative text, a good classifier will be able to detect patterns in word distributions and learn to predict the sentiment of a text based on which words occur and how many times they do.

6.5 Summarization

- A summary is a reductive transformation of a source text into a summary text by extraction or generation.
- The goal of summarization is to produce a shorter version of a source text by preserving the meaning and the key contents of the original document. A well written summary can significantly reduce the amount of work needed to digest large amounts of text.
- Types of Text summarization There are two types summaries

1. Extractive summaries
2. Abstractive summaries

1. Extractive summaries

Extractive summaries are created by reusing portions (words, sentences, etc.) of the input text document. The system extracts text from the entire collection, without modifying the text document. Most of the summarization research today is on extractive summarization.

2. Abstractive summaries

Requires deep understanding and reasoning over the text. It Provides own summary over input text without using same word or sentence in the input text. Determines the actual and short meaning of each element, such as words, sentences and paragraphs.

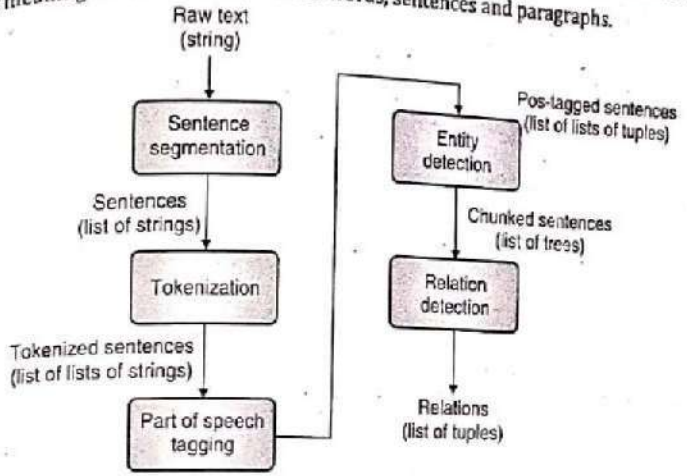


Fig. 6.5.1 : Pipeline architecture of an information extraction process

- Linguistic Pre-processing for Automatic Summarization
- Sentence Segmentation
  - Converts raw text into sentences ¬ List of strings ¬ Sentence tokenizer
  - **Input Text :** John owns a car. It is a Toyota.
  - **Output :** Segm1: John owns a car.

    Segm2 : It is a Toyota.
- Tokenization
  - Identifies the word tokens from given sentence ¬ Provides a list of tokens as output ¬ Word tokenizer
  - **Input :** John owns a car.
  - **Output :** [[John], [owns], [a], [car], [.]]
- **Part Of Speech(POS) tagging (POS Tagging)**
  - Assigns appropriate part of speech tag to each word ¬ POS is useful in extraction of nouns, adverbs, adjective, which provide some meaningful information about text ¬ Generates a list of tuples with POS annotation
  - **Input :** [[John], [owns], [a], [car], [.]]

- **Output :** (NP (NNP John)) (VP (VBZ owns) (NP (DT a) (NN car))) (. .)

- **Entity detection**
  - Identification of predefined categories such as person, location, quantities, organizations etc ¬ NER provides the entity detection for linguistic processing ¬ NER system uses linguistic grammar-based techniques and also statistical model to identify the entity
  - **Input :** (NP (NNP John)) (VP (VBZ owns) (NP (DT a) (NN car))) (. .)
  - **Output :** John->Person

- **Relation detection**
  - Identifies the possible relation between two or more chunked sentences ¬ Co-reference chain provides a relation between two or more sentences ¬ Provides the link between pronouns and its corresponding nouns ¬ Replacement of the pronouns with proper nouns
  - **Input Text :** John owns a car. It is a Toyota. (In form of parse tree)
  - **Output :** "a car" -> "a Toyota"; "It" -> "a Toyota"

6.6 Sentiment Analyses

- Sentiment analysis is the process of detecting positive or negative sentiment in text.
- It's often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers.
- Since customers express their thoughts and feelings more openly than ever before, sentiment analysis is becoming an essential tool to monitor and understand that sentiment.
- Automatically analysing customer feedback, such as opinions in survey responses and social media conversations, allows brands to learn what makes customers happy or frustrated, so that they can tailor products and services to meet their customers' needs.
- For example, using sentiment analysis to automatically analyse 4,000+ reviews about your product could help you discover if customers are happy about your pricing plans and customer service.

Types of Sentiment Analysis

- Sentiment analysis, otherwise known as opinion mining, works thanks to natural language processing (NLP) and machine learning algorithms, to automatically determine the emotional tone behind online conversations.

- There are different algorithms you can implement in sentiment analysis models, depending on how much data you need to analyse, and how accurate you need your model to be. We'll go over some of these in more detail, below.
- **Sentiment analysis algorithms fall into one of three buckets:**
  - **Rule-based :** these systems automatically perform sentiment analysis based on a set of manually crafted rules.
  - **Automatic :** systems rely on machine learning techniques to learn from data.
  - **Hybrid :** systems combine both rule-based and automatic approaches.

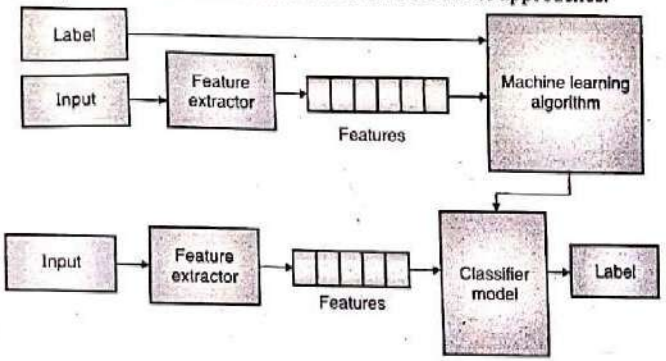


Fig .6.6.1 :Training and Classification Sentiment analysis

The applications of sentiment analysis are endless and can be applied to any industry, from finance and retail to hospitality and technology. Below, we've listed some of the most popular ways that sentiment analysis is being used in business:

1.	Social Media Monitoring	2.	Brand Monitoring
3.	Voice of Customer (VoC)	4.	Customer Service
5.	Market Research		

1. Social Media Monitoring

- Sentiment analysis is used in social media monitoring, allowing businesses to gain insights about how customers feel about certain topics, and detect urgent issues in real time before they spiral out of control.
- On the fateful evening of April 9th, 2017, United Airlines forcibly removed a passenger from an overbooked flight.
- This is exactly the kind of PR catastrophe you can avoid with sentiment analysis.

- It's an example of why it's important to care, not only about if people are talking about your brand, but how they're talking about it. More mentions don't equal positive mentions.
- Brands of all shapes and sizes have meaningful interactions with customers, leads, even their competition, all across social media.
- By monitoring these conversations you can understand customer sentiment in real time and over time, so you can detect disgruntled customers immediately and respond as soon as possible.

2. Brand Monitoring

- Brand monitoring offers a wealth of insights from conversations happening about your brand from all over the internet.
- Analyze news articles, blogs, forums, and more to guage brand sentiment, and target certain demographics or regions, as desired. Automatically categorize the urgency of all brand mentions and route them instantly to designated team members.
- Get an understanding of customer feelings and opinions, beyond mere numbers and statistics. Understand how your brand image evolves over time, and compare it to that of your competition. You can tune into a specific point in time to follow product releases, marketing campaigns, IPO filings, etc., and compare them to past events.
- Real-time sentiment analysis allows you to identify potential PR crises and take immediate action before they become serious issues. Or identify positive comments and respond directly, to use them to your benefit.
- Example : Expedia Canada
- Sentiment analysis allows you to automatically monitor all chatter around your brand and detect and address this type of potentially-explosive scenario while you still have time to defuse it.

3. Voice of Customer (VoC)

- Net Promoter Score (NPS) surveys are one of the most popular ways for businesses to gain feedback with the simple question: Would you recommend this company, product, and/or service to a friend or family member? These result in a single score on a number scale.
- Businesses use these scores to identify customers as promoters, passives, or detractors. The goal is to identify overall customer experience, and find ways to elevate all customers to "promoter" level, where they, theoretically, will buy more, stay longer, and refer other customers.

- Open-ended survey responses were previously much more difficult to analyze, but with sentiment analysis these texts can be classified into positive and negative (and everywhere in between) offering further insights into the Voice of Customer (VoC).
- Sentiment analysis can be used on any kind of survey – quantitative and qualitative – and on customer support interactions, to understand the emotions and opinions of your customers. Tracking customer sentiment over time adds depth to help understand why NPS scores or sentiment toward individual aspects of your business may have changed.

  Example: McKinsey City Voices project

4. Customer Service

- Analyze customer support interactions to ensure the employees are following appropriate protocol. Increase efficiency, so customers aren't left waiting for support. Decrease churn rates; after all it's less hassle to keep customers than acquire new ones.

5. Market Research

- Sentiment analysis empowers all kinds of market research and competitive analysis. Whether you're exploring a new market, anticipating future trends, or seeking an edge on the competition, sentiment analysis can make all the difference.
- You can analyze online reviews of your products and compare them to your competition. Maybe your competitor released a new product that landed as a flop. Find out what aspects of the product performed most negatively and use it to your advantage.

6.7 Named Entity Recognition (NER)

- Named entity recognition (NER) also called entity identification or entity extraction is a natural language processing (NLP) technique that automatically identifies named entities in a text and classifies them into predefined categories. Entities can be names of people, organizations, locations, times, quantities, monetary values, percentages, and more.

Ousted WeWork founder Adam Neumann lists his Manhattan penthouse for $3.75 million

[organization] [person] [location] [Monetary value]

- With named entity recognition, you can extract key information to understand what a text is about, or merely use it to collect important information to store in a database. In this guide, we'll explore how named entity recognition works, its applications in business, and how to perform entity extraction using no-code tools.

6.7.1 Types of Named Entity Recognition

- The Named entity hierarchy is divided into three major classes Entity, Name, Time and Numerical expressions.

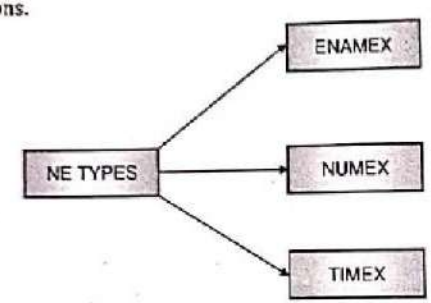


Fig. 6.7.1

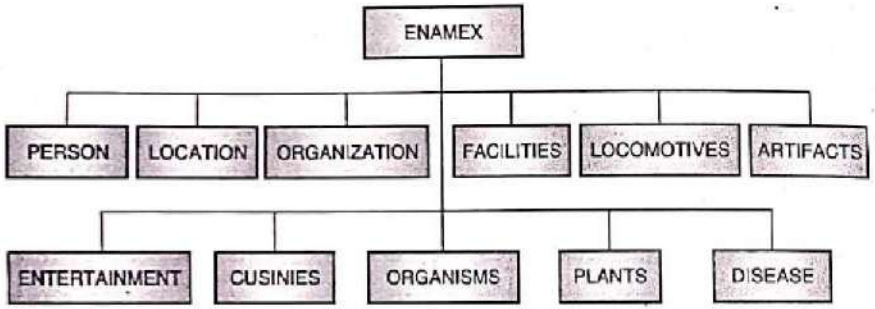


Fig. 6.7.2

1. Entity Name Types

- Persons are entities limited to humans. A person may be a single individual or a group. Individual refer to names of each individual person. Group refers to set of individuals
- Location entities are limited to geographical entities such as geographical areas like names of countries, cities, continents and landmasses, bodies of water, and geological formations.
- Organization entities are limited to corporations, agencies, and other groups of people defined by an established organizational structure

2. Numerical Expressions

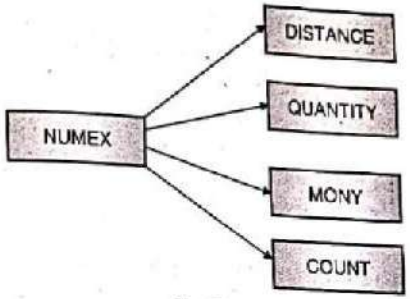


Fig. 6.7.3

3. Time Expressions

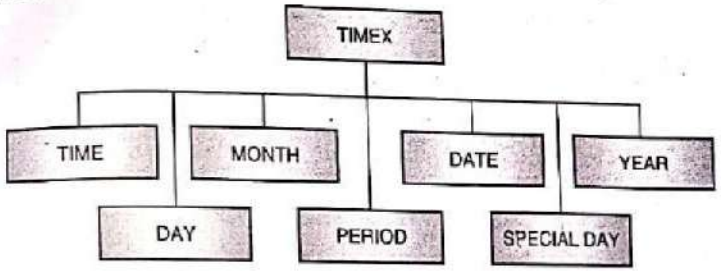


Fig. 6.7.4

6.7.2 Challenges in Named Entity Recognition

- Indian languages belong to several language families, the major ones being the Indo-European languages, Indo-Aryan and the Dravidian languages.
- The challenges in NER arise due to several factors. Some of the main factors are listed below

1. Morphologically rich

- Most of the Indian languages are morphologically rich and agglutinative
- There will be lot of variations in word forms which make machine learning difficult.

2. No Capitalization feature .

- In English, capitalization is one of the main features, whereas that's not there in Indian languages
- Machine learning algorithms have to identify different features.

3. **Ambiguity**
   - Ambiguity between common and proper nouns.
   - Eg: common words such as "Roja" meaning Rose flower is a name of a person

4 **Spell variations**
   - One of the major challenges in the web data is that we find different people spell the same entity with differently.

5 **Less Resources**
   - Most of the Indian languages are less resource languages.
   - Either there are no automated tools available to perform pre-processing tasks required for NER such as Part-of-speech tagging, chunking.
   - Or for languages where such tools are available, they have less performance.

6. **Lack of easy availability of annotated data**
   - there are isolated efforts in the development of NER systems for Indian languages,
   - there is no easy availability and access for NE annotated corpus in the community

Review Questions

Q.1 Write a short note on: Machine Translation.

Q.2 Explain different types of Machine Translation.

Q.3 Difference between Rule-Based MT vs. Statistical MT.

Q.4 What is Question answers system in NLP.

Q.5 Write a short note on: categorization, summarization, sentiment analysis, Named Entity Recognition.

□□□