Vyasapuri, Bandlaguda, Post:Keshavgiri
Hyderabad-500005,Telangana, India
Tel:040-29880079, 86,8978380692, 9642703342
9652216001, 9550544411, Website:www.mist.ac.in
Email:principal@mist.ac.in
principal.mahaveer@gmail.com
Counseling code:MHVR, University Code:E3

MAHAVEER
INSTITUTE OF SCIENCE & TECHNOLOGY
(AN UGC AUTONOMOUS INSTITUTION)
Approved by AICTE, Affiliated to JNTU,Hyderabad
Accredited by NAAC with 'A' Grade
Recognized Under 2(f) of UGC Act 1956,ISO 9001:2015 Certified

ESTD : 2001

Department of Computer Science and Engineering (AIML)

(R18)
Natural Language Processing
Lecture Notes

B. Tech III YEAR – II SEM

Prepared by

Mrs.Swapna
(Professor&HOD-CSM)
Dept. CSE(AIML)

Syllabus

NATURAL LANGUAGE PROCESSING

B.Tech. III Year II Sem.

L	T	P	C
3	1	0	4

Prerequisites: Data structures, finite automata and probability theory

Course Objectives:

- Introduce to some of the problems and solutions of NLP and their relation to linguistics and statistics.

Course Outcomes:

- Show sensitivity to linguistic phenomena and an ability to model them with formal grammars.
- Understand and carry out proper experimental methodology for training and evaluating empirical NLP systems
- Able to manipulate probabilities, construct statistical models over strings and trees, and estimate parameters using supervised and unsupervised training methods.
- Able to design, implement, and analyze NLP algorithms
- Able to design different language modeling Techniques.

UNIT - I

Finding the Structure of Words: Words and Their Components, Issues and Challenges, Morphological Models

Finding the Structure of Documents: Introduction, Methods, Complexity of the Approaches, Performances of the Approaches

UNIT - II

Syntax Analysis: Parsing Natural Language, Treebanks: A Data-Driven Approach to Syntax, Representation of Syntactic Structure, Parsing Algorithms, Models for Ambiguity Resolution in Parsing, Multilingual Issues

UNIT - III

Semantic Parsing: Introduction, Semantic Interpretation, System Paradigms, Word Sense Systems, Software.

UNIT - IV

Predicate-Argument Structure, Meaning Representation Systems, Software.

UNIT - V

Discourse Processing: Cohension, Reference Resolution, Discourse Cohension and Structure **Language Modeling:** Introduction, N-Gram Models, Language Model Evaluation, Parameter Estimation, Language Model Adaptation, Types of Language Models, Language-Specific Modeling Problems, Multilingual and Crosslingual Language Modeling

TEXT BOOKS:

1. Multilingual natural Language Processing Applications: From Theory to Practice – Daniel M. Bikel and Imed Zitouni, Pearson Publication

Natural Language Processing and Information Retrieval: Tanvier Siddiqui, U.S. Tiwary

Unit 1

Unit-1

UNIT - I Finding the Structure of Words: Words and Their Components, Issues and Challenges, Morphological Models **Finding the Structure ofDocuments**: Introduction, Methods, Complexity of the Approaches, Performances of the Approaches

NLP INTRODUCTION:

Natural Language Processing (NLP) refers to AI method of communicating with an intelligent systems using a natural language such as English.

Processing of Natural Language is required when you want an intelligent system like robot to perform as per your instructions, when you want to hear decision from a dialogue based etc.

The field of NLP involves making computers to perform useful tasks with the natural languages humans use. The input and output of an NLP system can be −

- Speech
- Written Text

Components of NLP

There are two components of NLP as given −

Natural Language Understanding (NLU)

Understanding involves the following tasks −

- Mapping the given input in natural language into useful representations.
- Analyzing different aspects of the language.

Natural Language Generation (NLG)

It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation.

It involves −

- **Text planning** − It includes retrieving the relevant content from knowledge base.
- **Sentence planning** − It includes choosing required words, forming meaningful phrases, setting tone of the sentence.
- **Text Realization** − It is mapping sentence plan into sentence structure.

Words and Their components

The **general objective** of an Information Retrieval Systemis to minimize the overhead of a user locating needed information.

Overhead can be expressed as the time a user spends in all of the steps leading to reading an item containing the needed information (e.g., query generation, queryexecution,scanningresultsofquerytoselectitemsto read,readingnon-relevantitems).

The two major measures commonly associated with

Precision and **recall**.

When a user decides to issuea search looking for information on a topic,the total database is logically

Divided into four segments

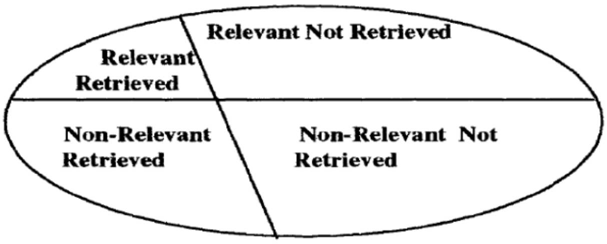


Figure 1.1 Effects of Search on Total Document Space

Relevant items are those documents that contain information that helps the searcher in answering his question.

Non-relevant items are those items that do not provide any directly useful information.

There are two possibilities with respect to each item: it can be retrieved or notretrieved by the user'squery.

$$\text{Precision} = \frac{Number\_Retrieved\_Relevant}{Number\_Total\_Retrieved}$$

$$\text{Recall} = \frac{Number\_Retrieved\_Relevant}{Number\_Possible\_Relevant}$$

Where:

Number_Possible_Relevantarethe

Number of relevant items in the database.

*Number Total Relevant*is the total number of items retrieved from the query.

*Number_Retrieved_Relevant*is the number of items retrieved hat are relevant to the user's search need.

Precision measures one aspect of information retrieval overhead for a user associated with a particular search.

If a search has a 85 per cent precision,then 15 per cent of the user effort is overhead reviewing non-relevant items.

Recall gauges how well a system processing a particular query is able to retrieve the relevant items Functional Overview

A total Information Storage and Retrieval System is composed of four major functional processes:

1) Item Normalization

2) Selective Dissemination of Information (i.e., “Mail”)

3) Archival Document Database Search, and an Index

4) Database Search along with the Automatic File Build process that supportsIndexFiles.

1) Item Normalization:

The first step in any integrated system is to normalize the incoming items to a standard format. Item normalization provides logical restructuring of the item. Additional operations during item normalization are needed to create a searchable data structure: identification of processing tokens (e.g., words), characterization of the tokens, and stemming (e.g., removing word endings) of the tokens.

The processing tokens and their characterization are used to define the searchable text from the total received text. Figure 1.5 shows the normalization process. Standardizing the input takes the different external formats of input data and performs the translation to the formats acceptable to the system. A system may have a single format for all items or allow multiple formats. One example of standardization could be translation of foreign languages into Unicode. Every language has a different internal binary encoding for the characters in the language. One standard encoding that covers English, French, Spanish, etc. is ISO-Latin.

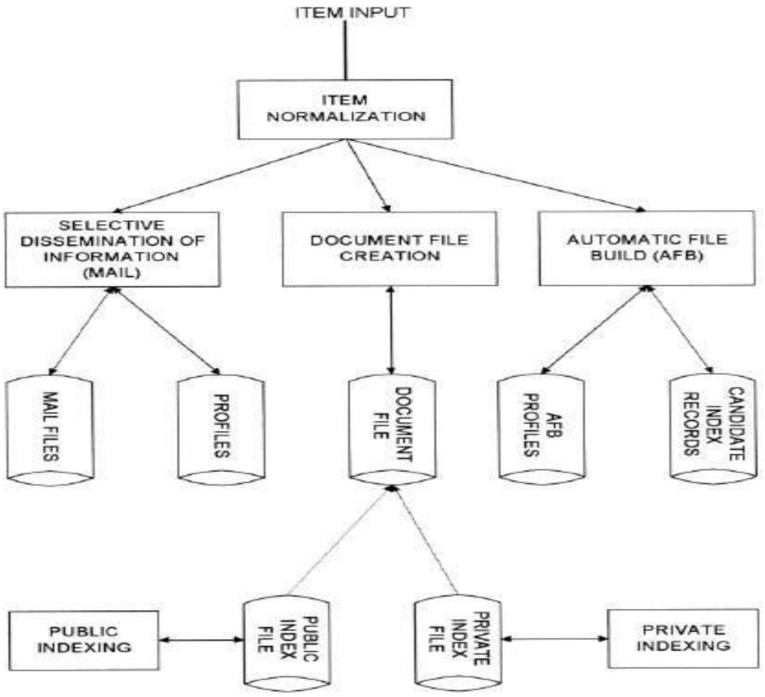


Figure 1.4 Total Information Retrieval System

To assist the users in generating indexes, especially the professional indexers, the system provides a process called *Automatic File Build(AFB)*.

Multi-media adds an extra dimension to the normalization process. In addition to normalizing the textual input, the multi-media input also needs to be standardized. There are a lot of options to the standards being applied to the normalization. If the input is video the likely digital standards will be either MPEG-2, MPEG-1, AVI or Real Media. MPEG (Motion Picture Expert Group) standards are the most universal standards for higher quality video where Real Media is the most common standard for lower quality video being used on the Internet. Audio standards are typically WAV or Real Media (Real Audio). Images vary from JPEG to BMP.

The next process is to parse the item into logical sub-divisions that have meaning to the user. This process, called "Zoning," is visible to the user and used to increase the precision of a search and optimize the display. A typical item is sub-divided into zones, which may overlap and can be hierarchical, such as Title, Author, Abstract, Main Text, Conclusion, and References. The zoning information is passed to the processing token identification operation to store the information, allowing searches to be restricted to a specific zone. For example, if the user is interested in articles discussing "Einstein" then the search should not include the Bibliography, which could include references to articles written by "Einstein."

Systems determine words by dividing input symbols into 3 classes:

1) Valid word symbols

2) Inter-word symbols

3) Special processing symbols.

A word is defined as a contiguous set of word symbols bounded by inter-word symbols. In many systems inter-word symbols are non-searchable and should be carefully selected. Examples of word symbols are alphabetic characters and numbers. Examples of possible inter-word symbols are blanks, periods and semicolons. The exact definition of an inter-word symbol is dependent upon the aspects of the language domain of the items to be processed by the system. For example, an apostrophe may be of little importance if only used for the possessive case in English,

but might be critical to represent foreign names in the database.

Next, a *Stop List/Algorithm* is applied to the list of potential processing tokens. The objective of the Stop function is to save system resources by eliminating from the set of searchable processing tokens those that have little value to the system. Given the significant increase in available cheap memory, storage and processing power, the need to apply the Stop function to processing tokens is decreasing.

Examples of Stop algorithms are: Stop all numbers greater than "999999" (this was selected to allow dates to be searchable) Stop any processing token that has numbers and characters intermixed

2) Selective Dissemination (Distribution, Spreading) of Information

The Selective Dissemination of Information (Mail) Process provides the capability to dynamically compare newly received items in the information system against standing statements of interest of users and deliver the item to those users whose statement of interest matches the contents of the item. The Mail process is composed of the search process, user statements of interest (Profiles) and user mail files. As each item is received, it is processed against every user's profile. A profile contains a typically broad search statement along with a list of user mail files that will receive the document if the search statement in the profile is satisfied. Selective Dissemination of Information has not yet been applied to multimedia sources.

3) Document Database Search

The Document Database Search Process provides the capability for a query to search against all items received by the system. The Document Database Search process is composed of the search process, user entered queries (typically ad hoc queries) and the document database which contains all items that have been received, processed and stored by the system. Typically items in the Document Database do not change (i.e., are not edited) once received.

Index Database Search

When an item is determined to be of interest, a user may want to save it for future reference. This is in effect filing it. In an information system this is accomplished via

the index process. In this process the user can logically store an item in a file along with additional index terms and descriptive text the user wants to associate with the item. The Index Database Search Process (see Figure 1.4) provides the capability to create indexes and search them.

There are 2 classes of index files:

1) Public Index files

2) Private Index files

Every user can have one or more Private Index files leading to a very large number of files. Each Private Index file references only a small subset of the total number of items in the Document Database. Public Index files are maintained by professional library services personnel and typically index every item in the Document Database. There is a small number of Public Index files. These files have access lists (i.e., lists of users and their privileges) that allow anyone to search or retrieve data. Private Index files typically have very limited access lists. To assist the users in generating indexes, especially the professional indexers, the system provides a process *called Automatic File Build* shown in Figure 1.4 (also called Information Extraction).

Multimedia Database Search

From a system perspective, the multi-media data is not logically its own data structure, but an augmentation to the existing structures in the Information Retrieval System.

Relationship to Database Management Systems

From a practical standpoint, the integration of DBMS's and Information Retrieval Systems is very important. Commercial database companies have already integrated the two types of systems. One of the first commercial databases to integrate the two systems into a single view is the INQUIRE DBMS. This has been available for over fifteen years. A more current example is the ORACLE DBMS that now offers an imbedded capability called CONVECTIS, which is an informational retrieval system that uses a comprehensive thesaurus which provides the basis to generate "themes" for a particular item. The INFORMIX DBMS has the ability to link to RetrievalWare to provide integration of structured data and information along with functions associated with Information Retrieval Systems.

that the user is interested in seeing.

Digital Libraries and Data Warehouses (DataMarts)

As the Internet continued its exponential growth and project funding became available, the topic of Digital Libraries has grown. By 1995 enough research and pilot efforts had started to support the 1ST ACM International Conference on Digital Libraries (Fox-96). Indexing is one of the critical disciplines in library science and significant effort has gone into the establishment of indexing and cataloging standards. Migration of many of the library products to a digital format introduces both opportunities and challenges. Information Storage and Retrieval technology has addressed a small subset of the issues associated with Digital Libraries.

Data warehouses are similar to information storage and retrieval systems in that they both have a need for search and retrieval of information. But a data warehouse is more focused on structured data and decision support technologies. In addition to the normal search process, a complete system provides a flexible set of analytical tools to “mine” the data. Data mining (originally called Knowledge Discovery in Databases - KDD) is a search process that automatically analyzes data and extract relationships and dependencies that were not part of the database design.

Information Retrieval System Capabilities

Search Capabilities

Browse Capabilities

Miscellaneous Capabilities

Standards

The search capabilities address both Boolean and Natural Language queries. The algorithms used for searching are called Boolean, natural language processing and probabilistic. Probabilistic algorithms use frequency of occurrence of processing tokens (words) in determining similarities between queries and items and also in predictors on the potential relevance of the found item to the searcher.

The newer systems such as TOPIC, RetrievalWare, and INQUERY all allow for natural language queries.

Browse functions to assist the user in filtering the search results to find relevant information are very important.

2.1 Search Capabilities

The objective of the search capability is to allow for a mapping between a user's specified need and the items in the information database that will answer that need. It can consist of natural language text in composition style and/or query terms (referred to as terms in this book) with Boolean logic indicators between them. One concept that has occasionally been implemented in commercial systems (e.g., RetrievalWare), and holds significant potential for assisting in the location and ranking of relevant items, is the "weighting" of search terms. This would allow a user to indicate the importance of

search terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

Boolean Logic

Boolean logic allows a user to logically relate multiple concepts together to define what information is needed. Typically the Boolean functions apply to processing tokens identified anywhere within an item. The typical Boolean operators are **AND, OR,** and **NOT**. These operations are implemented using set intersection, set union and set difference procedures. Asearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

the search statement may apply to the complete item or contain additional paramesearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular

search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

few systems introduced the concept of "exclusive or" but it is equivalent to a slightly more complex query using the other operators and is not generally useful to users since most users do not understand it.

A special type of Boolean search is called "M of N" logic. The user lists a set of possible search terms and identifies, as acceptable, any item that contains a subset of the terms. For example, "Find any item containing any two of the following terms:

"AA," "BB," "CC." This can be expanded into a Boolean search that performs an AND between all combinations of two terms and "OR"s the results together ((AA AND BB) or (AA AND CC) or (BB AND CC)).

Proximity

Proximity is used to restrict the distance allowed within an item between two search terms. The semantic concept is that the clossearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between

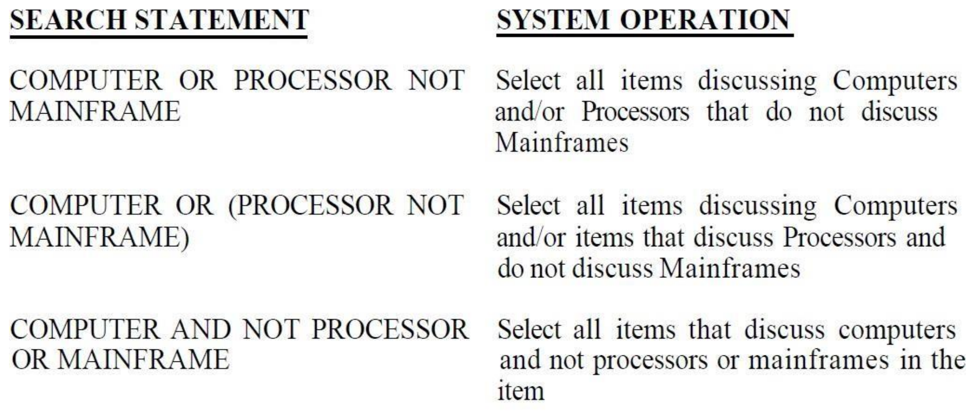
0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

two terms are found in a text the more likely they are related in the description of a particular concept. Proximity is used to increase the precision of a search. If the terms COMPUTER and DESIGN are found within a few words of each other then the item is more likely to be discussing the design of computers than if the words are paragraphs apart. The typical format for proximity is:

TERM1 within "m" "units" of TERM2

The distance operator “m” is an integer number and units are in Characters, Words, Sentences, or Paragraphs.



SEARCH STATEMENT	SYSTEM OPERATION
COMPUTER OR PROCESSOR NOT MAINFRAME	Select all items discussing Computers and/or Processors that do not discuss Mainframes
COMPUTER OR (PROCESSOR NOT MAINFRAME)	Select all items discussing Computers and/or items that discuss Processors and do not discuss Mainframes
COMPUTER AND NOT PROCESSOR OR MAINFRAME	Select all items that discuss computers and not processors or mainframes in the item

Figure 2.1 Use of Boolean Operators

A special case of the Proximity operator is the Adjacent (ADJ) operator that normally has a distance operator of one and a forward only direction (i.e., in WAIS). Another special case is where the distance is set to zero meaning within the same semantic unit.

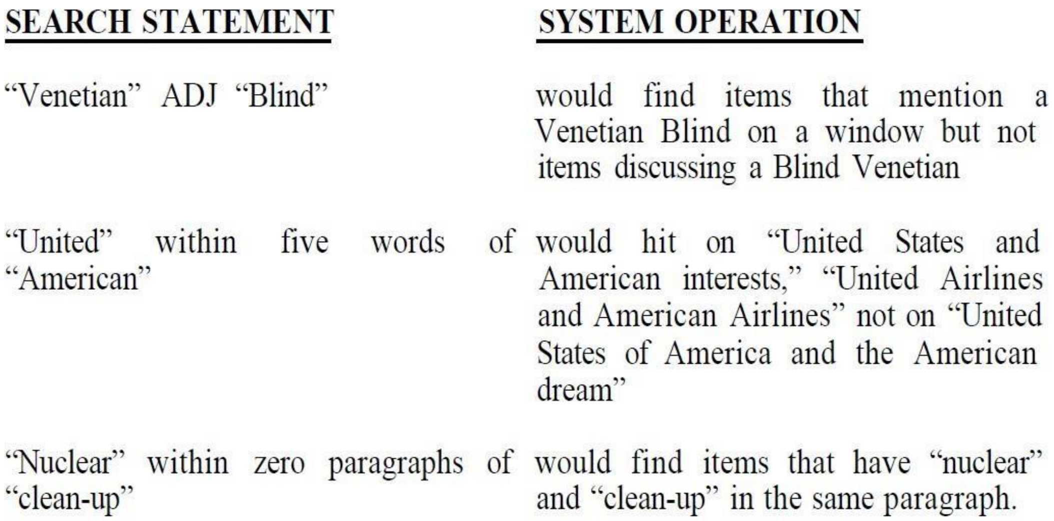
contiguous Word Phrases

A Contiguous Word Phrase (CWP) is both a way of specifying a query term and a special search operator. A Contiguous Word Phrase is two or more words that are treated as a single semantic unit. An example of a CWP is “United States of America.” It is four words that specify a search term representing a single specific semantic concept (a country) that can be used with any of the operators discussed above. Thus a query could specify “manufacturing” AND “United States of America” which returns any item that contains the word “manufacturing” and the contiguous words “United States of America.”

A contiguous word phrase also acts like a special search operator that is similar to the proximity (Adjacency) operator but allows for additional specificity. If two

terms are specified, the contiguous word phrase and the proximity operator using directional one word parameters or the Adjacent operator are identical. For contiguous word phrases of more than two terms the only way of creating an equivalent search statement using proximity and Boolean operators is via nested Adjacencies which are not found in most commercial systems. This is because Proximity and Boolean operators are binary operators but contiguous word phrases are an “N”ary operator where “N” is the number of words in the CWP.

Contiguous Word Phrases are called Literal Strings in WAIS and Exact Phrases in RetrievalWare. In WAIS multiple Adjacency (ADJ) operators are used to define a Literal String (e.g., “United” ADJ “States” ADJ “of” ADJ “America”).



SEARCH STATEMENT	SYSTEM OPERATION
“Venetian” ADJ “Blind”	would find items that mention a Venetian Blind on a window but not items discussing a Blind Venetian
“United” within five words of “American”	would hit on “United States and American interests,” “United Airlines and American Airlines” not on “United States of America and the American dream”
“Nuclear” within zero paragraphs of “clean-up”	would find items that have “nuclear” and “clean-up” in the same paragraph.

Figure 2.2 Use of Proximity

Fuzzy Searches

Fuzzy Searches provide the capability to locate spellings of words that are similar to the entered search term. This function is primarily used to compensate for errors in spelling of words. Fuzzy searching increases recall at the expense of decreasing precision (i.e., it can erroneously identify terms as the search term). In the process of expanding a query term fuzzy searching includes other terms that have similar spellings, giving more weight (in systems that rank output) to words in the database that have similar word lengths and position of the characters as the entered term. A Fuzzy Search on the term “computer” would automatically include the following words from the information database: “computer,” “compiter,” “conputer,” “computter,” “compute.”

Term Masking

Term masking is the ability to expand a query term by masking a portion of the term and accepting as valid any processing token that maps to the unmasked portion of the term. The value of term masking is much higher in systems that do not perform stemming or only provide a very simple stemming algorithm. There are two types of search term masking: fixed length and variable length. Sometimes they are called fixed and variable length “don’t care” functions.

Fixed length masking is a single position mask. It masks out any symbol in a particular position or the lack of that position in a word. Variable length "don't cares" allows masking of any number of characters within a processing token. The masking may be in the front, at the end, at both front and end, or imbedded. The first three of these cases are called suffix search, prefix search and imbedded character string search, respectively. The use of an imbedded variable length don't care is seldom used.Figure 2.3 provides examples of the use of variable length term masking. If "*" represents a variable length don't care then the following are examples of its use: "*COMPUTER" Suffix Search

"COMPUTER*" Prefix Search

"*COMPUTER*" Imbedded String Search

SEARCH STATEMENT	SYSTEM OPERATION
multi$national	Matches“multi-national,” “multinational,” “multinational” but does not match “multi national” since it is two processing tokens.
computer	Matches,“minicomputer” “microcomputer” or “computer”
comput*	Matches “computers,” “computing,” “computes”
comput	Matches “microcomputers” , “minicomputing,” “compute”

Figure 2.3 Term Masking

Numeric and Date Ranges

Term masking is useful when applied to words, but does not work for finding ranges of numbers or numeric dates. To find numbers larger than “125,” using a term “125*” will not find any number except those that begin with the digits “125.”

Concept/Thesaurus Expansion

Associated with both Boolean and Natural Language Queries is the ability to expand the search terms via Thesaurus or Concept Class database reference tool. A Thesaurus is typically a one-level or two-level expansion of a term to other terms that are similar in meaning. A Concept Class is a tree structure that expands each meaning of a word into potential concepts that are related to the initial term (e.g., in the TOPIC system). Concept classes are sometimes implemented as a network structure that links word stems (e.g., in the RetrievalWare system). An example of Thesaurus and Concept Class structures are shown in Figure 2.4 (Thesaurus-93) and Figure 2.5.

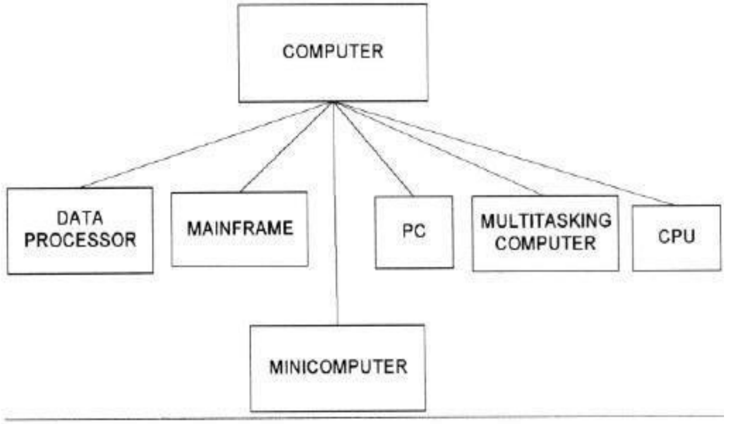


Figure 2.4 Thesaurus for term "computer"

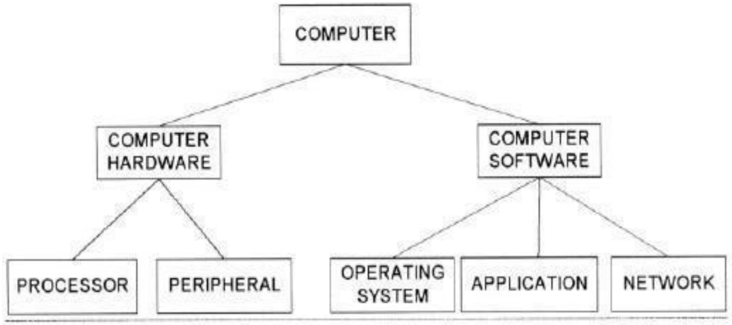


Figure 2.5 Hierarchical Concept Class Structure for "Computer"

Thesauri are either semantic or based upon statistics. A semantic thesaurus is alisting of words and then other words that are semantically similar.

The problem with thesauri is that they are generic to a language and can introduce many search terms that are not found in the document database. An alternative uses the database or a representative sample of it to create statistically related terms. It is conceptually a thesaurus in that words that are statistically related to other words by their frequently occurring together in the same items. This type of thesaurus is very dependent upon the database being searched and may not be portable to other databases.

Natural Language Queries

Natural language interfaces improve the recall of systems with a decrease in precision when negation is required.

Browse Capabilities

Once the search is complete, Browse capabilities provide the user with the capability to determine which items are of interest and select those to be displayed. There are two ways of displaying a summary of the items that are associated with a query: line item status and data visualization. From these summary displays, the user can select the specific items and zones within the items for display.

Ranking

Typically relevance scores are normalized to a value between 0.0 and 1.0. The highest value of 1.0 is interpreted that the system is sure that the item is relevant to the search statement. In addition to ranking based upon the characteristics of the item and the database, in many circumstances collaborative filtering is providing an option forselecting and ordering output.

Collaborative filtering has been very successful in sites such as AMAZON.COM MovieFinder.com, and CDNow.com in deciding what products to display to users based upon their queries.

Rather than limiting the number of items that can be assessed by the number of lines on a screen, other graphical visualization techniques showing the relevance

relationships of the hit items can be used. For example, a two or three dimensional graph can be displayed where points on the graph represent items and the location of the points represent their relative relationship between each other and the user's query. In some cases color is also used in this representation. This technique allows a user to see the clustering of items by topics and browse through a cluster or move to another topical cluster.

Zoning

Related to zoning for use in minimizing what an end user needs to review from a hit item is the idea of locality and passage based search and retrieval.

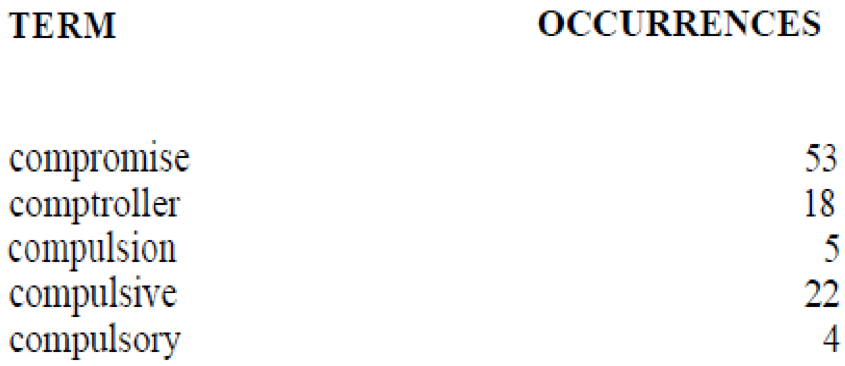
Highlighting

Most systems allow the display of an item to begin with the first highlight within the item and allow subsequent jumping to the next highlight. The DCARS system that acts as a user frontend to the Retrieval Ware search system allows the user to browse an item in the order of the paragraphs or individual words that contributed most to the rank value associated with the item. The highlighting may vary by introducing colors and intensities to indicate the relative importance of a particular word in the item in the decision to retrieve the item.

Miscellaneous Capabilities2.3.1Vocabulary Browse

Vocabulary Browse provides the capability to display in alphabetical sorted order words from the document database. Logically, all unique words (processing tokens) in the database are kept in sorted order along with a count of the number of unique items in which the word is found. The user can enter a word or word fragment and thesystem will begin to display the dictionary around the entered text.

It helps the user determine the impact of using a fixed or variable length mask on a search term and potential mis-spellings. The user can determine that entering thesearch term “compul*” in effect is searching for “compulsion” or “compulsive” or “compulsory.” It also shows that someone probably entered the word “computen” when they really meant “computer.”



TERM	OCCURRENCES
compromise	53
comptroller	18
compulsion	5
compulsive	22
compulsory	4

Iterative Search and Search History Log

Frequently a search returns a Hit file containing many more items than the user wants to review. Rather than typing in a complete new query, the results of the previous search can be used as a constraining list to create a new query that is applied against it. This has the same effect as taking the original query and adding additional search statement against it in an AND condition. This process of refining the results of a previous search to focus on relevant items is called iterative search. This also applies when a user uses relevance feedback to enhance a previous search. The search history log is the capability to display all the previous searches that were executed during the current session.

Canned Query

The capability to name a query and store it to be retrieved and executed during a later user session is called canned or stored queries. A canned query allows a user to create and refine a search that focuses on the user's general area of interest one time and then retrieve it to add additional search criteria to retrieve data that is currently needed. Canned query features also allow for variables to be inserted into the query and bound to specific values at execution time.

Difficulties in NLP:

Issues and Challenges

NL has an extremely rich form and structure.

It is very ambiguous. There can be different levels of ambiguity −

Lexical ambiguity − It is at very primitive level such as word-level.

For example, treating the word "board" as noun or verb?

Syntax Level ambiguity − A sentence can be parsed in different ways.

For example, "He lifted the beetle with red cap." − Did he use cap to lift the beetle or he lifted a beetle that had red cap?

Referential ambiguity − Referring to something using pronouns. For example, Rima went to Gauri. She said, "I am tired." − Exactly who is tired?

One input can mean different meanings.

Many inputs can mean the same thing.

NLP Terminology

Phonology − It is study of organizing sound systematically.

Morphology − It is a study of construction of words from primitive meaningful units.

Morpheme − It is primitive unit of meaning in a language.

Syntax − It refers to arranging words to make a sentence. It also involves determining the structural role of words in the sentence and in phrases.

Semantics − It is concerned with the meaning of words and how to combine words into meaningful phrases and sentences.

Pragmatics − It deals with using and understanding sentences in different situations and how the interpretation of the sentence is affected.

Discourse − It deals with how the immediately preceding sentence can affect the interpretation of the next sentence.

World Knowledge − It includes the general knowledge about the world.

Natural Language Processing

- Humans communicate through some form of language either by text or speech.
- To make interactions between computers and humans, computers need to understand natural languages used by

  humans.
- Natural language processing is all about making computers learn, understand, analyse, manipulate and interpret

  natural(human) languages.
- NLP stands for **Natural Language Processing**, which is a part of **Computer Science, Humanlanguage,** and **Artificial Intelligence**.
- Processing of Natural Language is required when you want an intelligent system like robot to perform as per yourinstructions, when you want to hear decision from a dialogue based clinical expert system, etc.
- The ability of machines to interpret human language is now at the core of many applications that we use every day

  - chatbots, Email classification and spam filters, search engines, grammar checkers, voice assistants, and sociallanguage translators.
- The input and output of an NLP system can be Speech or Written Text

Components of NLP

- **There are two components of NLP, Natural Language Understanding (NLU) and Natural Language Generation (NLG).**
- Natural Language Understanding (NLU) which involves transforming human language into a machine-readable format.
- It helps the machine to understand and analyse human language by extracting the text from large data such as keywords, emotions, relations, and semantics.
- Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation.
- It mainly involves Text planning, Sentence planning, and Text realization.

NLP Terminology

- **Phonology −** It is study of organizing sound systematically.
- ***Morphology***: The study of the formation and internal structure of words.
- **Morpheme −** It is primitive unit of meaning in a language.

- ***Syntax***: The study of the formation and internal structure of sentences.
- ***Semantics***: The study of the meaning of sentences.
- **Pragmatics** − It deals with using and understanding sentences in different situations

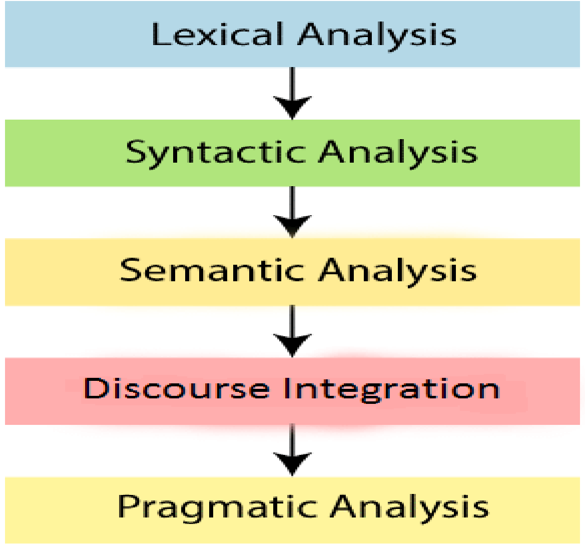
and how the interpretation of the sentence is affected.

- **Discourse** − It deals with how the immediately preceding sentence can affect theinterpretation of the next sentence.
- **World Knowledge** − It includes the general knowledge about the world.

Steps in NLP

- There are general five steps :

1. Lexical Analysis
2. Syntactic Analysis (Parsing)
3. Semantic Analysis
4. Discourse Integration



Lexical Analysis

Syntactic Analysis

Semantic Analysis

Discourse Integration

Pragmatic Analysis

5. Pragmatic Analysis

Lexical Analysis –

- The first phase of NLP is the Lexical Analysis.
- This phase scans the source code as a stream of characters and converts it into meaningful lexemes.
- It divides the whole text into paragraphs, sentences, and words.

Syntactic Analysis (Parsing) –

- Syntactic Analysis is used to check grammar, word arrangements, and shows the relationship among the words.
- The sentence such as “The school goes to boy” is rejected by English syntactic analyzer.

Semantic Analysis –

- Semantic analysis is concerned with the meaning representation.

- It mainly focuses on the literal meaning of words, phrases, and sentences.
- The semantic analyzer disregards sentence such as “hot ice-cream”.

Discourse Integration -

- Discourse Integration depends upon the sentences that proceeds it and also invokes the meaning of the sentences that follow it.

Pragmatic Analysis -

- During this, what was said is re-interpreted on what it actually meant.
- It involves deriving those aspects of language which require real world knowledge.
- **Example:** "Open the door" is interpreted as a request instead of an order.

Finding the Structure of Words

- Human language is a complicated thing.
- We use it to express our thoughts, and through language, we receive information and infer its meaning.
- Trying to understand language all together is not a viable approach.
- The point of **morphology**, for instance, is to study the variable forms and functions of words,
- The syntax is concerned with the arrangement of words into phrases, clauses, and sentences.
- Word structure constraints due to pronunciation are described by **phonology**,
- The conventions for writing constitute the **orthography** of a language.
- The meaning of a linguistic expression is its semantics, and etymology and lexicology cover especially the evolution of words and explain the semantic, morphological, and other links among them.
- Words are perhaps the most intuitive units of language, yet they are in general tricky to define.
- Knowing how to work with them allows, in particular, the development of **syntactic** and

semantic abstractions and simplifies other advanced views on language.

- Here, first we explore how to identify words of distinct types in human languages, and how the internal structure of words can be modelled in connection with the grammatical properties and lexical concepts the words should represent.
- The discovery of word structure is **morphological parsing**.
- In many languages, words are delimited in the orthography by whitespace and punctuation.
- But in many other languages, the writing system leaves it up to the reader to tell words apart or determine their exact phonological forms.

Words and Their Components

- Words are defined in most languages as the smallest linguistic units that can form a complete utterance by themselves.
- The minimal parts of words that deliver aspects of meaning to them are called **morphemes**.

Tokens

- Suppose, for a moment, that words in English are delimited only by whitespace andpunctuation (the marks, such as full stop, comma, and brackets)
- Example: Will you read the newspaper? Will you read it? I won’t read it.
- If we confront our assumption with insights from syntax, we notice two here: words ***newspaper*** and ***won’t***.
- Being a compound word, ***newspaper*** has an interesting **derivational structure**.
- In writing, *newspaper* and the associated concept is distinguished from the isolated *news* and *paper*.
- For reasons of generality, linguists prefer to analyze *won’t* as two syntactic words, or tokens, each of which has its independent role and can be reverted to its normalized form.
- The structure of ***won’t*** could be parsed as ***will*** followed by ***not***.
- In English, this kind of tokenization and **normalization** may apply to just a limited set of

cases, but in other languages, these phenomena have to be treated in a less trivial manner.

- Tokens behaving in this way can be found in various languages and are often called **clitics**.

Lexemes

- By the term word, we often denote not just the one **linguistic form** in the given context but also the **concept behind the form** and the **set of alternative forms** that can express it.

- Such **sets** are called **lexemes or lexical items**, and they constitute the **lexicon** of a

language.

- Lexemes can be divided by their behaviour into the lexical categories of verbs, nouns,

adjectives, conjunctions, particles, or other parts of speech.

- The citation **form of a lexeme**, by which it is commonly identified, is also called its

lemma.

- When we convert a word into its other forms, such as turning the **singular *mouse*** into the **plural *mice* or *mouses***, we say we **inflect** the lexeme.
- When we transform a lexeme into another one that is morphologically related, regardless of its lexical category, we say we derive the lexeme: for instance, thenouns ***receiver* and *reception*** are derived from the verb *to* ***receive***.

- Example: Did you see him? I **didn’t** see him. I didn’t see **anyone**.

- Example presents the problem of tokenization of ***didn't*** and the investigation of the internal structure of ***anyone***.

Morphemes

- Morphological theories differ on whether and how to associate the properties of wordforms with their structural components.
- These components are usually called **segments** or **morphs**.
- The morphs that by themselves represent some aspect of the meaning of a word arecalled **morphemes** of some function.
- Human languages employ a variety of devices by which morphs and morphemes are

  combined into word forms.

Morphology

- Morphology is the domain of linguistics thatanalyses the internal structure of

  words.

- Morphological analysis – exploring the structure of words

- Words are built up of minimal meaningful elements called morphemes:played = play-ed

cats = cat-s

unfriendly = un-friend-ly

- Two types of morphemes: i Stems: play, cat,
friendii Affixes: -ed, -s, un-, -ly

- Two main types of affixes:

i Prefixes precede the stem: un-

ii Suffixes follow the stem: -ed, -s, un-, -ly

- Stemming = find the stem by stripping off affixes

- play = play

replayed = re-play-ed

computerized = comput-er-ize-d

Problems in morphological processing

- Inflectional morphology: inflected forms are constructed from base forms and inflectional affixes.
- Inflection relates different forms of the same word

Lemma	Singular	Plural
cat	cat	cats
dog	dog	dogs
knife	knife	knives
sheep	sheep	sheep
mouse	mouse	mice

- Derivational morphology: words are constructed from roots (or stems) and derivational affixes:

inter+national = international international+ize = internationalize
internationalize+ation = internationalization

- The simplest morphological process concatenates morphs one by one, as in *dis- agree-ment-s*, where *agree* is a free lexical morpheme and the other elements are bound grammatical morphemes contributing some partial meaning to the whole word.
- in a more complex scheme, morphs can interact with each other, and their forms may become subject to additional phonological and orthographic changes denoted as morphophonemic.
- The alternative forms of a morpheme are termed **allomorphs**.

Typology

- Morphological typology divides languages into groups by characterizing the prevalent

morphological phenomena in those languages.

- It can consider various criteria, and during the history of linguistics, different classifications

have been proposed.

- Let us outline the typology that is based on quantitative relations between words, theirmorphemes, and their features:
- **Isolating**, or **analytic**, languages include no or relatively few words that would comprise more

than one morpheme

- **Synthetic** languages can combine more morphemes in one word and are further divided into agglutinative and fusional languages.
- **Agglutinative** languages have morphemes associated with only a single function at a

time (as in Korean, Japanese, Finnish, and Tamil, etc.)

- **Fusional** languages are defined by their feature-per-morpheme ratio higher than one

(as in Arabic, Czech, Latin, Sanskrit, German, etc.).

- In accordance with the notions about word formation processes mentioned earlier, we

can also find out using concatenative and nonlinear:

- **Concatenative** languages linking morphs and morphemes one after another.
- **Nonlinear** languages allowing structural components to merge nonsequentially to

apply tonal morphemes or change the consonantal or vocalic templates of words.

Issues and Challenges

- **Irregularity**: word forms are not described by a prototypical linguistic model.
- **Ambiguity**: word forms be understood in multiple ways out of the context
- **Productivity**: is the inventory of words in a language finite, or is it unlimited?
- Morphological parsing tries to eliminate the variability of word forms to provide higher- level linguistic units whose lexical and morphological properties are explicit and well defined.
- It attempts to remove unnecessary irregularity and give limits to ambiguity, both of which are present inherently in human language.
- By irregularity, we mean existence of such forms and structures that are not described

appropriately by a prototypical linguistic model.

- Some irregularities can be understood by redesigning the model and improving its rules, but other lexically dependent irregularities often cannot be generalized
- Ambiguity is indeterminacy (not being interpreted)in interpretation of expressions of

language.

- Morphological modelling also faces the problem of productivity and creativity in language, bywhich unconventional but perfectly meaningful new words or new senses are coined.

Irregularity

- Morphological parsing is motivated by the quest for generalization and abstraction in theworld of words.

- Immediate descriptions of given linguistic data may not be the ultimate ones, due to either their inadequate accuracy or inappropriate complexity, and better formulations may beneeded.

- The design principles of the morphological model are therefore very important.

- With the proper abstractions made, irregular morphology can be seen as merely enforcing some extended rules, the nature of which is phonological, over the underlying or prototypical regular word forms.

- Morphophonemic templates capture morphological processes by just organizing stem patterns and generic affixes without any context-dependent variation of the affixes or ad hoc modification of the stems.

- The merge rules, indeed very neatly or effectively concise, then ensure that such structured representations can be converted into exactly the surface forms, both orthographic and phonological, used in the natural language.

- Applying the merge rules is independent of and irrespective of any grammatical parametersor information other than that contained in a template.
- Most morphological irregularities are thus successfully removed.

Ambiguity

- Morphological ambiguity is the possibility that word forms be understood in multiple ways out of the context of their discourse (communication in speech or writing).
- Words forms that look the same but have distinct functions or meaning are called homonyms.
- Ambiguity is present in all aspects of morphological processing and language processing at large.

Productivity

- Is the inventory of words in a language finite, or is it unlimited?

- This question leads directly to discerning two fundamental approaches to language, summarized in the distinction between *langue* and *parole*, or in the competence versus performance

- In one view, language can be seen as simply a collection of utterances (parole) actually pronounced or written (performance).

- This ideal data set can in practice be approximated by linguistic corpora, which are finite collections of linguistic data that are studied with empirical(based on) methods and can be used for comparison when linguistic models are developed.

- Yet, if we consider language as a system (langue), we discover in it structural devices like recursion, iteration, or compounding(make up; constitute)that allow to produce (competence) an infinite set of concrete linguistic utterances.

- This general potential holds for morphological processes as well and is called morphological productivity.

- We denote the set of word forms found in a corpus of a language as its vocabulary.
- The members of this set are word types, whereas every original instance of a word form is a word token.
- The distribution of words or other elements of language follows the “80/20 rule frame which is a protocol,” also knownas the law of the vital few.
- It says that most of the word tokens in a given corpus(a collection of written texts)can be identified with just a couple of word types in its vocabulary, and words from the

  rest of the vocabulary occur much less commonly if not rarely in the corpus.
- Furthermore, new, unexpected words will always appear as the collection of linguistic data is

enlarged.

Morphological Models

- There are many possible approaches to designing and implementing morphological models.
- Over time, computational linguistics has witnessed the development of a number of formalisms and frameworks, in particular grammars of different kinds and expressive power, with which to address whole classes of problems in processing natural as well as formal languages.
- Let us now look at the most prominent types of computational approaches to morphology.

Dictionary Lookup

- Morphological parsing is a process by which word forms of a language are associated with

corresponding linguistic descriptions.

- Morphological systems that specify these associations by merely enumerating(is **the act or process of making or stating a list of things one after another)** them case by case do not offer any generalization means.

Most common data structure

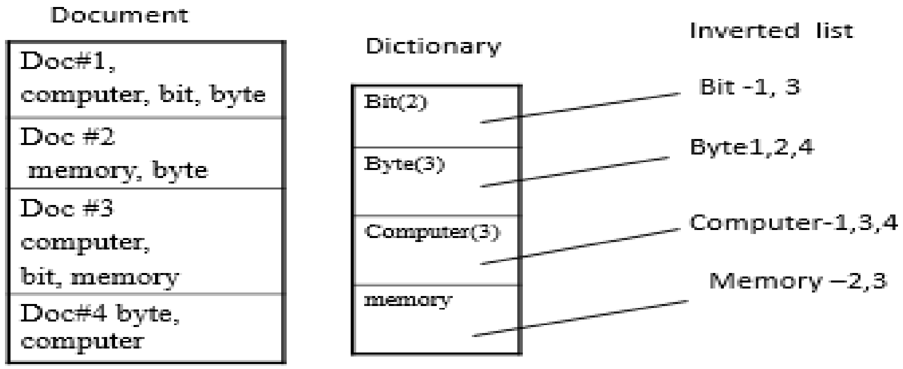
Inverted file structures are composed of three filesThe document file

1. The inversion list (Posting List)
2. Dictionary
3. The inverted file : based on the methodology of storing an inversion ofdocuments.

4. For each word a listof documents in which the word is found isstored(inversion of document
5. Each document is given a unique the numerical identifier that is stored in inversion list . Dictionary is used to located the inversion list for a particular word.

Which is a sorted list(processing tokens) in the system and a pointer to thelocation of its inversion list.

Dictionary can also store other information used in query optimizationsuch as length of inversion lists to increase the precision.



Document

Doc#1, computer, bit, byte
Doc #2 memory, byte
Doc #3 computer, bit, memory
Doc#4 byte, computer

Dictionary

Bit(2)
Byte(3)
Computer(3)
memory

Inverted list

Bit -1, 3

Byte1,2,4

Computer-1,3,4

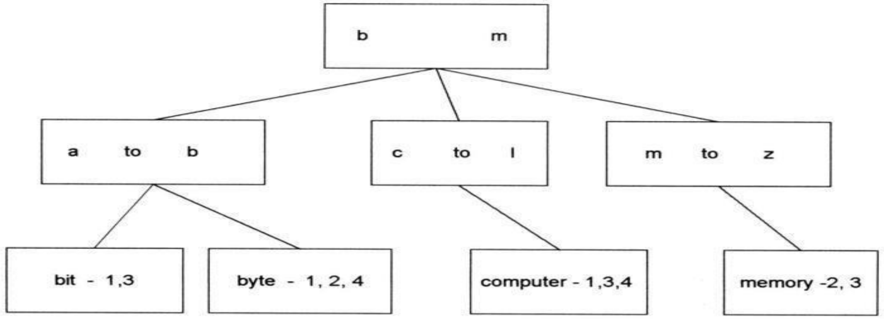
Memory –2,3

- ➢ Use zoning to improve
- ➢ precision and Restrict entries.
- ➢ Inversion list consists of document identifier for each documentin which the word is found.

Ex: bit 1(10),1(12) 1(18) is in 10,12, 18 position of the word bit in the document #1.

- ➢ When a search is performed, the inversion lists for the terms in the query arelocate and appropriate logic is applied between inversion lists.
- ➢ Weights can also be stored in the inversion list.

- Inversion list are used to store concept and their relationship.
- Words with special characteristics can be stored in their own dictionary. Ex: Date… which require date ranging and numbers.
- Systems that support ranking are re-organized in ranked order.
- B trees can also be used for inversion instead of dictionary.
- The inversion lists may be at the leaf level or referenced in higher level pointers.
- A B-tree of order m is defined as:
- A root node with between 2 and 2m keys
- All other internal nodes have between m and 2m keys
- All keys are kept in order from smaller to larger.
- All leaves are at the same level or differ by at most one level.



b m

a to b

c to l

m to z

bit - 1,3

byte - 1, 2, 4

computer - 1,3,4

memory -2, 3

Finite-State Morphology

- By finite-state morphological models, we mean those in which the specifications written by human programmers are directly compiled into finite-states

- The two most popular tools supporting this approach, XFST (Xerox Finite-State Tool),

  LexTools.

- They consist of a finite set of nodes connected by directed edges labeled with pairs of input

  and output symbols.
- In such a network or graph, nodes are also called states, while edges are called arcs.

- Traversing the network from the set of initial states to the set of final states along the arcs is equivalent to reading the sequences of encountered input symbols and writing the sequencesof corresponding output symbols.

- The set of possible sequences accepted by the defines the input language; the set of possible sequences emitted by the defines the output language.

Input	Input Morphological parsed output
Cats	cat +N +PL
Cat	cat +N +SG
Cities	city +N +PL
Geese	goose +N +PL
Goose	goose +N +SG) or (goose +V)
Gooses	goose +V +3SG
merging	merge +V +PRES-PART
Caught	(caught +V +PAST-PART) or (catch +V +PAST)

- matching words in the infinite regular language definedby *grandson*, *great-grandson*, *great-great-grandson*.
- In finite-state computational morphology, it is common to refer to the input word forms as **surface strings** and to the output descriptions as **lexical strings**, if the transducer is used for morphological analysis, or vice versa, if it is

used for morphological generation.

- In English, a finite-state transducer could analyze the surface string children into the lexical string child [+plural], for instance, or generate women from woman [+plural].

- Relations on languages can also be viewed as functions. Let us have a relation *R*, and let us denote by [Σ] the set of all sequences over some set of symbols Σ, so that the domain and the range of *R* are subsets of [Σ].
- We can then consider *R* as a function mapping an input string into a set of output strings, formally denoted by this

type signature, where [Σ] equals *String*:

$$\mathcal{R} :: [\Sigma] \rightarrow \{[\Sigma]\} \qquad \mathcal{R} :: \mathit{String} \rightarrow \{\mathit{String}\} \qquad (1.1)$$

- A theoretical limitation of finite-state models of morphology is the problem of capturing **reduplication** of words or their elements (e.g., to express plurality) found in several human languages.

Unification-Based Morphology

- The concepts and methods of these formalisms are often closely connected to those of logic programming.

- In finite-state morphological models, both surface and lexical forms are by themselvesunstructured strings of atomic symbols.

- In higher-level approaches, linguistic information is expressed by more appropriate

data structures that can include complex values or can be recursively nested if needed.

- Morphological parsing *P* thus associates linear forms φ with alternatives of structured

content ψ, cf.

$$\mathcal{P} :: \phi \to \{\psi\} \qquad \mathcal{P} :: form \to \{content\} \qquad (1.2)$$

- morphological modelling, word forms are best captured by regular expressions, while the linguistic content is best described through **typed feature structures**.

- Feature structures can be viewed as directed acyclic graphs.

- A node in a feature structure comprises a set of attributes whose values can be

- Nodes are associated with types, and atomic values are attribute less nodes distinguished by their type.
- Unification is the key operation by which feature structures can be merged into a more informative feature structure.
- Unification of feature structures can also fail, which means that the information in themis mutually incompatible.
- Morphological models of this kind are typically formulated as logic programs, and unification is used to solve the system of constraints imposed by the model.
- Advantages of this approach include better abstraction possibilities for developing a morphological grammar as well as elimination of redundant information from it.
- Unification-based models have been implemented for Russian, Czech, Slovene, Persian, Hebrew, Arabic, and other languages.

Functional Morphology

- Functional morphology defines its models using principles of functional programming and type theory.

- It treats morphological operations and processes as pure mathematical functions and organizes the linguistic as well as abstract elements of a model into distinct types of values and type classes.

- Though functional morphology is not limited to modelling particular types of morphologies in human languages, it is especially useful for fusional morphologies.

- Functional morphology implementations are intended to be reused as programming libraries capable of handling the complete morphology of a language and to be incorporated into various kinds of applications.

- Morphological parsing is just one usage of the system, the others being morphological generation, lexicon browsing, and so on.

we can describe inflection *I*, derivation *D*, and lookup *L* as functions of these generic type

$$\mathcal{I} :: \mathit{lexeme} \to \{\mathit{parameter}\} \to \{\mathit{form}\} \qquad (1.3)$$

$$\mathcal{D} :: \mathit{lexeme} \to \{\mathit{parameter}\} \to \{\mathit{lexeme}\} \qquad (1.4)$$
$$\mathcal{L} :: \mathit{content} \to \{\mathit{lexeme}\} \qquad (1.5)$$

- Many functional morphology implementations are embedded in a general-purpose programming language, which gives programmers more freedom with advanced programming techniques and allows them to develop full-featured, real-world applications for their models.

- It influenced the functional morphology framework in with which morphologies of Latin, Swedish, Spanish, Urdu, and other languages have been implemented.

- The notation then constitutes a so-called domain-specific embedded language, which makes programming even more fun.

- Even without the options provided by general-purpose programming languages, functional morphology models achieve high levels of abstraction.
- Morphological grammars in Grammatical Framework can be extended with descriptions of the syntax andsemantics of a language.
- Grammatical Framework itself supports multilinguality, and models of more than a dozen languages are available in it as open-source software.

2.Finding structure of Documents

Introduction

- In human language, words and sentences do not appear randomly but have structure.
- For example, combinations of words from sentences- meaningful grammatical units, such as statements, requests, and commands.
- Automatic extraction of structure of documents helps subsequent NLP tasks: for example, parsing, machine translation, and semantic role labelling use sentences as the basic processing unit.
- Sentence boundary annotation(labelling) is also important for aiding human readability of

automatic speech recognition (ASR) systems.

- Task of deciding where sentences start and end given a sequence of characters(made of wordsand typographical cues) **sentences boundary detection.**
- **Topic segmentation** as the task of determining when a topic starts and ends in a sequence of

sentences.

- The statistical classification approaches that try to find the presence of sentence and topicboundaries given human-annotated training data, for segmentation.

- These methods base their predictions on features of the input: local characteristics that give evidence toward the presence or absence of a sentence, such as aperiod(.), a question mark(?), an exclamation mark(!), or another type of punctuation.

- Features are the core of classification approaches and require careful design and selection in order to be successful and prevent overfitting and noise problem.

- Most statistical approaches described here are language independent, every language is achallenging in itself.

- For example, for processing of Chinese documents, the processor may need to first segmentthe character sequences into words, as the words usually are not separated by a space.

- Similarly, for morphological rich languages, the word structure may need to be analyzed toextract additional features.

- Such processing is usually done in a pre-processing step, where a sequence of tokens isdetermined.
- Tokens can be word or sub-word units, depending on the task and language.
- These algorithms are then applied on tokens.

Sentence Boundary Detection

- **Sentence boundary detection** (Sentence segmentation) deals with automatically segmenting

a sequence of word tokens into sentence units.

- In written text in English and some other languages, the beginning of a sentence is usuallymarked with an uppercase letter, and the end of a sentence is explicitly marked with a

period(.), a question mark(?), an exclamation mark(!), or another type of punctuation.

- In addition to their role as sentence boundary markers, capitalized initial letters are used

distinguish proper nouns, periods are used in abbreviations, and numbers and punctuationmarks are used inside proper names.

- The period at the end of an abbreviation can mark a sentence boundary at the same time.
- Example: I spoke with Dr. Smith. and My house is on Mountain Dr.
- In the first sentence, the abbreviation Dr. does not end a sentence, and in the second it does.
- Especially **quoted sentences** are always problematic, as the speakers may have utteredmultiple sentences, and sentence boundaries inside the quotes are also marked with

punctuation marks.

- An automatic method that outputs word boundaries as ending sentences according to the

presence of such punctuation marks would result in cutting some sentences incorrectly.

- Ambiguous abbreviations and capitalizations are not only problem of sentence segmentationin written text.
- Spontaneously written texts, such as short message service (SMS) texts or instant

messaging(IM) texts, tend to be nongrammatical and have poorly used or missing punctuation, which makes sentence segmentation even more challenging.

- Similarly, if the text input to be segmented into sentences comes from an **automatic system**,

such as optical character recognition (OCR) or ASR, that aims to translate images of handwritten, type written, or printed text or spoken utterances into machine editable text, thefinding of sentences boundaries must deal with the errors of those systems as well.

- On the other hand, for conversational speech or text or multiparty meetings with

ungrammatical sentences and disfluencies, in most cases it is not clear where the boundaries are.

- Code switching -that is, the use of words, phrases, or sentences from multiple languages bymultilingual speakers- is another problem that can affect the characteristics of sentences.

- For example, when switching to a different language, the writer can either keep the

punctuation rules from the first language or resort to the code of the second language.

- Conventional rule-based sentence segmentation systems in well-formed texts rely on patterns

to identify potential ends of sentences and lists of abbreviations for disambiguating them.

- For example, if the word before the boundary is a known abbreviation, such as “Mr.” or “Gov.,”the text is not segmented at that position even though some periods are exceptions.
- To improve on such a rule-based approach, sentence segmentation is stated as a classificationproblem.
- Given the training data where all sentence boundaries are marked, we can train a classifier torecognize them.

Topic Boundary Detection

- **Segmentation**(Discourse or text segmentation) is the task of automatically dividing a streamof text or speech into topically homogenous blocks.
- This is, given a sequence of(written or spoken) words, the **aim of topic segmentation** is to

find the boundaries where topics change.

- Topic segmentation is an important task for various language understanding applications, such as information extraction and retrieval and text summarization.
- For example, in information retrieval, if a long documents can be segmented into shorter, topically coherent segments, then only the segment that is about the user’s query could be retrieved.

- During the late1990s, the U.S defence advanced research project agency(DARPA) initiated the**topic detection and tracking program** to further the state of the art in finding and following **new topic** in a stream of broadcast news stories.
- One of the tasks in the TDT effort was segmenting a **news stream into individual stories**.

Methods

- Sentence segmentation and topic segmentation have been considered as a **boundary classification problem**.
- Given a boundary candidate(between two word tokens for sentence segmentation andbetween two sentences for topic segmentation), the goal is to predict whether or not the candidate is an actual boundary (sentence or topic boundary).
- Formally, let **xƐX be the vector of features** (the observation) associated with a candidate and **y ƐY be the label** predicted for that candidate.
- The label y can be **b for boundary** and $\bar{b}$**for nonboundary**.

- **Classification problem**: given a set of training examples$(x,y)_{train}$, find a function that will assignthe most accurate possible label y of unseen examples x_{unseen}.
- Alternatively to the binary classification problem, it is possible to model boundary types using

finer-grained categories.

- For segmentation in text be framed as a three-class problem: sentence boundary b^a, withoutan abbreviation and abbreviation not as a boundary
- Similarly spoken language, a three way classification can be made between non-boundaries

  statements b^s, and question $b^{\bar{a}}$ boundaries b^q. $\bar{b}^a$. $\bar{b}$
- For sentence or topic segmentation, the problem is defined as finding the most probablesentence or topic boundaries.
- The natural unit of sentence segmentation is words and of topic segmentation is sentence, as

we can assume that topics typically do not change in the middle of a sentences.

- The words or sentences are then grouped into categories stretches belonging to one sentences or topic- that is word or sentence boundaries are classified into sentences or topic boundaries and -non-boundaries.

- The classification can be done at each potential boundary *i* (local modelling); then, the aim is

to estimate the most probable boundary type $\hat{y}$ for each candidate x_i

$$\hat{y} = \underset{y_i\ in\ Y}{argmax}\ P(y_i | x_i)$$

Here, the ^ is used to denote estimated categories, and a variable without a ^ is used to showpossible categories.

- In this formulation, a category is assigned to each example in isolation; hence, decision is made locally.
- However, the consecutive types can be related to each other. For example, in broadcast newsspeech, two consecutive sentences boundaries that form a single word sentence are very infrequent.
- In local modelling, features can be extracted from surrounding example context of the

candidate boundary to model such dependencies.

- It is also possible to see the candidate boundaries as a sequence and search for the sequence of boundary types $\hat{Y} = \hat{y_1}, \ldots, \hat{y_n}$

that have the maximum probability given the candidate examples, $X = \mathbf{x}_1, \ldots, \mathbf{x}_n.$

$$\hat{Y} = \underset{y}{argmax}\ P(Y|X)$$

- We categorize the methods into local and sequence classification.
- Another categorization of methods is done according to the type of the machine learning algorithm: **generative versus discriminative**.
- Generative sequence models estimate the **joint distribution** of the observations P(X,Y) (words, punctuation) and thelabels(sentence boundary, topic boundary).
- Discriminative sequence models, however, **focus on features** that categorize the differences between the labelling of that examples.

1.Generative Sequence Classification Methods

- Most commonly used generative sequence classification method for topic and sentence is the hidden Markov model (HMM) function is being used in which the model is proposed according to bayers rule

Hmm Means: A hidden Markov model (HMM) is a statistical model that can be used **to describe the evolution of observable events that depend on internal factors, which are not directly observable**

- generative models can be handled byHELMs(hidden event language model) which can handle **multiple orders of magnitude larger**

Training data sets

- The probability in equation 2.2 is rewritten as the following, using the Bayes rule:

$$\hat{Y} = {}^{rgmax}P(Y|X) \quad (\ 2.1)$$

$$\hat{Y} = \underset{y}{argmax}\ {}_{P}\left(Y|X\right) = \underset{y}{argmax}\left(P(X|Y)(Y)\mathrm{T}P(X)\right) = \underset{y}{argmax}\left(P(X|Y)\right)^{(Y)(\ 2.2)}$$

Here $\hat{Y}$ = Predicted class(boundary) label

$Y = (y_1, y_2, \ldots y_k)$= Set of class(boundary) labels

$X = (x_1, x_2, \ldots x_n)$= set of feature vectors

P(Y|X) = the probability of given the X (feature vectors),

what is the probability of X
belongs to the class(boundary) label.

P(x) = Probability of word sequence
P(Y) = Probability of the class(boundary)

$$\hat{Y} = \operatorname*{argmax}_{Y} P(Y|X) = \operatorname*{argmax}_{Y} \frac{P(X|Y)P(Y)}{P(X)} = \operatorname*{argmax}_{Y} P(X|Y)P(Y) \qquad (2.3)$$

- P(X) in the denominator is dropped because it is fixed for different Y and hence does notchange the argument of max.
- P(X|Y) and P(Y) can be estimated as

$$P(X|Y) = \prod_{i=1}^{n} P(\mathbf{x}_i|y_1, \ldots, y_i) \tag{2.4}$$

and

$$P(Y) = \prod_{i=1}^{n} P(y_i|y_1, \ldots y_{i-1}) \tag{2.5}$$

2.Discriminative Local Classification Methods

- Discriminative classifiers aim to model **$P(y_i \, |x_i)$ equation 2.1 directly**.
- The most important distinction is that c**lass densities P(x|y)** are model assumptions **in generative approaches**
- A number of discriminative classification approaches are used, such as support vector machines, boosting, maximum entropy, and regression. Are based on different machine learning algorithms which are used in discrimination process in classifying the sentence boundary.
- While discriminative approaches have been shown to outperform generative methods in many speech and language processing tasks.
- For **sentence segmentation, supervised learning methods – have primarily been applied to newspaper articles**.

Supervised learning methods are used where a machine is being trained by giving instruction and accordingly it will perform .there are many supervised learning algorithms for different purpose.

- Stamatatos, Fakotakis and Kokkinakis are authors who used **transformation based learning** (TBL) to infer rulesfor **finding sentence boundaries**.

- Many supervised learning method classifiers have been tried for the sentence boundary task such as regression trees, neural networks, classification, maximum entropy classifiers, support vector machines, and naïve Bayes classifiers.

- The most Text tiling method is used for topic segmentation which uses a l**exical cohesion (binding of word to another) metric** in a

word vector space as an indicator for topic similarity.

- Figure depicts a typical graph of similarity with respect to **consecutive segmentation units**.

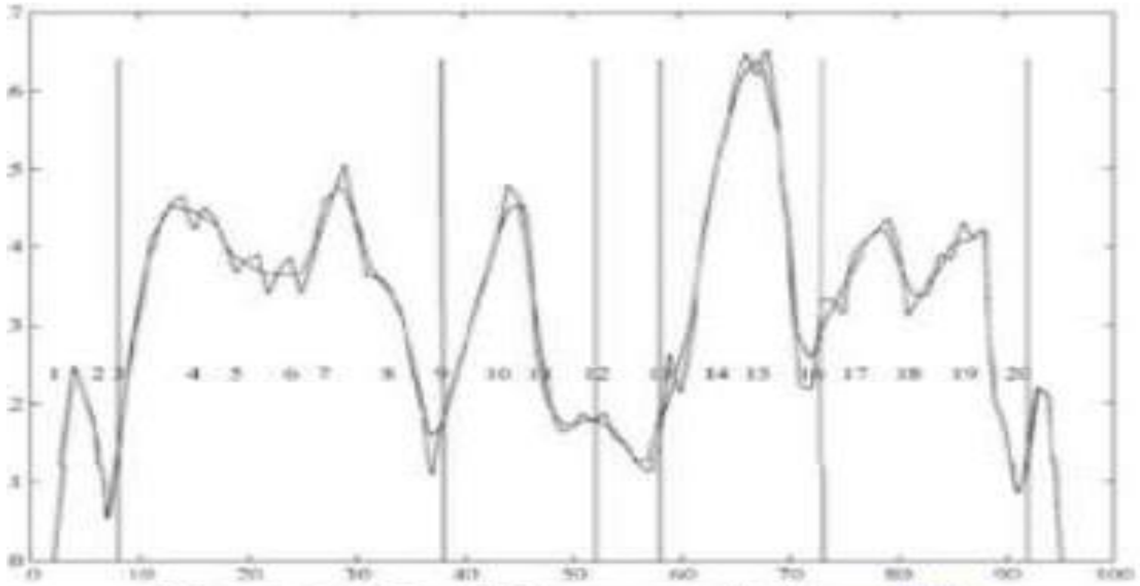


Figure 2–4. Text Tiling example (from [22])

- The document is chopped when the similarity is below some threshold.

- Originally, **two methods** for computing the similarity scores were proposed: **block**

comparison and **vocabulary introduction**.

- The first, **block comparison, compares adjacent blocks of text to see how similar** they areaccording to how many words the adjacent blocks have in common.
- **Given two blocks, b_1 and b_2, each having k tokens** (sentences or paragraphs),
- **the similarity(or topical cohesion) between two blocks score** is computed by the **formula**:

$$\frac{\sum_t w_{t,b_1} . w_{t,b_2}}{\sqrt{\sum_t w_{t,b_1}^2 \sum_t w_{t,b_2}^2}}$$

- **Where $w_{t,b}$ is the weight assigned to term t in block b**.
- The weights can be binary or may be computed using other information retrieval- metricssuch as term frequency(calculation of weight).
- The second method is, the **vocabulary introduction method**, **assigns a score to a token-sequence gap**

on the basis of **how many new words are seen in the interval in which it is the midpoint**.

- Similar to the **block comparison formulation, given two consecutive blocks b_1 and b_2, of equal number of words w**, the
-
- **topical cohesion** score is computed with the following formula:

$$\frac{NumNewTerms(b_1) + NumNewTerms(}{2 \times w}$$

- Where **NumNewTerms(b) returns the number of terms in block b seen the first time in text**.

3.Discriminative Sequence Classification Methods

- In segmentation tasks, the sentence or topic decision for a given example(word, sentence, paragraph) highly depends on the decision for the examples in its vicinity(the area near to the topic or surrounding a particular database).
- Discriminative sequence classification methods are in general extensions of local discriminative models with additional decoding stages that find the best assignment of labels by looking at neighbouring decisions to label.
- Machine learning algorithms are used to discriminative sequence example(word, sentence, paragraph) commonly used are

  Conditional random fields(CRFs), SVM-support vector machine which are extenstion of HMM

.

- Contrary to local classifiers that predict sentences or topic boundaries independently, CRFscan oversee the whole sequence of boundary hypotheses to make their decisions.

4.Hybrid Approaches

In this approaches we use segamwnt classification method s by applying Viteribe Algorithm which is implemented byHmm

The Viterbi algorithm is a dynamic programming algorithm for obtaining the maximum a posteriori probability estimate of the most likely sequence of hidden states —called the Viterbi path—that results in a sequence of observed events, especially in the context of hidden Markov models (HMM).

Complexity of the Approaches

- The above approaches described here have **advantages** and **disadvantages**.
- In a **given context** and under a set of observation features, one **approach may be better than**

other.

- These approaches can be rated in terms of **complexity** (time and memory) of **their training** and **prediction algorithms** and in terms of their **performance on real-world datasets**.

- In terms of complexity, **training of discriminative approaches** is **more complex than training of generative ones** because they require multiple passes over the training data to adjust for feature weights**.**
- However, generative models can be handled byHELMs(hidden event language model) which can handle **multiple orders of magnitude larger**

Training data sets

On the other hand the disadvantage is , they work with **only a few features .**

Vyasapuri, Bandlaguda, Post:Keshavgiri
Hyderabad-500005,Telangana, India
Tel:040-29880079, 86,8978380692, 9642703342
9652216001, 9550544411, Website:www.mist.ac.in
Email:principal@mist.ac.in
principal.mahaveer@gmail.com
Counseling code:MHVR, University Code:E3

MAHAVEER
INSTITUTE OF SCIENCE & TECHNOLOGY
(AN UGC AUTONOMOUS INSTITUTION)
Approved by AICTE, Affiliated to JNTU,Hyderabad
Accredited by NAAC with 'A' Grade
Recognized Under 2(f) of UGC Act 1956,ISO 9001:2015 Certified
ESTD : 2001

Department of Computer Science and Engineering (AIML)

(R18)

Natural Language Processing

Lecture Notes

B. Tech III YEAR – II SEM

Prepared by

Mrs.Swapna
(Professor&HOD-CSM)
Dept. CSE(AIML)

Syllabus

Faculty Name : Mrs Swapna
Subject Name :NLP

NATURAL LANGUAGE PROCESSING

B.Tech. III Year II Sem.

L	T	P	C
3	1	0	4

Prerequisites: Data structures, finite automata and probability theory

Course Objectives:

- Introduce to some of the problems and solutions of NLP and their relation to linguistics and statistics.

Course Outcomes:

- Show sensitivity to linguistic phenomena and an ability to model them with formal grammars.
- Understand and carry out proper experimental methodology for training and evaluating empirical NLP systems
- Able to manipulate probabilities, construct statistical models over strings and trees, and estimate parameters using supervised and unsupervised training methods.
- Able to design, implement, and analyze NLP algorithms
- Able to design different language modeling Techniques.

UNIT - I

Finding the Structure of Words: Words and Their Components, Issues and Challenges, Morphological Models

Finding the Structure of Documents: Introduction, Methods, Complexity of the Approaches, Performances of the Approaches

UNIT - II

Syntax Analysis: Parsing Natural Language, Treebanks: A Data-Driven Approach to Syntax, Representation of Syntactic Structure, Parsing Algorithms, Models for Ambiguity Resolution in Parsing, Multilingual Issues

UNIT - III

Semantic Parsing: Introduction, Semantic Interpretation, System Paradigms, Word Sense Systems, Software.

UNIT - IV

Predicate-Argument Structure, Meaning Representation Systems, Software.

UNIT - V

Discourse Processing: Cohension, Reference Resolution, Discourse Cohension and Structure **Language Modeling:** Introduction, N-Gram Models, Language Model Evaluation, Parameter Estimation, Language Model Adaptation, Types of Language Models, Language-Specific Modeling Problems, Multilingual and Crosslingual Language Modeling

TEXT BOOKS:

Multilingual natural Language Processing Applications: From Theory to Practice – DAniel M. Bikel and Imed Zitouni, Pearson Publication

Natural Language Processing and Information Retrieval: Tanvier Siddiqui, U.S. Tiwary

Unit 2

Unit 2

Parsing Natural Language:

Natural Language Processing (NLP) is a field of study that deals with understanding, interpreting, and manipulating human spoken languages using computers. Since most of the significant information is written down in natural languages such as English, French, German, etc. thus, NLP helps computers communicate with humans in their own languages and perform other language-related tasks.

In conclusion, NLP makes it possible for computers to read the text, hear speech, interpret and realize it, understand the sentiment, and identify important parts of a text or speech.

What is Syntax?

A natural language typically follows a hierarchical structure, and contains the following components:

- Sentences
- Clauses
- Phrases
- Words

Syntax refers to the set of rules, principles, processes that govern the structure of sentences in a natural language. One basic description of *syntax* is how different words such as Subject, Verbs, Nouns, Noun Phrases, etc. are sequenced in a sentence.

Some of the syntactic categories of a natural language are as follows:

- Sentence(S)
- Noun Phrase(NP)

- Determiner(Det)
- Verb Phrase(VP)
- Prepositional Phrase(PP)
- Verb(V)
- Noun(N)

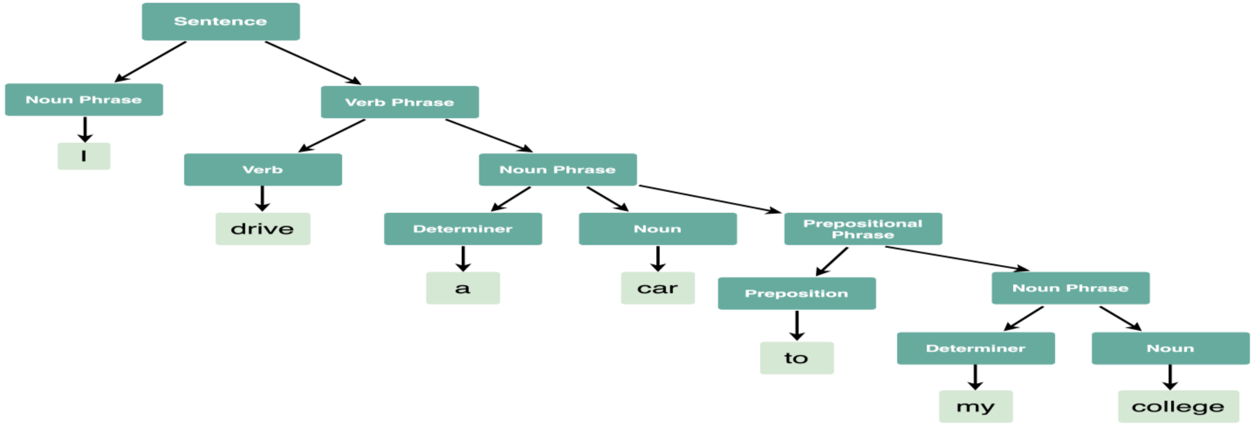
Syntax Tree:

A Syntax tree or a parse tree is a tree representation of different syntactic categories of a sentence. It helps us to understand the syntactical structure of a sentence.

Example:

The syntax tree for the sentence given below is as follows:

I drive a car to my college.



One of the important use cases of Natural Language Processing (NLP) is generative text. Generative text is predicting what word must come next in a sentence. Applications of generative text are question-answering chatbots, sentence or word autocorrection, and autocompletion, grammar check, and these cases have now become indispensable and part of our day-to-day lives.

To help us know what word will come next we need to learn as much as we can what words previously came in a sentence. To fulfil this need and to understand what words came priorly is where parts of speech and syntactic parsing are very important and integral topics in NLP.

One of the important use cases of Natural Language Processing (NLP) is generative text. Generative text is predicting what word must come next in a sentence. Applications of generative text are question-answering chatbots, sentence or word autocorrection, and autocompletion, grammar check, and these cases have now become indispensable and part of our day-to-day lives.

To help us know what word will come next we need to learn as much as we can what words previously came in a sentence. To fulfil this need and to understand what words came priorly is where parts of speech and syntactic parsing are very important and integral topics in NLP.

Language Syntax

The language syntax is fundamental for generative text and sets the foundation for parts of speech and parse trees.

The word *syntax* originates from the Greek word syntaxis, meaning “arrangement”, and refers to how the words are arranged together. Henceforth, language syntax means how the language is structured or arranged.

How are words arranged together?

There are many different ways to categorize these structures or arrangements. One way to classify how the words are arranged is by grouping them as the words behave as a single unit or phrase, which is also known as a constituent.

A sentence can have different language rules applied to it and have different types of structure. As different parts of the sentence are based on different parts of the syntax that follow the same grammar rules that are of a noun phrase, verb phrase, and prepositional phrase.

A sentence is structured as follows:

Sentence = S = Noun Phrase + Verb Phrase + Preposition Phrase

S = NP + VP + PP

The different word groups that exist according to English grammar rules are:

Noun Phrase(NP): Determiner + Nominal Nouns = DET + Nominal
Verb Phrase (VP): Verb + range of combinations
Prepositional Phrase (PP): Preposition + Noun Phrase = P + NP

We can make different forms and structures versions of the noun phrase, verb phrase, and prepositional phrase and join in a sentence.

For instance, let us see a sentence: *The boy ate the pancakes*. This sentence has the following structure:

The boy: Noun Phrase
ate: Verb
the pancakes: Noun Phrase (Determiner + Noun)

This sentence is correct both structurally and contextually.

However, now taking another sentence: *The boy ate the pancakes under the door*.

The boy: Noun Phrase
ate: Verb
the pancakes: Noun Phrase (Determiner + Noun)
under: preposition
the door: Noun Phrase (Determiner + Noun)

Here, the preposition *under* is followed by the noun phrase *the door*, which is syntactically correct but not correct contextually.

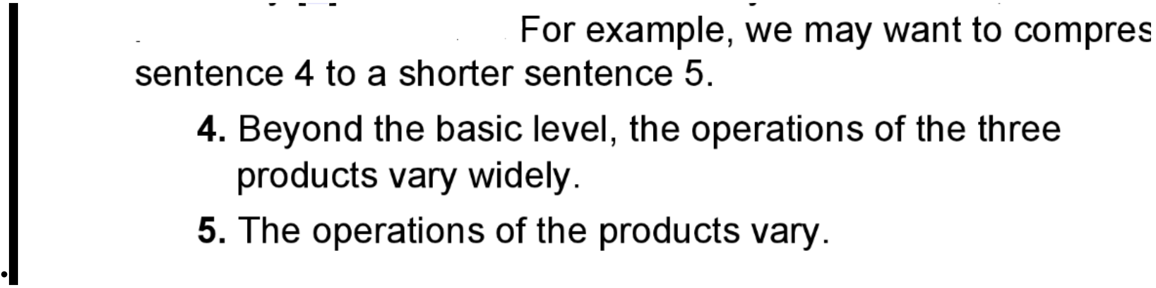
Taking the same sentence in another way: *The boy ate the pancakes from the jumping table.*

The boy: Noun Phrase
ate: Verb
the pancakes: Noun Phrase (Determiner + Noun)
from: preposition
jumping table: Verb Phrase

This sentence is syntactically incorrect as the preposition *form* is followed by a verb phrase *jumping table*.

Such sentence are marked to the root and are been cut off from the tree

The following below sentence in which the sentence is been shortened and it is been removed from the tree and you can observe that there is circle been rounded off.



For example, we may want to compres sentence 4 to a shorter sentence 5.

4. Beyond the basic level, the operations of the three products vary widely.

5. The operations of the products vary.

as a verb phrase or a noun phrase. The output of the parser for the input sentence 4 is shown in Figure 3–1. The parse tree produced by the parser can now be edited using a compression model that is aware of constituents, and a few choice constituent deletions can produce a fluent compressed version of the original sentence.

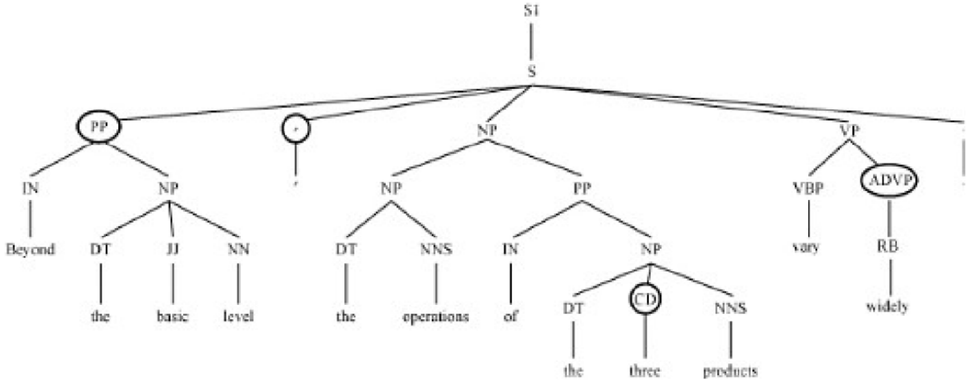
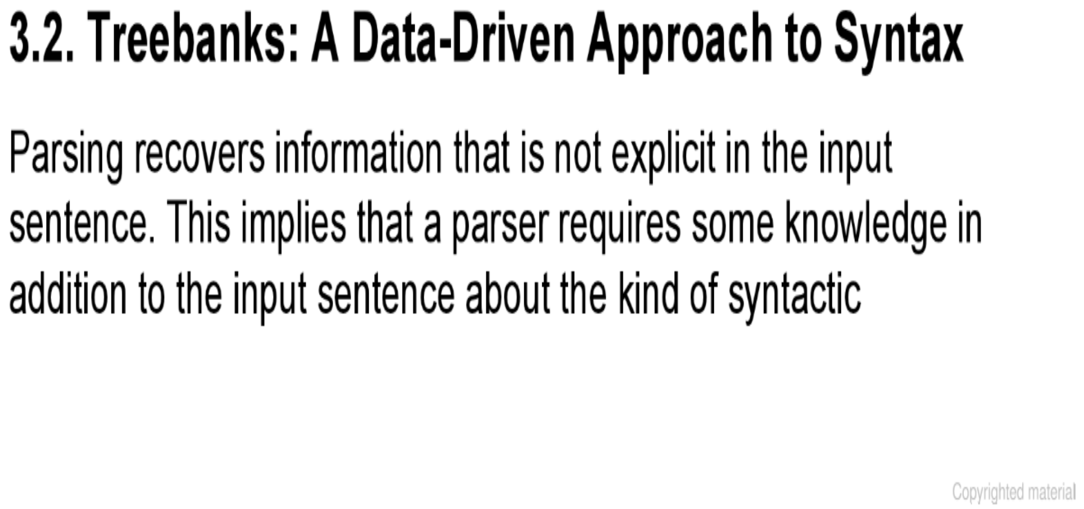
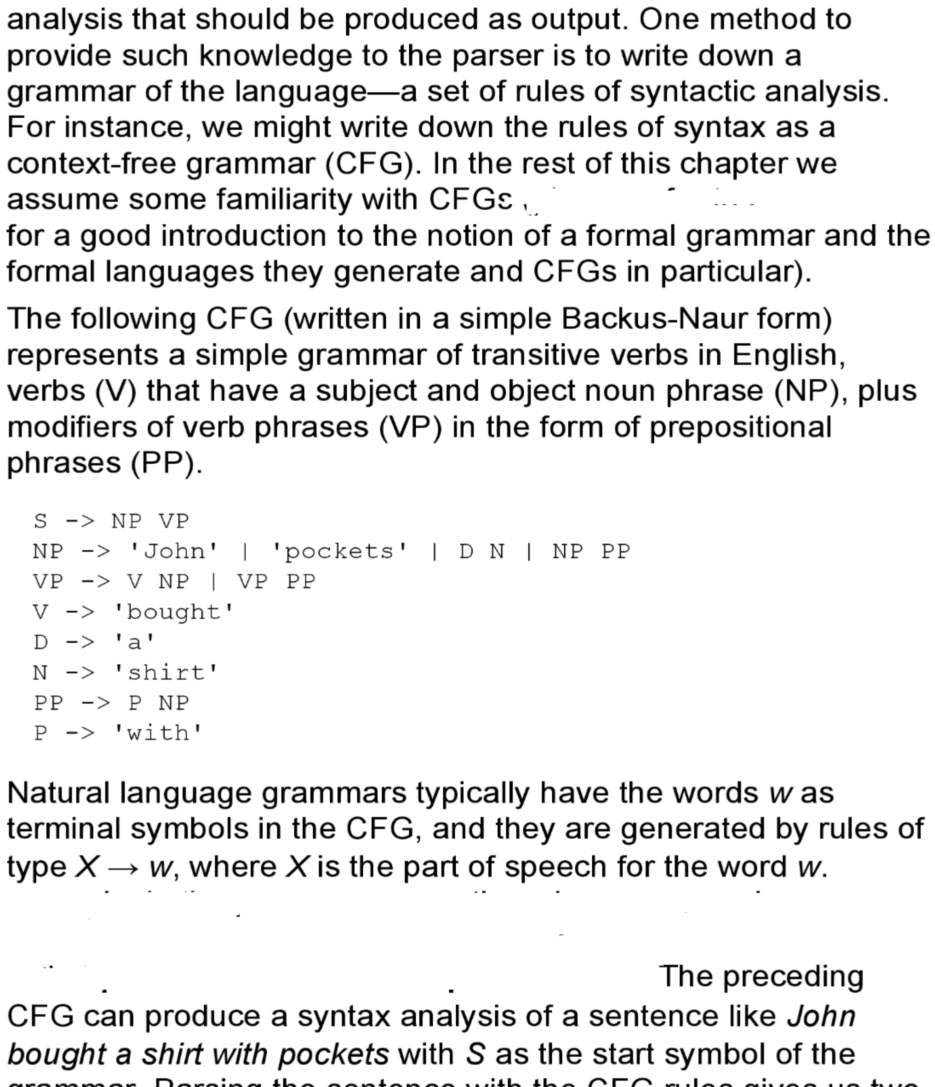


Figure 3–1. Parser output for sentence 4. Deleting the circled constituents PP, , , CD, and ADVP results in the shorter fluent sentence *The operations of the products vary*.



3.2. Treebanks: A Data-Driven Approach to Syntax

Parsing recovers information that is not explicit in the input sentence. This implies that a parser requires some knowledge in addition to the input sentence about the kind of syntactic



analysis that should be produced as output. One method to provide such knowledge to the parser is to write down a grammar of the language—a set of rules of syntactic analysis. For instance, we might write down the rules of syntax as a context-free grammar (CFG). In the rest of this chapter we assume some familiarity with CFGs

for a good introduction to the notion of a formal grammar and the formal languages they generate and CFGs in particular).

The following CFG (written in a simple Backus-Naur form) represents a simple grammar of transitive verbs in English, verbs (V) that have a subject and object noun phrase (NP), plus modifiers of verb phrases (VP) in the form of prepositional phrases (PP).

```
S -> NP VP
NP -> 'John' | 'pockets' | D N | NP PP
VP -> V NP | VP PP
V -> 'bought'
D -> 'a'
N -> 'shirt'
PP -> P NP
P -> 'with'
```

Natural language grammars typically have the words *w* as terminal symbols in the CFG, and they are generated by rules of type *X* → *w*, where *X* is the part of speech for the word *w*.

The preceding CFG can produce a syntax analysis of a sentence like *John bought a shirt with pockets* with *S* as the start symbol of the kind of shirt that has pockets.

```
(S  (NP John)                               (S  (NP John)
   (VP (VP (V bought)                          (VP (V bought)
           (NP (D a)                               (NP (NP (D a)
               (N shirt)))                                 (N
shirt))
       (PP (P with)                                        (PP (P
with)
           (NP pockets))))                                     (NP
pockets)))))
```

However, writing down a CFG for the syntactic analysis of natural language is problematic. Unlike a programming language, natural language is far too complex to simply list all the syntactic rules in terms of a CFG. A simple list of rules does not consider interactions between different components in the grammar. We could extend this grammar to include other types of verbs and other syntactic constructions, but listing all possible syntactic constructions in a language is a difficult task. In addition, it is difficult to exhaustively list lexical properties of words, for instance, to list all the grammar rules in which a particular word can be a participant. This is a typical knowledge acquisition problem.

Apart from this knowledge acquisition problem, there is a less apparent problem: it turns out that the rules interact with each other in combinatorially explosive ways. Consider a simple CFG that provides a syntactic analysis of noun phrases as a binary branching tree:

```
N -> N N
N -> 'natural' | 'language' | 'processing' | 'book'
```

natural language processing we use the recursive rule twice in each parse, and there are two ambiguous parses:

```
(N (N (N natural)                   (N (N natural)
      (N language))                    (N (N language)
   (N processing))                        (N processing)))
```

Note that the ambiguity in the syntactic analysis reflects a real ambiguity: is it a processing of natural language, or is it a natural way to do language processing? So this issue cannot be resolved by changing the formalism in which the rules are written (e.g., by using finite-state automata, which can be deterministic but cannot simultaneously model both meanings in a single grammar). Any system of writing down syntactic rules should represent this ambiguity. However, by using the recursive rule three times, we get five parses for *natural language processing book* and for longer and longer input noun phrases, using the recursive rule four times, we get 14 parses; using it five times, we get 42 parses; using it six times, we get 132 parses. In fact, for CFGs it can be proved that the number of parses obtained by using the recursive rule *n* times is the Catalan number of *n*:

$$\mathrm{Cat}(n) = \frac{1}{n+1}\binom{2n}{n}$$

A Treebank is simply collection of sentences where each sentence is provided a complete syntax analysis.

Two main approaches to syntax analysis are used to construct tree bank

1.Dependency graphs

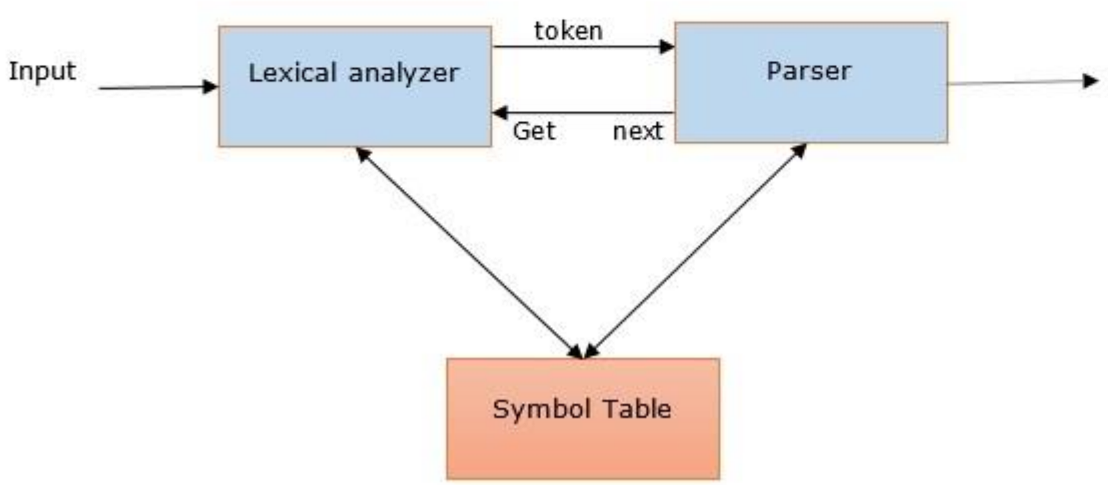
2.Pharse structure graphs

Parsing and its relevance in NLP

The word ‘Parsing’ whose origin is from Latin word **‘pars’** (which means **‘part’**), is used to draw exact meaning or dictionary meaning from the text. It is also called Syntactic analysis or syntax analysis. Comparing the rules of formal grammar, syntax analysis checks the text for meaningfulness. The sentence like “Give me hot ice-cream”, for example, would be rejected by parser or syntactic analyzer.

In this sense, we can define parsing or syntactic analysis or syntax analysis as follows −

It may be defined as the process of analyzing the strings of symbols in natural language conforming to the rules of formal grammar.



We can understand the relevance of parsing in NLP with the help of following points −

- Parser is used to report any syntax error.
- It helps to recover from commonly occurring error so that the processing of the remainder of program can be continued.
- Parse tree is created with the help of a parser.
- Parser is used to create symbol table, which plays an important role in NLP.
- Parser is also used to produce intermediate representations (IR).

Deep Vs Shallow Parsing

Deep Parsing	**Shallow Parsing**
In deep parsing, the search strategy will give a complete syntactic structure to a sentence.	It is the task of parsing a limited part of the syntactic information from the given task.
It is suitable for complex NLP applications.	It can be used for less complex NLP applications.
Dialogue systems and summarization are the examples of NLP applications where deep parsing is used.	Information extraction and text mining are the examples of NLP applications where deep parsing is used.
It is also called full parsing.	It is also called chunking.

Dependency Parsing

The term Dependency Parsing (DP) refers to the process of examining the dependencies between the phrases of a sentence in order to determine its grammatical structure. A sentence is divided into many sections based mostly on this. The process is based on the assumption that there is a direct relationship between each linguistic unit in a sentence. These hyperlinks are called dependencies.

Consider the following statement: “I prefer the morning flight through Denver.”

The diagram below explains the sentence’s dependence structure:

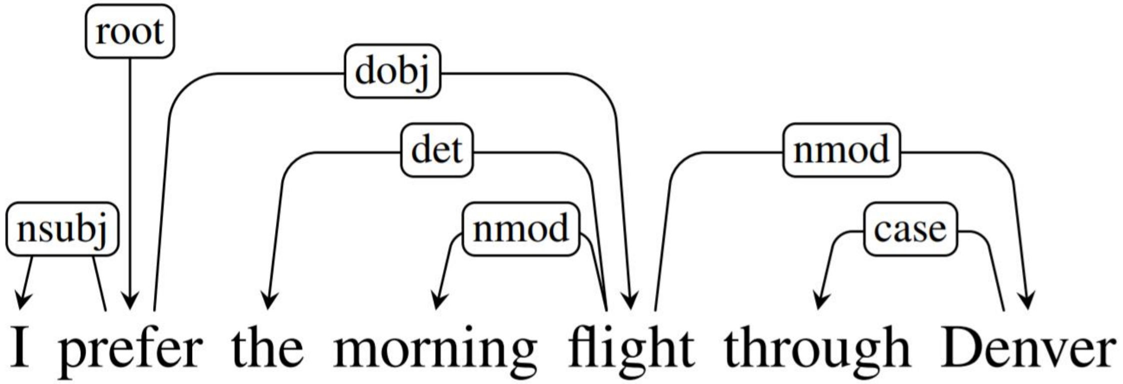


IMAGE – 1

In a written dependency structure, the relationships between each linguistic unit, or phrase, in the sentence are expressed by directed arcs. The root of the tree “prefer” varies the pinnacle of the preceding sentence, as labelled within the illustration.

A dependence tag indicates the relationship between two phrases. For example, the word “flight” changes the meaning of the noun “Denver.” As a result, you may identify a dependence from

flight -> Denver, where flight is the pinnacle and Denver is the kid or dependent. It’s represented by nmod, which stands for the nominal modifier.

This distinguishes the scenario for dependency between the two phrases, where one serves as the pinnacle and the other as the dependent. Currently, the Common Dependency V2 taxonomy consists of 37 common syntactic relationships, as shown in the table below:

Dependency Tag	**Description**
acl	clausal modifier of a noun (adnominal clause)
acl:relcl	relative clause modifier
advcl	adverbial clause modifier
advmod	adverbial modifier
advmod:emph	emphasizing phrase, intensifier
advmod:lmod	locative adverbial modifier
amod	adjectival modifier
appos	appositional modifier
aux	auxiliary
aux:move	passive auxiliary

case	case-marking
cc	coordinating conjunction
cc:preconj	preconjunct
ccomp	clausal complement
clf	classifier
compound	compound
compound:lvc	gentle verb building
compound:prt	phrasal verb particle
compound:redup	reduplicated compounds
compound:svc	serial verb compounds
conj	conjunct
cop	copula
csubj	clausal topic

csubj:move	clausal passive topic
dep	unspecified dependency
det	determiner
det:numgov	pronominal quantifier governing the case of the noun
det:nummod	pronominal quantifier agreeing with the case of the noun
det:poss	possessive determiner
discourse	discourse ingredient
dislocated	dislocated parts
expl	expletive
expl:impers	impersonal expletive
expl:move	reflexive pronoun utilized in reflexive passive
expl:pv	reflexive clitic with an inherently reflexive verb
mounted	mounted multiword expression

flat	flat multiword expression
flat:overseas	overseas phrases
flat:title	names
goeswith	goes with
iobj	oblique object
checklist	checklist
mark	marker
nmod	nominal modifier
nmod:poss	possessive nominal modifier
nmod:tmod	temporal modifier
nsubj	nominal topic
nsubj:move	passive nominal topic
nummod	numeric modifier

nummod:gov	numeric modifier governing the case of the noun
obj	object
obl	indirect nominal
obl:agent	agent modifier
obl:arg	indirect argument
obl:lmod	locative modifier
obl:tmod	temporal modifier
orphan	orphan
parataxis	parataxis
punct	punctuation
reparandum	overridden disfluency
root	root
vocative	vocative

xcomp	open clausal complement

Dependency Parsing using NLTK

The Pure Language Toolkit (NLTK) package deal will be used for Dependency Parsing, which is a set of libraries and codes used during statistical Pure Language Processing (NLP) of human language.

We may use NLTK to do dependency parsing in one of several ways:

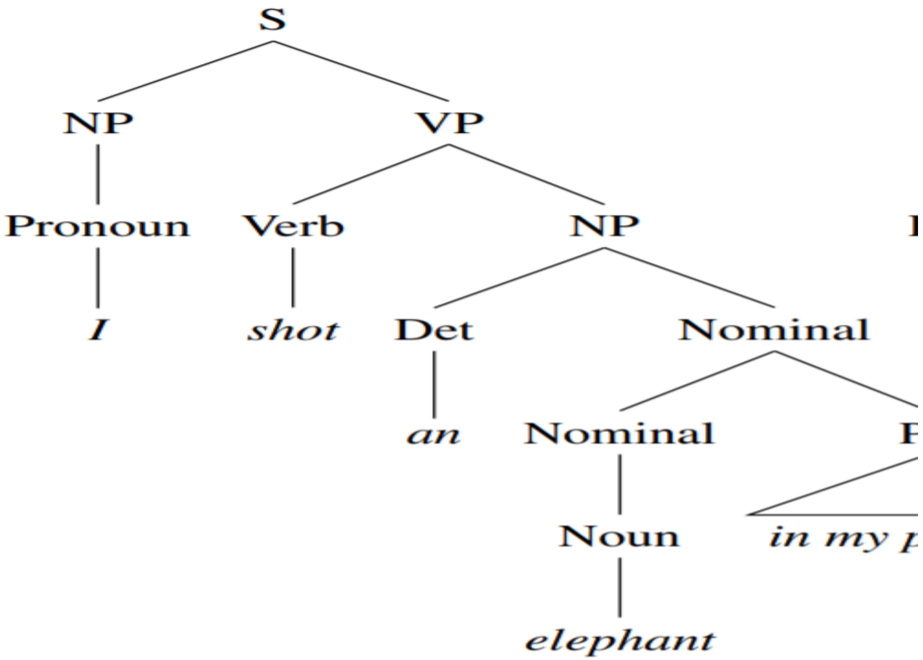
1. <u>Probabilistic, projective dependency parser</u>: These parsers predict new sentences by using human language data acquired from hand-parsed sentences. They’re known to make mistakes and work with a limited collection of coaching information.

2. <u>Stanford parser</u>: It is a Java-based pure language parser. You would want the Stanford CoreNLP parser to perform dependency parsing. The parser supports a number of languages, including English, Chinese, German, and Arabic.

Constituency Parsing or Pharse Structure Parser

Constituency Parsing is based on context-free grammars. Constituency Context-free grammars are used to parse text. Right here the parse tree includes sentences that have been broken down into sub-phrases, each of which belongs to a different grammar class. A terminal node is a linguistic unit or phrase that has a mother or father node and a part-of-speech tag.

For example, Consider the following example sentence: “I shot an elephant in my pajamas.” The constituency parse tree is shown graphically as follows:



The entire sentence is broken down into sub-phases till we’ve got terminal phrases remaining. VP stands for verb phrases, whereas NP stands for noun phrases.

Dependency Parsing vs

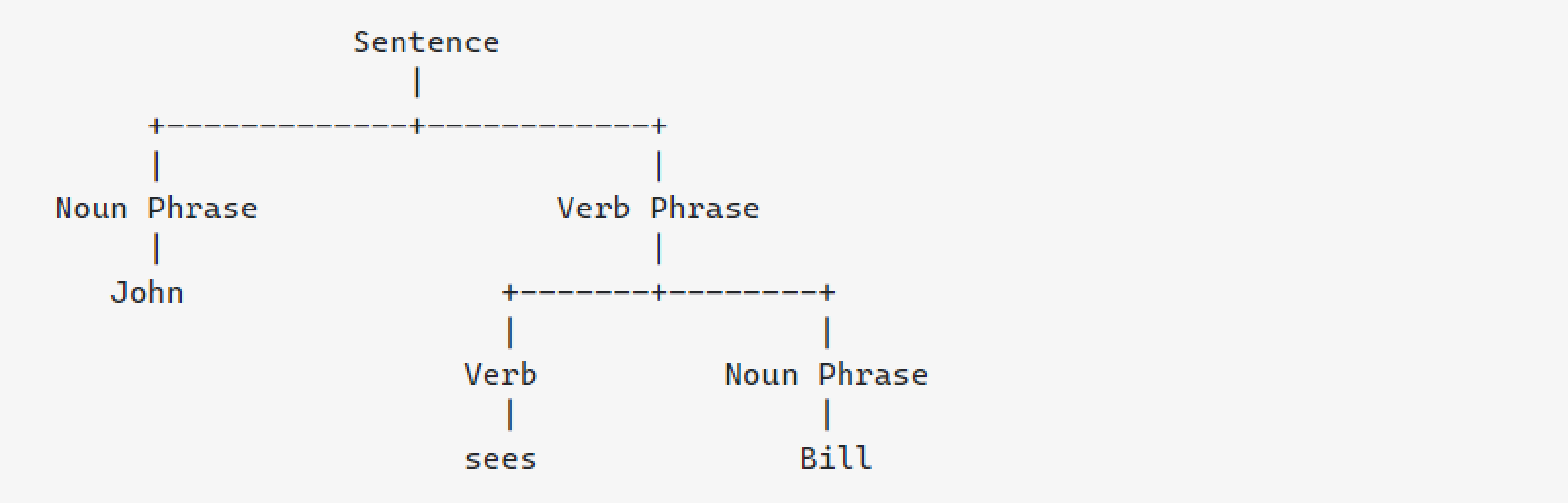
Constituency Parsing 0r Pharse parser

The Stanford parser will also be used to do constituency parsing. It begins by parsing a phrase using the constituency parser and then transforms the constituency parse tree into a dependency tree.

In case your main objective is to interrupt a sentence into sub-phrases, it is ideal to implement constituency parsing. However, dependency parsing is the best method for discovering the dependencies between phrases in a sentence.

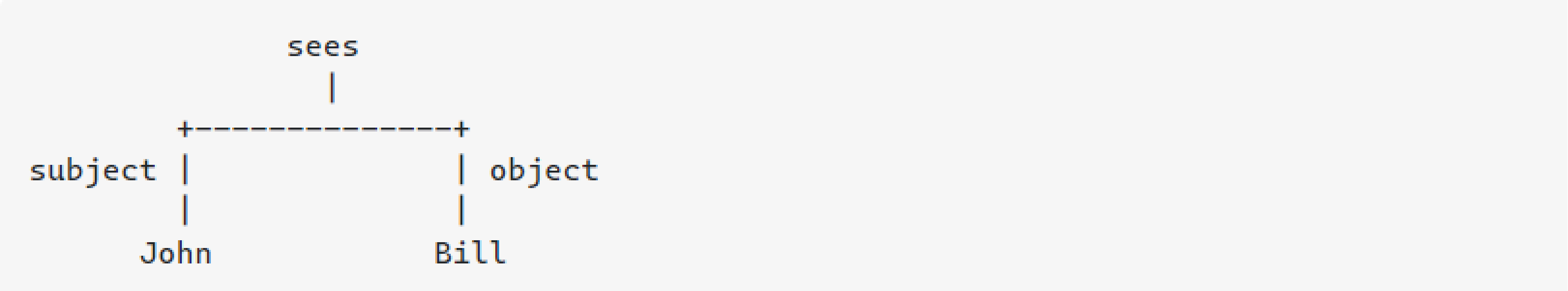
Let’s look at an example to see what the difference is:

A constituency parse tree denotes the subdivision of a text into sub-phrases. The tree’s non-terminals are different sorts of phrases, the terminals are the sentence’s words, and the edges are unlabeled. A constituency parse for the simple statement “John sees Bill” would be:



```
                Sentence
                   |
     +-------------+------------+
     |                          |
Noun Phrase                Verb Phrase
     |                          |
   John                 +-------+--------+
                        |                |
                      Verb          Noun Phrase
                        |                |
                      sees              Bill
```

A dependency parses links words together based on their connections. Each vertex in the tree corresponds to a word, child nodes to words that are reliant on the parent, and edges to relationships. The dependency parse for “John sees Bill” is as follows:



```
              sees
                |
        +--------------+
subject |              | object
        |              |
      John            Bill
```

You should choose the parser type that is most closely related to your objective. If you're looking for sub-phrases inside a sentence, you're definitely interested in the constituency parse. If you're interested in the connection between words, you're probably interested in the dependency parse.

- A projective dependency tree is one where if we put the words in a **linear order** based on the sentence with the **root symbol** in the **first position**, the dependency arcs can be drawn above the words without any crossing dependencies.

While non Projective dependency implies structure similar to phrase structure or constituency parse tree

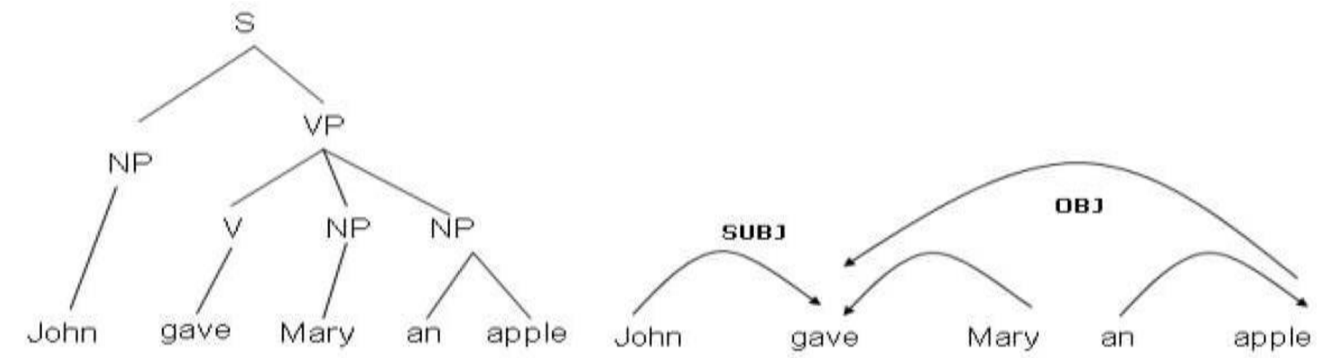


Figure 1
Phrase structure tree and the corresponding Dependency structure tree

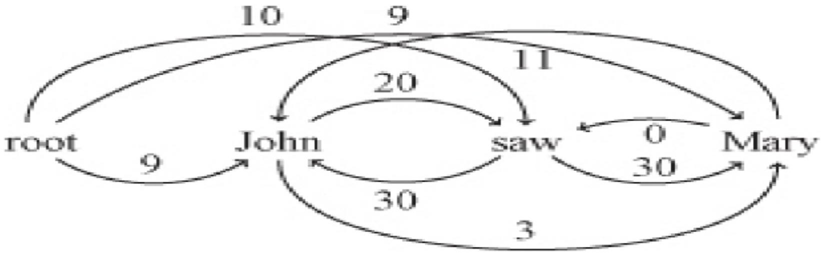
3.4.3. Minimum Spanning Trees and Dependency Parsing

Finding the optimum branching in a directed graph is closely related to the problem of finding a minimum spanning tree in an undirected graph. The directed graph case is of interest because it corresponds to a dependency tree, which is always rooted and cannot have cycles. A prerequisite is that each potential dependency link between words should have a score. In NLP, the tradition is to use the term **minimum spanning tree** (MST) to refer to the optimum branching problem in directed graphs. In the case of parsing with a dependency treebank, we assume we have some model that can be used to provide such a score based on estimates of likelihood of each dependency link in the dependency tree. These scores can be used to find the MST, which is the highest scoring dependency tree. Because the linear order of the words in the input is not taken into account in the MST formulation, crossing or nonprojective dependencies can be recovered by such a parser. This can be an issue in languages that are predominantly projective, like English, but provide a natural way to recover the crossing dependencies in languages like Czech.

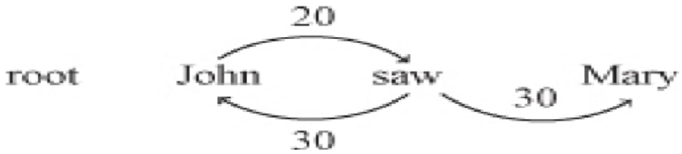
Rather than provide pseudocode for the MST algorithm for

dependency parsing (which is provided in McDonald [24]), we show how the MST algorithm works using an example dependency parse using this algorithm.

Let us consider the following fully connected graph for the input sentence *John saw Mary*. The edges have weights based on some scoring function on edges (these scores come from various features that are computed on the edge, as explained in the next section).

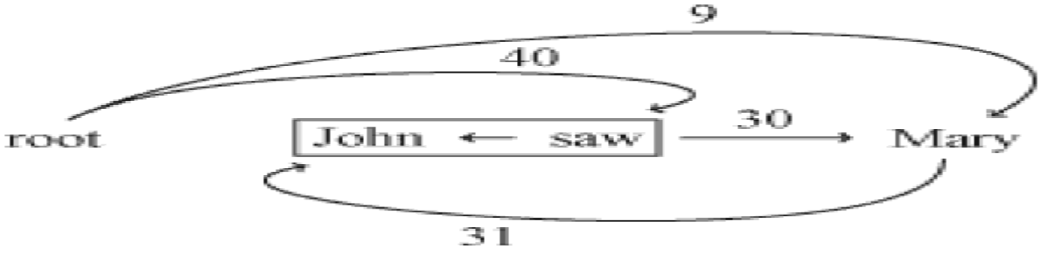


The first step is to find the highest scoring *incoming* edge. If this step results in a tree, then we report this as the parse because it would have to be an MST. In this example, however, after we pick only the highest scoring incoming edges from the graph, we do have a cycle.



We can contract the cycle into a single node, and we recalculate the edge weights. When we compute the edge weights from each node to that contracted node, we also have to keep track of which component of the merged node is the one with maximum weight. For example, in the preceding graph, we compare the

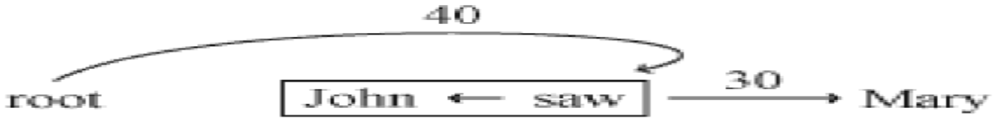
$Mary \longrightarrow \boxed{John \longrightarrow saw}$: wt $= 31$ to obtain the ones with maximum weight shown below:



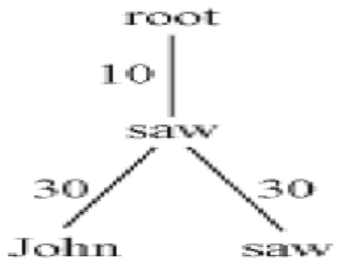
We now run the MST algorithm recursively on this graph, which means finding the graph with the best incoming edges to each word. In this case it means comparing

$root \longrightarrow Mary \longrightarrow \boxed{John\text{-}saw}$: wt $= 9 + 31$ with

$root \longrightarrow \boxed{John\text{-}saw} \longrightarrow Mary$: wt $= 40 + 30$ which results in the following graph:



Unwinding the recursive step provides us with the MST that is the highest scoring dependency parse of the input:



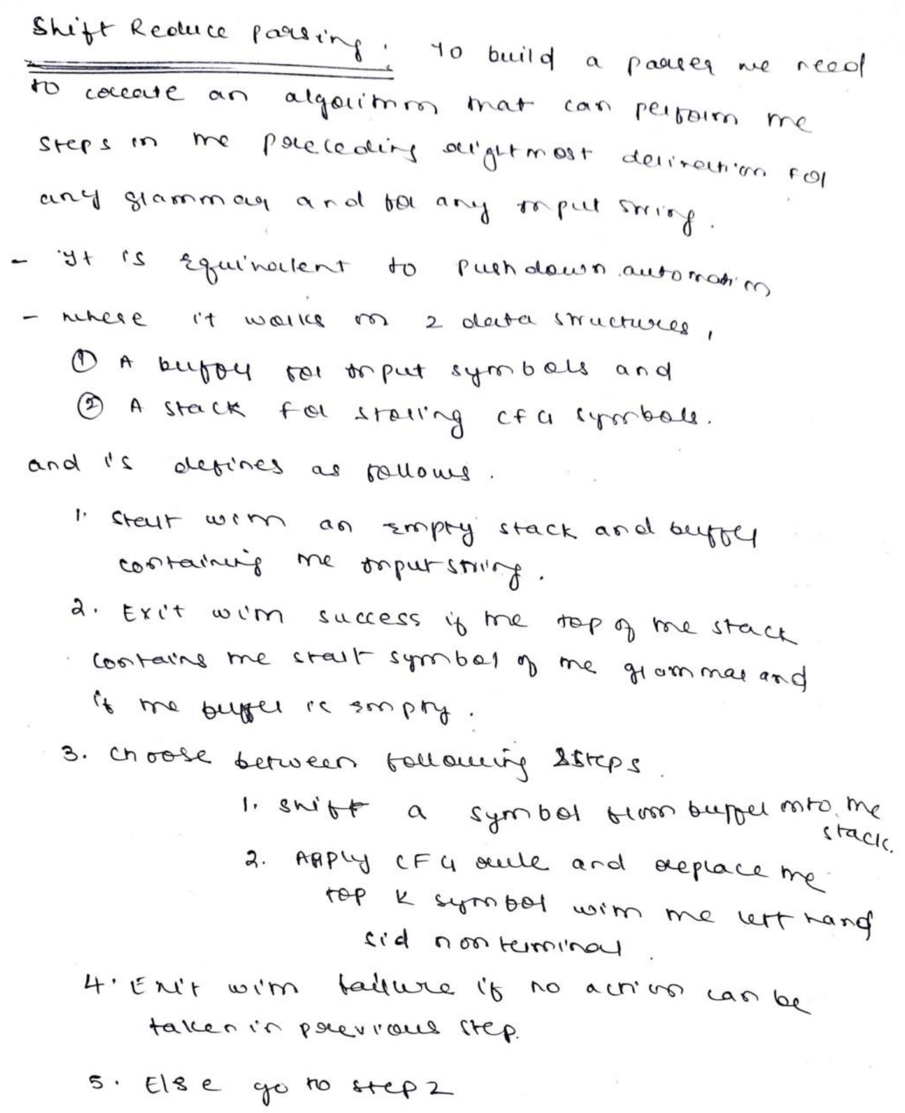
Various types of parsers

As discussed, a parser is basically a procedural interpretation of grammar. It finds an optimal tree for the given sentence after searching through the space of a variety of trees. Let us see some of the available parsers below −

1.Shift-reduce parser

Following are some important points about shift-reduce parser −

- It follows a simple bottom-up process.
- It tries to find a sequence of words and phrases that correspond to the right-hand side of a grammar production and replaces them with the left-hand side of the production.
- The above attempt to find a sequence of word continues until the whole sentence is reduced.
- In other simple words, shift-reduce parser starts with the input symbol and tries to construct the parser tree up to the start symbol.



Shift Reduce parsing : To build a parser we need to create an algorithm that can perform the steps in the preceding rightmost derivation for any grammar and for any input string.

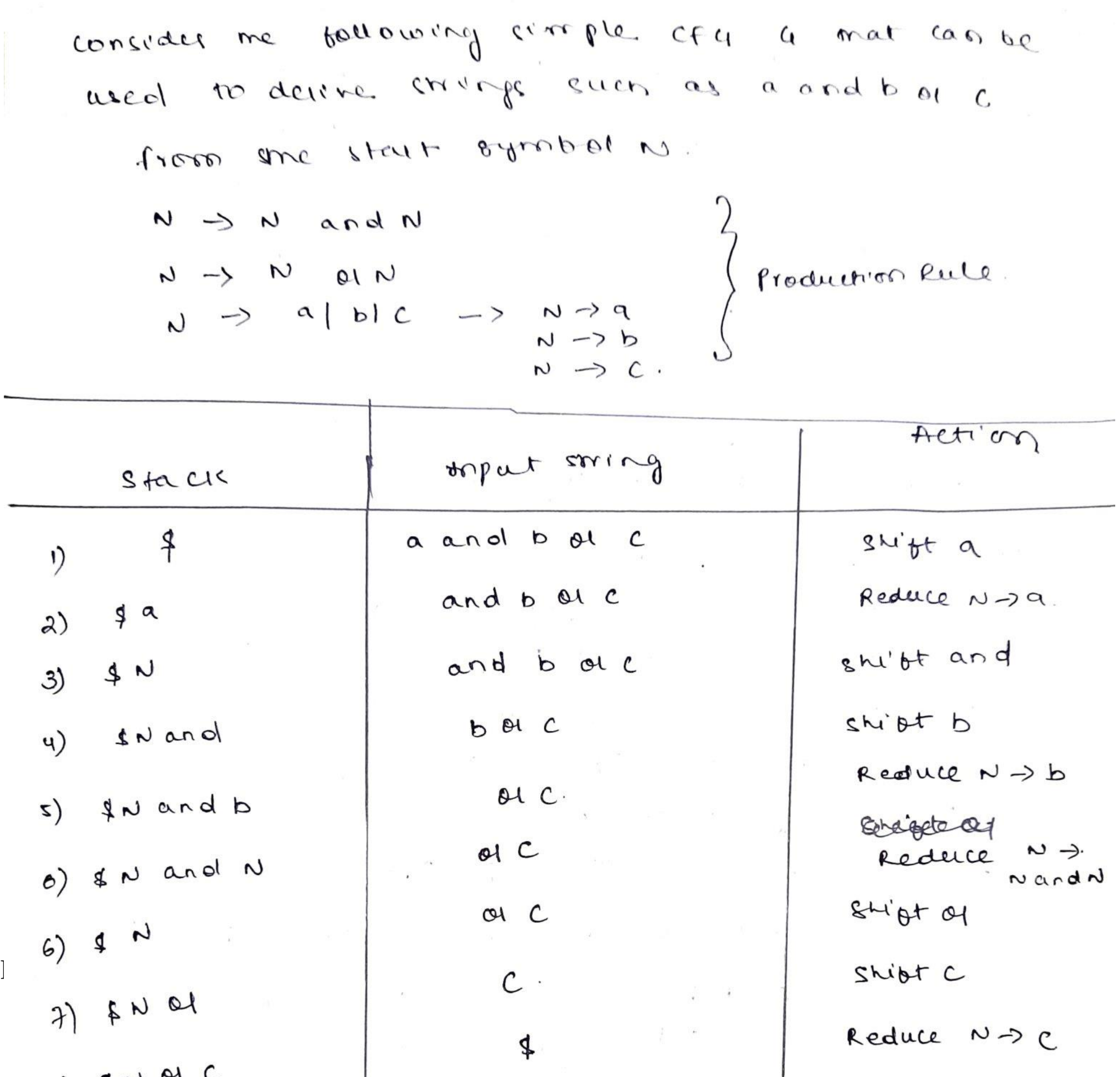
- It is equivalent to push down automation
- where it works on 2 data structures,

① A buffer for input symbols and

② A stack for storing CFG symbols.

and it's defines as follows.

1. Start with an empty stack and buffer containing the input string.
2. Exit with success if the top of the stack contains the start symbol of the grammar and if the buffer is empty.
3. Choose between following 2steps.
    1. shift a symbol from buffer onto the stack.
    2. Apply CFG rule and replace the top k symbol with the left hand sid non terminal.
4. Exit with failure if no action can be taken in previous step.
5. Else go to step 2



consider the following simple CFG G that can be used to derive strings such as a and b or c from the start symbol N.

N → N and N
N → N or N
N → a | b | c → N → a
N → b
N → c.

Production Rule.

Stack	input string	Action
1) $	a and b or c	Shift a
2) $ a	and b or c	Reduce N → a.
3) $ N	and b or c	shift and
4) $ N and	b or c	shift b
5) $ N and b	or c.	Reduce N → b
6) $ N and N	or c	~~Shift or~~ Reduce N → N and N
6) $ N	or c	Shift or
7) $ N or	c.	Shift c
$ N or c	$	Reduce N → c

2. Hypergraphs and chart Parser

Following are some important points about chart parser −

- It is mainly useful or suitable for ambiguous grammars, including grammars of natural languages.
- It applies dynamic programing to the parsing problems.
- Because of dynamic programing, partial hypothesized results are stored in a structure called a ‘chart’.
- The ‘chart’ can also be re-used.

The Representation and position is also been represented as tree parser
Similar way storage of the each word is been represented as Pat tree

PAT data structure (practical algorithm to retrieve information coded in alphanumeric)
PAT structure or PAT tree or PAT array : continuous text input data structures(string like N- Gram data structure).

The input stream is transformed into a searchable data structure consisting of substrings, all substrings are unique.

Each position in a input string is a anchor point for a sub string.

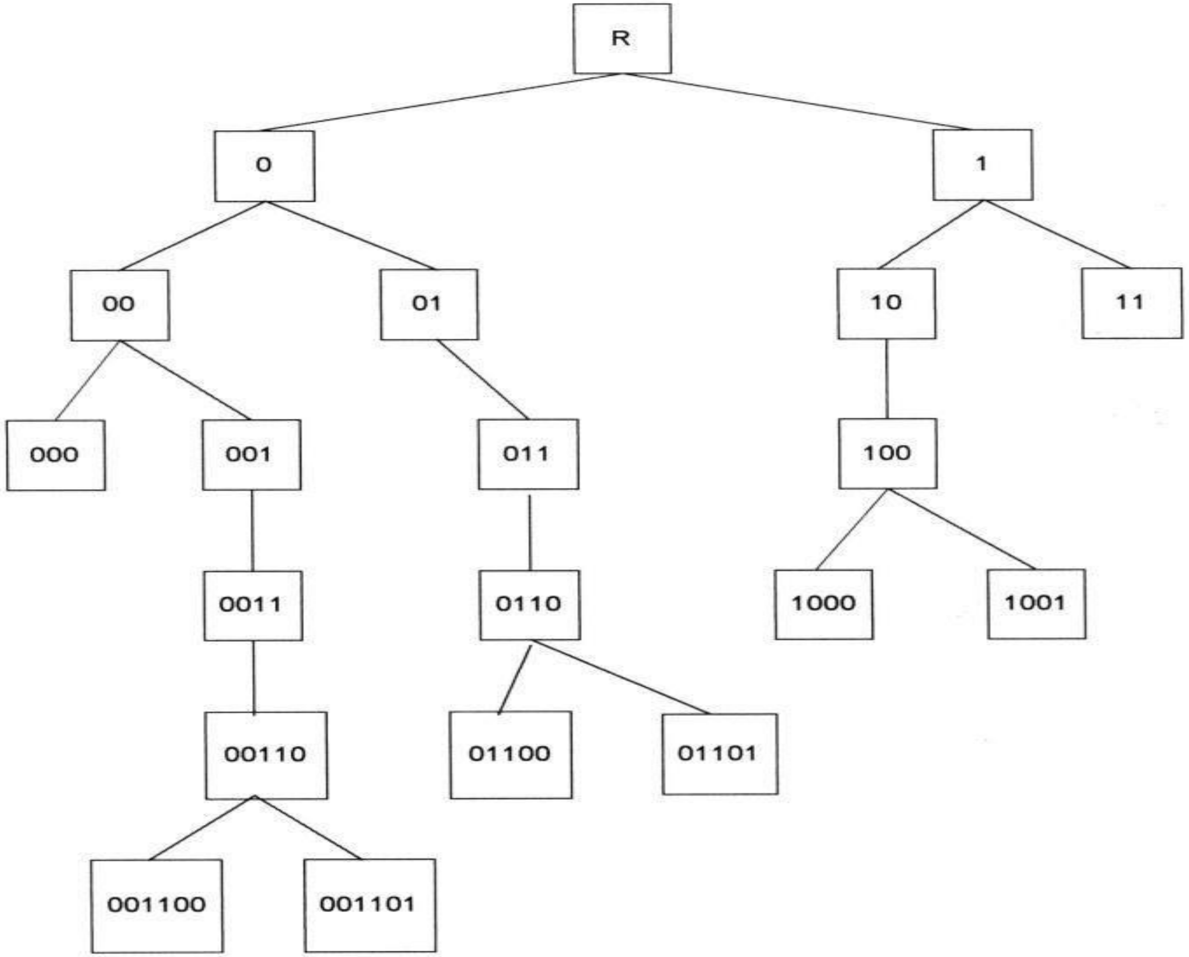
In creation of PAT trees each position in the input string is the anchor point for a sub-string that starts at that point and includes all new text up to the end of the input.
Binary tree, most common class for prefix search,But Pat trees are sorted logically which facilitate range search, and more accurate then inversion file .
PAT trees provide alternate structure if supporting strings search.

Examples of sistrings

- The key values are stored at the leaf nodes (bottom nodes) in the PAT Tree.
- For a text input of size “n” there are “n” leaf nodes and “n-1” at most higher level nodes.
- It is possible to place additional constraints on sistrings for the leaf nodes.

If the binary representations of “h” is (100), “o” is (110), “m” is (001) and “e” is (101) then the word “home” produces the input 100110001101.
Using the sistrings

The full PAT binary tree is


R
0
1
00
01
10
11
000
001
011
100
0011
0110
1000
1001
00110
01100
01101
001100
001101

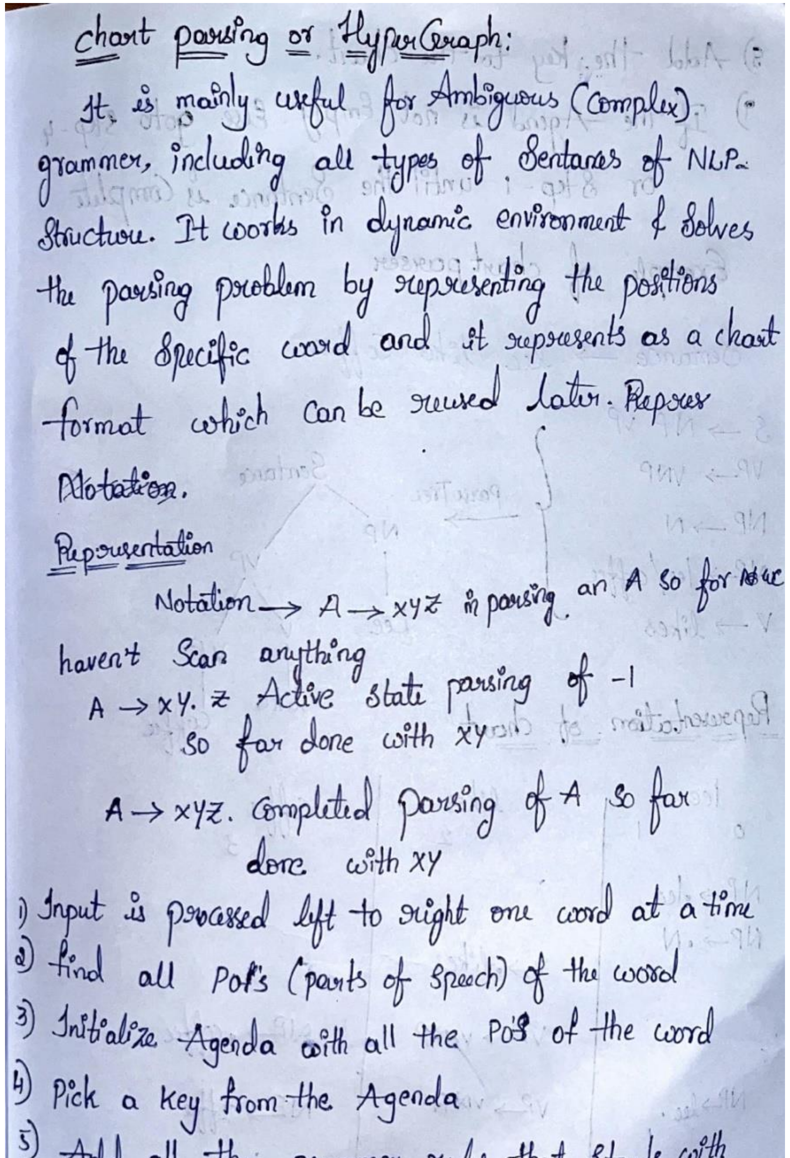


chart parsing or HyperGraph:

It is mainly useful for Ambiguous (Complex) grammer, including all types of Sentances of NLP. Structure. It works in dynamic environment & Solves the parsing problem by representing the positions of the Specific word and it represents as a chart format which can be reused later. Repres Notation.

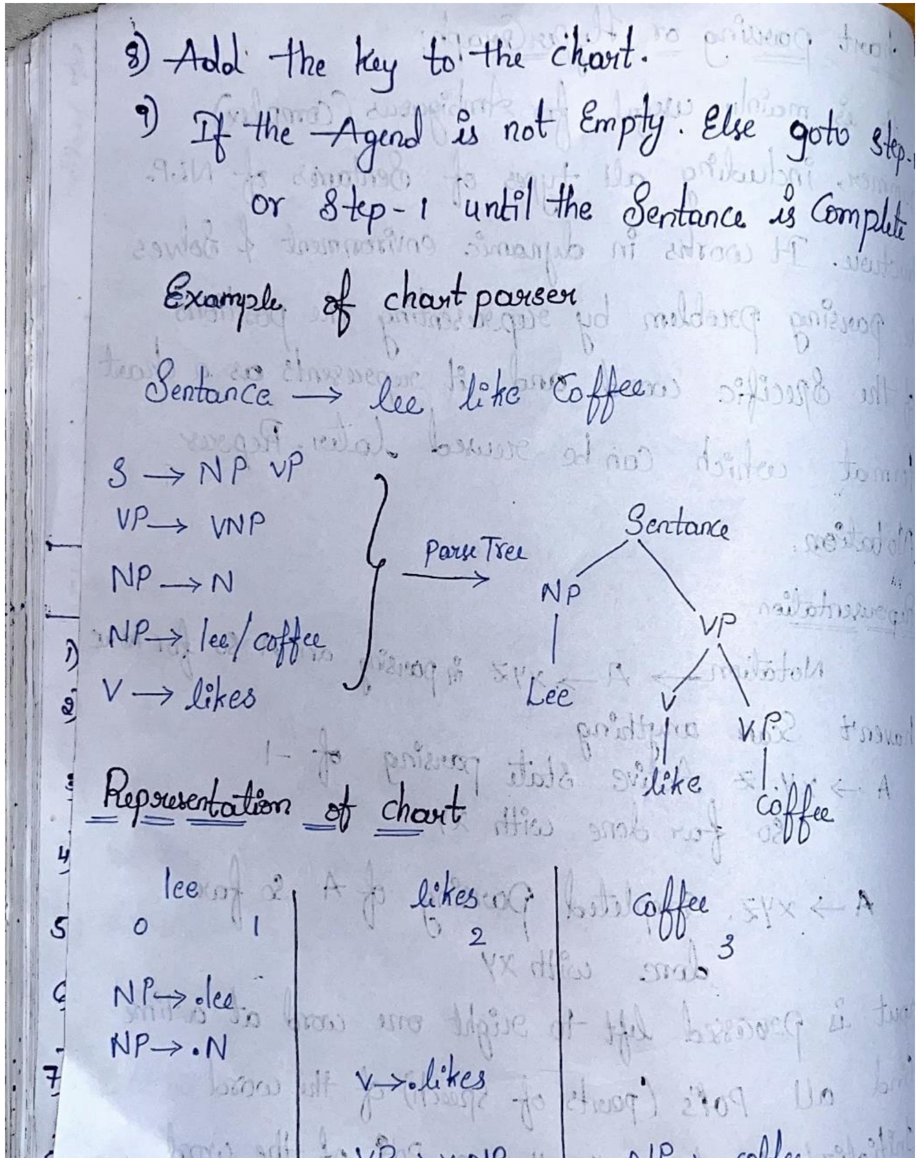
Representation

Notation → $A \to xyz$ in parsing an A so for we haven't Scan anything

$A \to xy.z$ Active state parsing of -1
so far done with xy

$A \to xyz.$ Completed Parsing of A so far done with xy

1) Input is processed left to right one word at a time
2) find all PoS's (parts of Speech) of the word
3) Initialize Agenda with all the PoS of the word
4) Pick a key from the Agenda
5) Add all the grammar rules that starts with



8) Add the key to the chart.

9) If the Agend is not Empty. Else goto step-
or Step-1 until the Sentance is Complete

Example of chart parser

Sentance → lee like Coffee

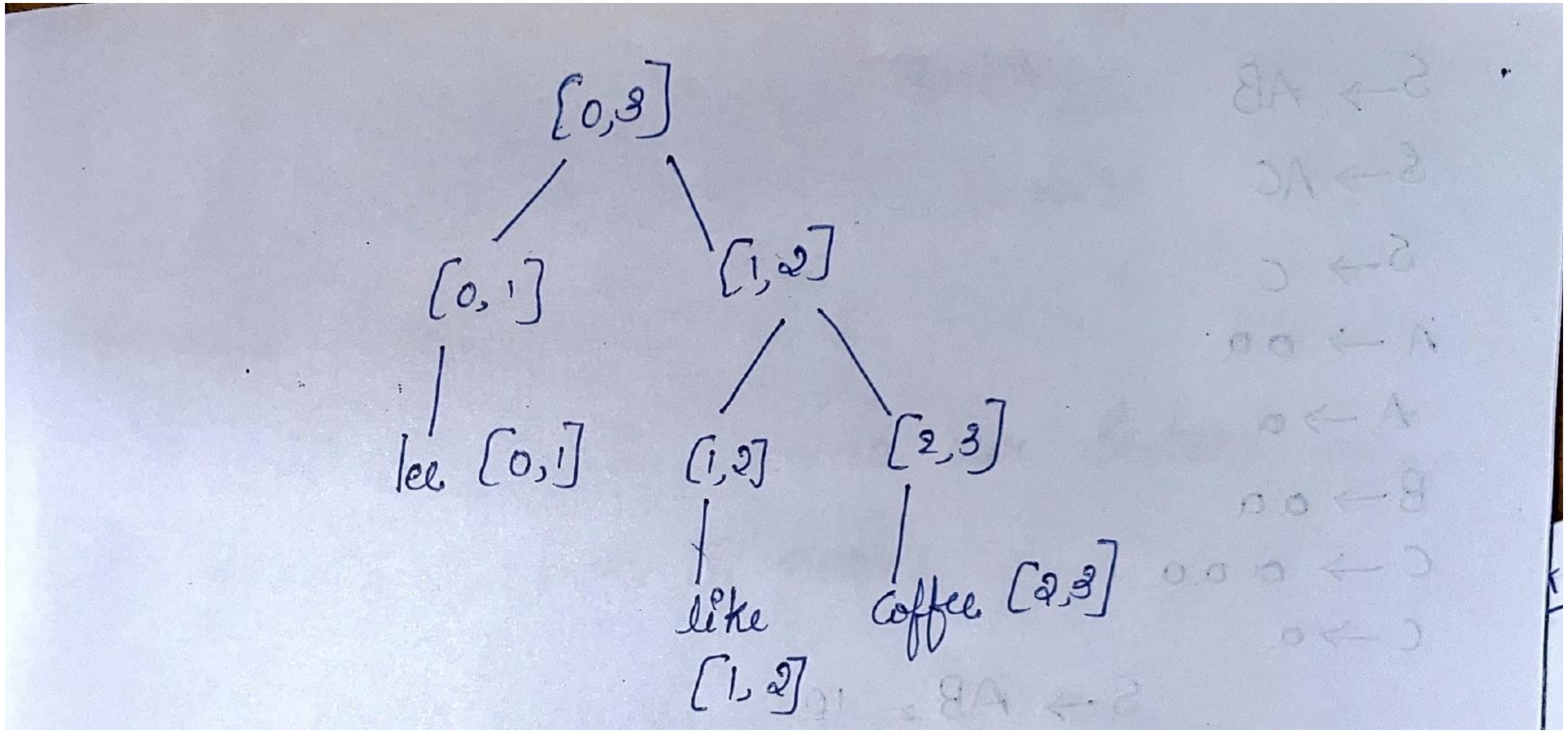
S → NP VP
VP → VNP
NP → N
NP → lee/coffee
V → likes

Parse Tree

Sentance
NP
VP
Lee
V
NP
like
Coffee

Representation of chart

lee 0 1	likes 2	coffee 3
NP→•lee NP→•N	V→•likes	
	VP→•VNP	NP→•coffee

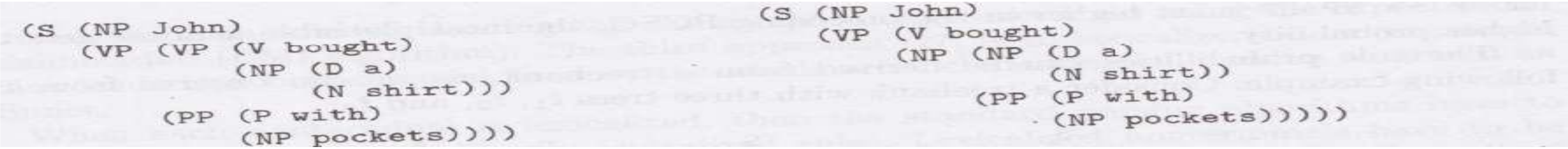

[0,3]
[0,1]
[1,2]
lee [0,1]
[1,2]
[2,3]
like
[1,2]
coffee [2,3]

Models for Ambiguity Resolution in Parsing

- Here we discuss on modelling aspects of parsing:
  A context-free grammar (CFG) is a list of rules that define the set of all well-formed sentences in a language. Each rule has a left-hand side, which identifies a syntactic category, and a right-hand side, which defines its alternative component parts, reading from left to right.
- how to evaluate design features and ways to resolve ambiguity in parsing which model to infer here we introduce concept of weights and known as PCFG and maximum probable tree would retrieved.
- A PCFG is a probabilistic version of a CFG where each production has a probability.
  Three Useful PCFG Tasks
  - Observation likelihood: To classify and order sentences.
  - Most likely derivation: To determine the most likely parse tree for a sentence.

Probabilistic context-free grammar Example

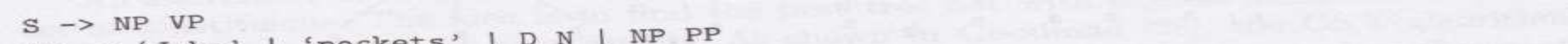
- Ex: *John bought a shirt with pockets*



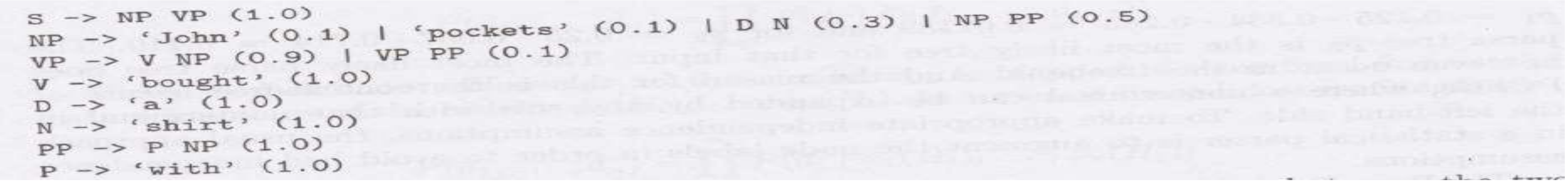
```
(S (NP John)
   (VP (VP (V bought)
           (NP (D a)
               (N shirt)))
       (PP (P with)
           (NP pockets))))
```

```
(S (NP John)
   (VP (V bought)
       (NP (NP (D a)
               (N shirt))
           (PP (P with)
               (NP pockets)))))
```

- Here we want to provide a model that matches the intuition that the second tree above ispreferred over the first.
- The parses can be thought of as ambiguous (leftmost to rightmost) derivation of the followingCFG:
- We assign scores or probabilities to the rules in CGF in order to provide a score or probabilityfor each derivation.



```
S -> NP VP
      'John' | 'pockets' | D N | NP PP
```



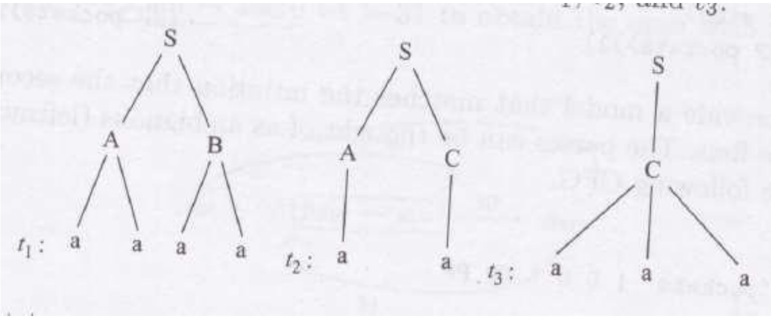
```
S -> NP VP (1.0)
NP -> 'John' (0.1) | 'pockets' (0.1) | D N (0.3) | NP PP (0.5)
VP -> V NP (0.9) | VP PP (0.1)
V -> 'bought' (1.0)
D -> 'a' (1.0)
N -> 'shirt' (1.0)
PP -> P NP (1.0)
P -> 'with' (1.0)
```

- From these rule probabilities, the only deciding factor for choosing between the two parses for John brought a shirt with pockets in the two rules NP->NP PP and VP-> VP PP. The probability higher between two rules been calculated and most probable tree is shortlisted

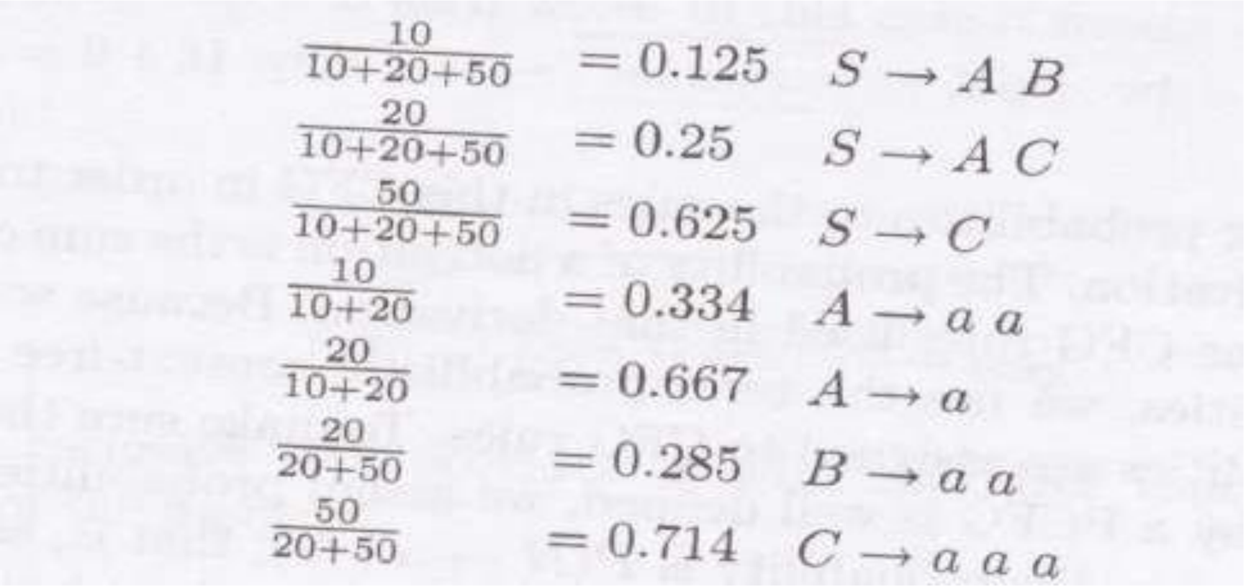
- Above example is complex to calculate the tree shown below is a example for PCFG calculation

- The rule probabilities can be derived from a treebank, consider a treebank with three

  tress t_1, t_2, t_3



- if we assume that tree t1 occurred 10 times in the treebank, t2 occurred 20 times and t3occurred 50 times, then the PCFG we obtain from this treebank is:



$$\frac{10}{10+20+50} = 0.125 \quad S \rightarrow A\ B$$

$$\frac{20}{10+20+50} = 0.25 \quad S \rightarrow A\ C$$

$$\frac{50}{10+20+50} = 0.625 \quad S \rightarrow C$$

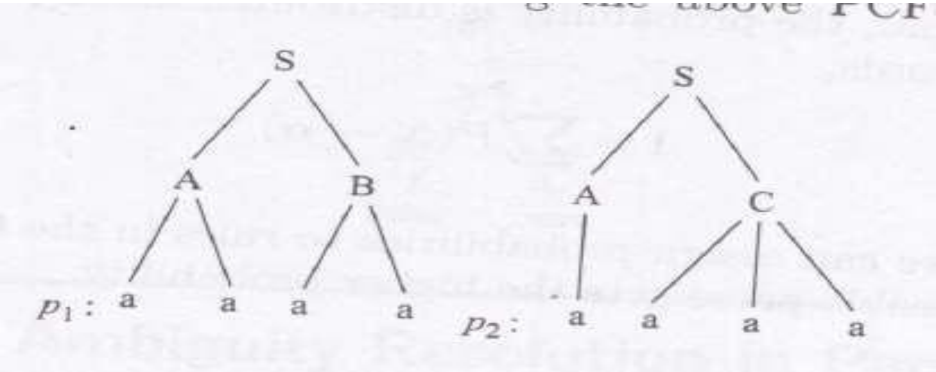
$$\frac{10}{10+20} = 0.334 \quad A \rightarrow a\ a$$

$$\frac{20}{10+20} = 0.667 \quad A \rightarrow a$$

$$\frac{20}{20+50} = 0.285 \quad B \rightarrow a\ a$$

$$\frac{50}{20+50} = 0.714 \quad C \rightarrow a\ a\ a$$

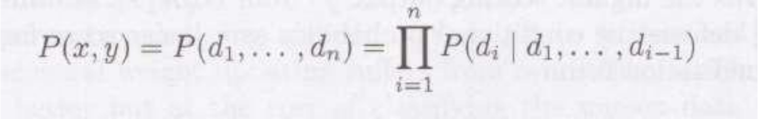
- For input a a a there are two parses using the above PCFG: the probability
- P1 =0.125 *0.334* 0.285 = 0.01189
- p2=0.25 *0.667 *0.714 =0.119.
- The parse tree p2 is the most likely tree for that input.



Generative models: A Generative Sequence of model tree for particular context is generated and is created according to the weights the particular parse tree is retrieved

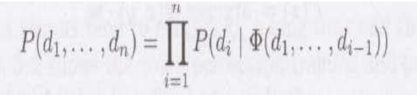
- To find the most plausible parse tree, the parser has to choose between the possible derivations each of which can be represented as a sequence of decisions.

- Let each derivation D = $d_1,d_2,\ldots..,d_n$, which is the sequence of decisions used to build theparse tree.
- Then for input sentence x, the output parse tree y is defined by the sequence of steps in thederivation.
- The probability for each derivation:

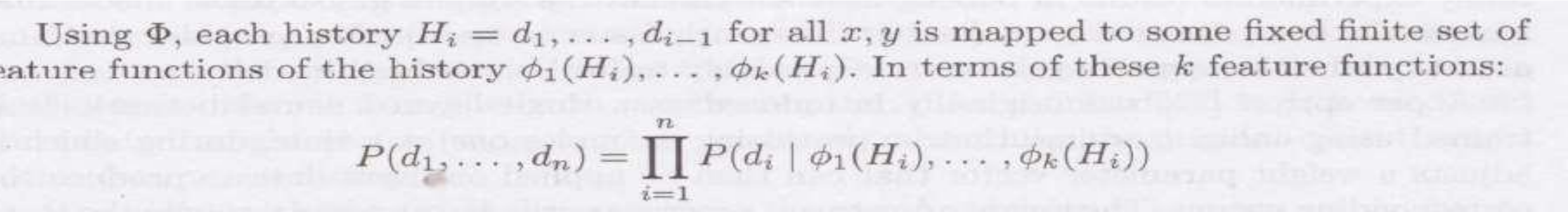


$$P(x,y) = P(d_1,\ldots,d_n) = \prod_{i=1}^{n} P(d_i \mid d_1,\ldots,d_{i-1})$$

- The conditioning context in the probability P(d_i|d_1
- We make a simplifying assumption that keeps the conditioning context to a finite set bygrouping the histories into equivalence classes using a function



$$P(d_1,\ldots,d_n) = \prod_{i=1}^{n} P(d_i \mid \Phi(d_1,\ldots,d_{i-1}))$$



Using Φ, each history $H_i = d_1, \ldots, d_{i-1}$ for all x, y is mapped to some fixed finite set of eature functions of the history $\phi_1(H_i), \ldots, \phi_k(H_i)$. In terms of these k feature functions:

$$P(d_1,\ldots,d_n) = \prod_{i=1}^{n} P(d_i \mid \phi_1(H_i),\ldots,\phi_k(H_i))$$

Depending on the score of the tree which is been evaluated with the help of

PCFG the next generative model or the tree sequence is being shortlisted Thus reducing the ambiguity

Discriminative models for Parsing: we can avoid or discriminate the tree based on the score levels for a particular topic

- Colins created a simple notation and framework that describes various discriminative

  approaches to learning for parsing .

  The discriminating or deleting the tree is based on the weights which is

  being calculated by PCFG

- This framework is called global linear model.
- Let x be a set of inputs and y be a set of possible outputs that can be a

  sequence of POS tags

or a parse tree or a dependency analysis.

- Each *x*Ɛx and *y*Ɛy is mapped to a d-dimensional feature vector ø(x,y), with each dimension

  being a real number.

- A weight parameter vector wƐR^d assigns a weight to each feature in ø(x,y), representing the

  importance of that feature.
- The value of ø(x,y).w is the score of (x,y) . The height the score, the more possible it is that y isthe output of x.
- The function GEN(x) generates the set of possible outputs y for a given x.
- Having ø(x,y).w and GEN(x) specified, we would like to choose the height scoring candidate

  y^* from GEN(x) as the most possible output

$$F(x) = \underset{y \in GEN(x)}{\operatorname{argmax}} \; p(y \mid x, \mathbf{w})$$

where F(x) returns the highest scoring output y^* from GEN(x)

- A conditional random field (CRF) defines the conditional probability is used in machine learning to calculations,as a linear score for each candidate

$$\log p(y \mid x, \mathbf{w}) = \Phi(x, y) \cdot \mathbf{w}$$

y and a global normalization term:

- **Conditional random fields (CRFs) are a class of statistical modeling**

methods often applied in pattern recognition and machine learning and used for structured prediction of parse tree Conditional Random Fields (CRF) CRF is a discriminant model for sequences data similar . It models the dependency between each state and the entire input sequences.

- A simple linear model that ignores the normalization term is:

$$F(x) = \underset{y \in GEN(x)}{\operatorname{argmax}} \; \Phi(x, y) \cdot \mathbf{w}$$

According to the probability score the parse tree is discriminated

Vyasapuri, Bandlaguda, Post:Keshavgiri
Hyderabad-500005,Telangana, India
Tel:040-29880079, 86,8978380692, 9642703342
9652216001, 9550544411, Website:www.mist.ac.in
Email:principal@mist.ac.in
principal.mahaveer@gmail.com
Counseling code:MHVR, University Code:E3

MAHAVEER
INSTITUTE OF SCIENCE & TECHNOLOGY
(AN UGC AUTONOMOUS INSTITUTION)
Approved by AICTE, Affiliated to JNTU,Hyderabad
Accredited by NAAC with 'A' Grade
Recognized Under 2(f) of UGC Act 1956,ISO 9001:2015 Certified

ESTD : 2001

Department of Computer Science and Engineering (AIML)

(R18)

Natural Language Processing

Lecture Notes

B. Tech III YEAR – II SEM

Prepared by

Mrs.Swapna
(Professor&HOD-CSM)
Dept. CSE(AIML)

Syllabus

NATURAL LANGUAGE PROCESSING

B.Tech. III Year II Sem.

L	T	P	C
3	1	0	4

Prerequisites: Data structures, finite automata and probability theory

Course Objectives:

- Introduce to some of the problems and solutions of NLP and their relation to linguistics and statistics.

Course Outcomes:

- Show sensitivity to linguistic phenomena and an ability to model them with formal grammars.
- Understand and carry out proper experimental methodology for training and evaluating empirical NLP systems
- Able to manipulate probabilities, construct statistical models over strings and trees, and estimate parameters using supervised and unsupervised training methods.
- Able to design, implement, and analyze NLP algorithms
- Able to design different language modeling Techniques.

UNIT - I

Finding the Structure of Words: Words and Their Components, Issues and Challenges, Morphological Models

Finding the Structure of Documents: Introduction, Methods, Complexity of the Approaches, Performances of the Approaches

UNIT - II

Syntax Analysis: Parsing Natural Language, Treebanks: A Data-Driven Approach to Syntax, Representation of Syntactic Structure, Parsing Algorithms, Models for Ambiguity Resolution in Parsing, Multilingual Issues

UNIT - III

Semantic Parsing: Introduction, Semantic Interpretation, System Paradigms, Word Sense Systems, Software.

UNIT - IV

Predicate-Argument Structure, Meaning Representation Systems, Software.

UNIT - V

Discourse Processing: Cohension, Reference Resolution, Discourse Cohension and Structure **Language Modeling:** Introduction, N-Gram Models, Language Model Evaluation, Parameter Estimation, Language Model Adaptation, Types of Language Models, Language-Specific Modeling Problems, Multilingual and Crosslingual Language

Modeling

TEXT BOOKS:

1. Multilingual natural Language Processing Applications: From Theory to Practice – Daniel M. Bikel and Imed Zitouni, Pearson Publication

Natural Language Processing and Information Retrieval: Tanvier Siddiqui, U.S. Tiwary

Unit 3 & 4

UNIT – III & UNIT IV

Semantic Parsing: Introduction, Semantic Interpretation, System Paradigms, Word Sense Systems, Software.
Predicate-Argument Structure, Meaning Representation Systems, Software.

SEMANTIC PARSING

A semantic system brings entities, concepts, relations and predicates together to provide more context to language so machines can understand text data with more accuracy. Semantic analysis derives meaning from language and lays the foundation for a semantic system to help machines interpret meaning.

Machines lack a reference system to understand the meaning of words, sentences and documents. Word sense disambiguation and meaning recognition can provide a better understanding of language data for machines. Here is how each part of semantic analysis work

Lexical analysis is the process of reading a stream of characters, identifying the lexemes and converting them into tokens that machines can read.

- **Grammatical analysis** correlates the sequence of lexemes (words) and applies formal grammar to them so part-of-speech tagging can occur.
- **Syntactical analysis** analyzes or parses the syntax and applies grammar rules to provide context to meaning at the word and sentence level.
- **Semantic analysis** uses all of the above to understand the meaning of words and interpret sentence structure so machines can understand language as humans do.

Why Is Semantic Analysis Important to NLP?

Language data is a rich source of business intelligence. However, many organizations struggle to capitalize on it because of their inability to analyze unstructured data. This challenge is a frequent roadblock for artificial intelligence (AI) initiatives that tackle language-intensive processes.

Every type of communication — be it a tweet, LinkedIn post, or review in the comments section of a website — may contain potentially relevant and even valuable information that companies must capture and understand to stay ahead of their competition. Capturing the information is the easy part but understanding what is being said (and doing this at scale) is a whole different story.

To understand how NLP and semantic processing work together, consider this:

- Basic NLP can identify words from a selection of text.
- Semantics gives meaning to those words in context (e.g., knowing an apple as a fruit rather than a company).

Semantics Makes Word Meaning Clear(WORD SENSE)

Semantic analysis, on the other hand, is crucial to achieving a high level of accuracy when analyzing text.

Consider the task of text summarization which is used to create digestible chunks of information from large quantities of text. Text summarization extracts words, phrases, and sentences to form a text summary that can be more easily consumed. The accuracy of the summary depends on a machine’s ability to understand language data.

Semantic Analysis Is Part of a Semantic System

A semantic system brings entities, concepts, relations and predicates together to provide more context to language so machines can understand text data with more accuracy. Semantic analysis derives meaning from language and lays the foundation for a semantic system to help machines interpret meaning.

To better understand this, consider the following elements of semantic analysis that help support language understanding:

- **Hyponymy**: A generic term.
- **Homonymy**: Two or more lexical terms with the same spelling and different meanings.
- **Polysemy**: Two or more terms that have the same spelling and similar meanings.
- **Synonymy**: Two or more lexical terms with different spellings and similar meanings.
- **Antonymyn**: A pair of lexical terms with contrasting meanings.
- **Metronome**: A relationship between a lexical term and a larger entity.

Understanding these terms is crucial to NLP programs that seek to draw insight from textual information, extract information and provide data. It is also essential for automated processing and question-answer systems like chat bots.

What is Word Sense Disambiguation?

Word Sense Disambiguation is an important method of NLP by which the meaning of a word is determined, which is used in a particular context. NLP systems often face the challenge of properly identifying words, and determining the specific usage of a word in a particular sentence has many applications.

Word Sense Disambiguation basically solves the ambiguity that arises in determining the meaning of the same word used in different situations.

Challenges in Word Sense Disambiguation

WSD faces a lot of challenges and problems.

- The most common problem is the difference between various dictionaries or text corpus. Different dictionaries have different meanings for words, which makes the sense of the words to be perceived as different. A lot of text information is out there and often it is not possible to process everything properly.
- Different applications need different algorithms and that is often a challenge for WSD.
- A problem also arises is that words cannot be divided into discrete meanings. Words often have related meanings and this causes a lot of problems.

Ease Semantic Analysis With Cognitive Platforms

Semantic analysis describes the process of machines understanding natural language as humans do based on meaning and context. Cognitive technology like that offered by expert.ai eases this process. How, you ask the query?

Expert.ai's rule-based technology starts by reading all of the words within a piece of content to capture its real meaning. It then identifies the textual elements and assigns them to their logical and grammatical roles. Finally, it analyzes the surrounding text and text structure to accurately determine the proper meaning of the words in context.

2.Semantic Interpretation

Semantic parsing can be considered as part of Semantic interpretation, which involves various components that together define a representation of text that can be fed into a computer to allow further computations manipulations and search, which are prerequisite for any language understanding system or application. Here we start with discus with structure of semantic theory.

A Semantic theory should be able to:

1.Explain sentence having ambiguous meaning: The bill is large is ambiguous in the sense that is could represent money or the beak of a bird.

2.Resolve the ambiguities of words in context. The bill is large but need not be paid, the theory should be able to disambiguate the monetary meaning of bill.

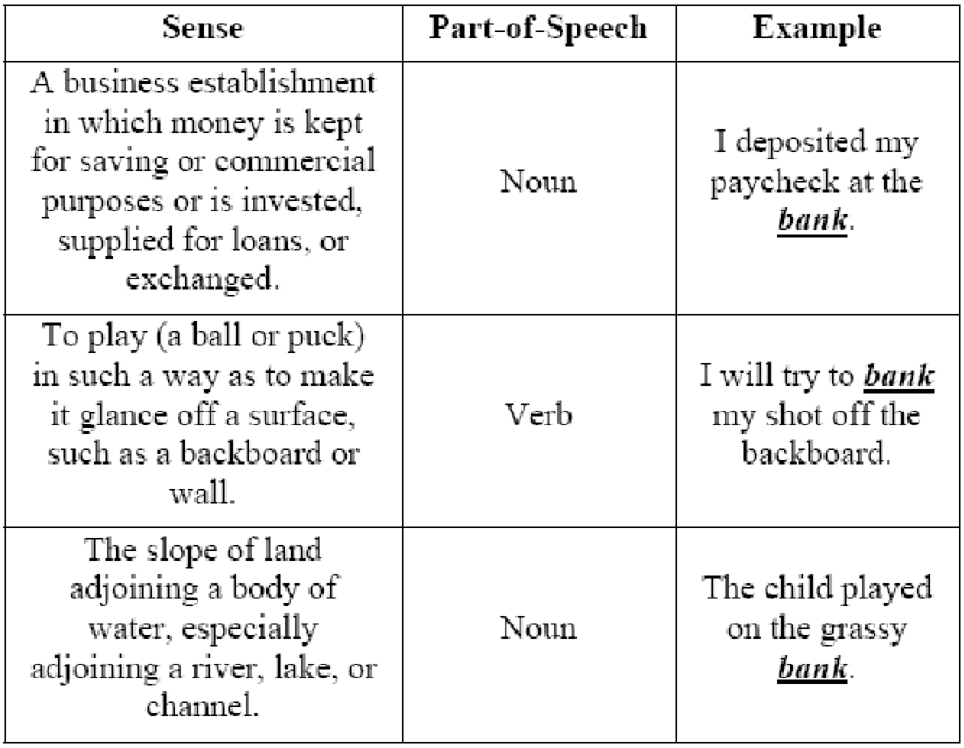
3.Identify meaningless sentence to syntactically well-formed sentence.

4.Identify syntactically or transformation ally unrelated paraphrases of concept having the same semantic content.

WORD SENSE EXAMPLES

In any given language, the same word type is used in different contexts and with different morphological variants to represent different entities or concepts in the world.

For example, we use the word **BANK** to represent a part of the human anatomy and also to represent the generally metallic object used to secure other objects.



Sense	Part-of-Speech	Example
A business establishment in which money is kept for saving or commercial purposes or is invested, supplied for loans, or exchanged.	Noun	I deposited my paycheck at the ***bank***.
To play (a ball or puck) in such a way as to make it glance off a surface, such as a backboard or wall.	Verb	I will try to ***bank*** my shot off the backboard.
The slope of land adjoining a body of water, especially adjoining a river, lake, or channel.	Noun	The child played on the grassy ***bank***.

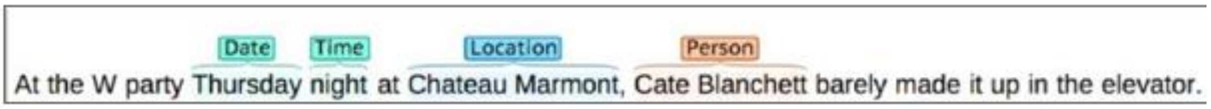
Table 1: Example of Word Sense Disambiguation for Bank

Entity and Event Resolution(Name Entity recognition and Co reference resolution):

Entities are the most important chunks of a particular sentence such as noun phrases, verb phrases, or both. Generally, Entity Detection algorithms are ensemble models of :

- Rule-based
- Dictionary lookups,
- POS Tagging,
- Dependency Parsing.

For Example,



At the W party Thursday night at Chateau Marmont, Cate Blanchett barely made it up in the elevator.

In the above sentence, the entities are:

Date: Thursday, Time: night, Location: Chateau Marmot, Person: Cate Blanchett

Now, we can start our discussion on Named Entity Recognition (NER),

1. Named Entity Recognition is one of the key entity detection methods in NLP.

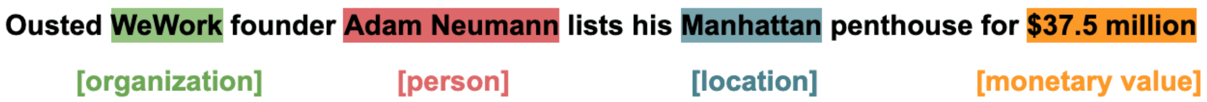
2. Named entity recognition is a natural language processing technique that can automatically scan entire articles and pull out some fundamental entities in a text and classify them into predefined categories. Entities may be,

- Organizations,
- Quantities,
- Monetary values,
- Percentages, and more.
- People’s names
- Company names
- Geographic locations (Both physical and political)
- Product names
- Dates and times
- Amounts of money
- Names of events

3. In simple words, Named Entity Recognition is the process of detecting the named entities such as person names, location names, company names, etc from the text.

4. It is also known as entity identification or entity extraction or entity chunking.

For Example,



Ousted WeWork founder Adam Neumann lists his Manhattan penthouse for $37.5 million

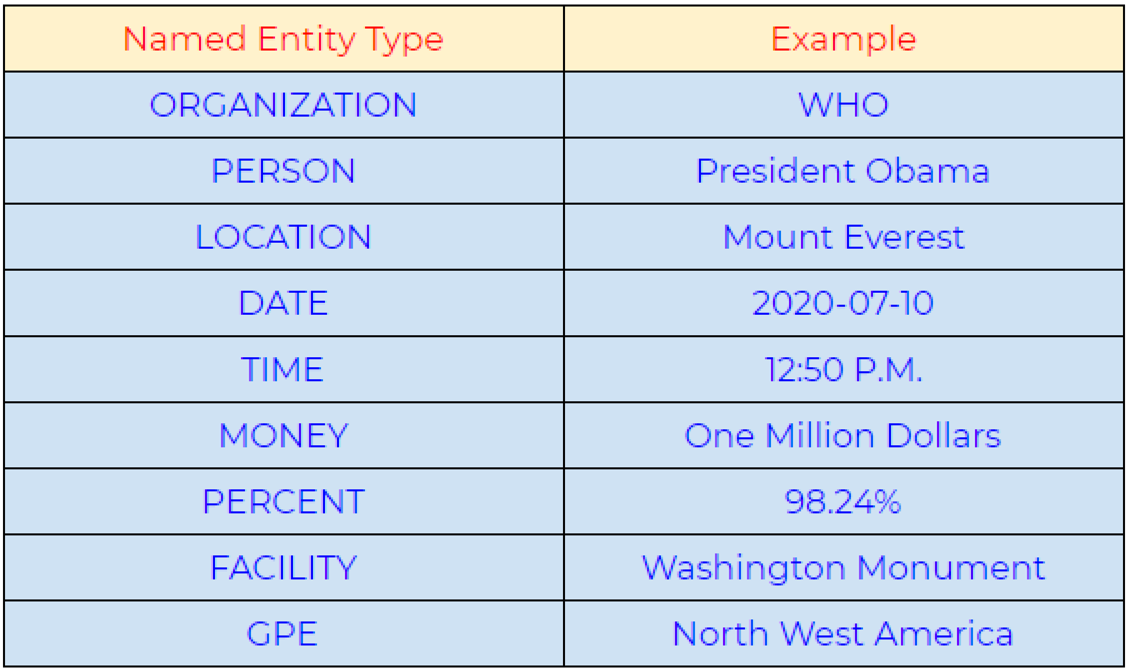
5. With the help of named entity recognition, we can extract key information to understand the text, or merely use it to extract important information to store in a database.

6. The applicability of entity detection can be seen in many applications such as

- Automated Chatbots,
- Content Analyzers,

- Consumer Insights, etc.

Commonly used types of named entity:



Named Entity Type	Example
ORGANIZATION	WHO
PERSON	President Obama
LOCATION	Mount Everest
DATE	2020-07-10
TIME	12:50 P.M.
MONEY	One Million Dollars
PERCENT	98.24%
FACILITY	Washington Monument
GPE	North West America

Different blocks present in a Typical Named Entity Recognition model

A typical NER model consists of the following three blocks:

Noun Phrase Identification

This step deals with extracting all the noun phrases from a text with the help of dependency parsing and part of speech tagging.

Phrase Classification

In this classification step, we classified all the extracted noun phrases from the above step into their respective categories. To disambiguate locations, Google Maps API can provide a very good path. and to identify person names or company names, the open databases from DBpedia, Wikipedia can be used. Apart from this, we can also make the lookup tables and dictionaries by combining information with the help of different sources.

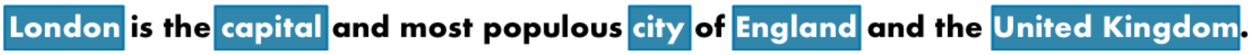
Entity Disambiguation

Sometimes what happens is that entities are misclassified, hence creating a validation layer on top of the results becomes useful. The use of knowledge graphs can be exploited for this purpose. Some of the popular knowledge graphs are:

- Google Knowledge Graph,
- IBM Watson,
- **Wikipedia**, etc.

Deep understanding of NER with an Example

Consider the following sentence:



London is the capital and most populous city of England and the United Kingdom.

The blue cells represent the nouns. Some of these nouns describe real things present in the world.

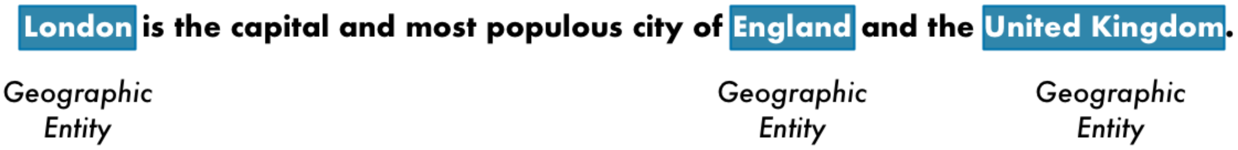
For Example, From the above, the following nouns represent physical places on a map.

“London”, “England”, “United Kingdom”

It would be a great thing if we can detect that! With that amount of information, we could automatically extract a list of real-world places mentioned in a document with the help of NLP.

Therefore, the goal of NER is to detect and label these nouns with the real-world concepts that they represent.

So, when we run each token present in the sentence through a NER tagging model, our sentence looks like as,



London is the capital and most populous city of England and the United Kingdom.

Let’s discuss what exactly the NER system does?

NER systems aren’t just doing a simple dictionary lookup. Instead, they are using the context of how a word appears in the sentence and used a statistical model to guess which type of noun that particular word represents.

Since NER makes it easy to grab structured data out of the text, therefore it has tons of uses. It’s one of the easiest methods to quickly get insightful value out of an NLP pipeline.

If you want to try out NER yourself, then refer to the link.

How does Named Entity Recognition work?

As we can simple observed that after reading a particular text, naturally we can recognize named entities such as people, values, locations, and so on.

For Example, Consider the following sentence:

Sentence: Sundar Pichai, the CEO of Google Inc. is walking in the streets of California.

From the above sentence, we can identify three types of entities: (Named Entities)

- (“person”: “Sundar Pichai”),
- (“org”: “Google Inc.”),
- (“location”: “California”).

But to do the same thing with the help of computers, we need to help them recognize entities first so that they can categorize them. So, to do so we can take the help of machine learning and Natural Language Processing (NLP).

Let’s discuss the role of both these things while implementing NER using computers:

- **NLP:** It studies the structure and rules of language and forms intelligent systems that are capable of deriving meaning from text and speech.
- **Machine Learning:** It helps machines learn and improve over time.

To learn what an entity is, a NER model needs to be able to detect a word or string of words that form an entity (e.g. California) and decide which entity category it belongs to.

So, as a concluding step we can say that the heart of any NER model is a two-step process:

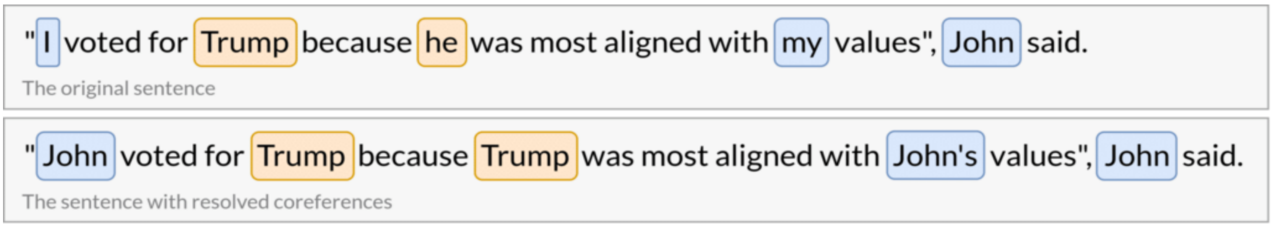
- Detect a named entity
- Categorize the entity

So first, we need to create entity categories, like Name, Location, Event, Organization, etc., and feed a NER model relevant training data.

Then, by tagging some samples of words and phrases with their corresponding entities, we’ll eventually teach our NER model to detect the entities and categorize them.

What is coreference resolution?

Coreference resolution (CR) is the task of finding all linguistic expressions (called mentions) in a given text that refer to the same real-world entity. After finding and grouping these mentions we can resolve them by replacing, as stated above, pronouns with noun phrases.



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

The original sentence

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

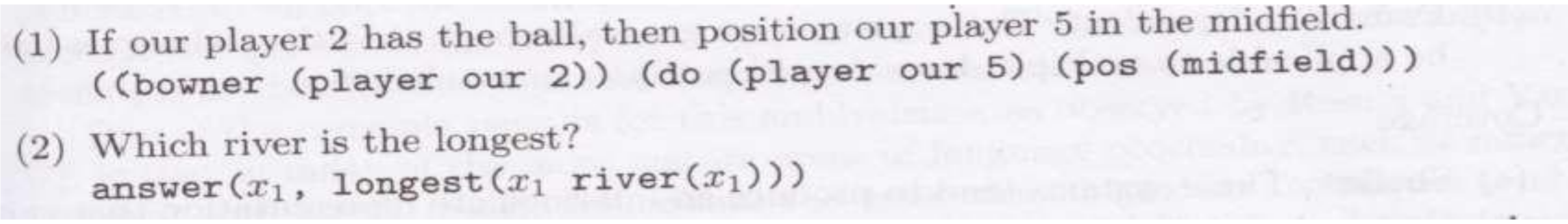
The sentence with resolved coreferences

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

Meaning Representation

The final process of the semantic interpretation is to build a semantic representation or meaning representation that can then be manipulated by algorithms to various application to better sense of word. This process is sometimes called the deep representation.

The following two examples



(1) If our player 2 has the ball, then position our player 5 in the midfield.
((bowner (player our 2)) (do (player our 5) (pos (midfield)))

(2) Which river is the longest?
answer(x_1, longest(x_1 river(x_1)))

The phrase "for every x" (sometimes "for all x") is called a universal quantifier and is denoted by ∀x. The phrase "there exists an x such that" is called an existential quantifier and is denoted by ∃x.

3.System Paradigms

It is important to get a perspective on the various primary dimensions on which the problem ofsemantic interpretation has been tackled.

The approaches generally fall into the following three categories: 1.System architecture 2.Scope 3. Coverage**.**

System Architectures

a.Knowledge based: These systems use a predefined set of rules or a knowledge base to obtain a solution to a new problem.

b.Supervised :

AI Chatbots and AI Virtual Assistants using Supervised Learning are trained using data that is well-labeled (or tagged). During training, those systems learn the best mapping function between known data input and the expected known output. Supervised NLP models then use the best approximating mapping learned during training to analyze unforeseen input data (never seen before) to accurately predict the corresponding output.

Usually, Supervised Learning models require extensive and iterative optimization cycles to adjust the input-output mapping until they converge to an expected and well-accepted level of performance. This type of learning keeps the word “supervised” because its way of learning from training data mimics the same process of a teacher supervising the end-to-end learning process. Supervised Learning models are typically capable of achieving excellent levels of performance but only when enough labeled data is available.

For example, a typical task delivered by a supervised learning model for AI chatbot / Virtual Assistants is the classification (via a variety of different algorithms like (Support Vector Machine, Random Forest, Classification Trees, etc.) of an input user utterance into a known class of user intents.

The precision achieved by those techniques is really remarkable though the shortfall is limited coverage of intent classes to only those for which labeled data is available for training.

c.Unsupervised Learning

To overcome the limitations of Supervised Learning, academia, and industry started pivoting towards the more advanced (but more computationally complex) **Unsupervised Learning** which promises effective learning using unlabeled data (no labeled data is required for training) and no human supervision (no data scientist or high-technical expertise is required). This is an important advantage compared to Supervised Learning, as unlabeled text in digital form is in abundance, but labeled datasets are usually expensive to construct or acquire, especially for common NLP tasks like PoS tagging or Syntactic Parsing.

Unsupervised Learning models are equipped with all the needed intelligence and automation to work on their own and automatically discover information, structure, and patterns from the data itself. This allows for the Unsupervised NLP to shine.

Advancing AI with Unsupervised Learning

The most popular applications of Unsupervised Learning in advanced AI chatbots / AI Virtual Assistants are clustering (like K-mean, Mean-Shift, Density-based, Spectral clustering, etc.) and association rules methods. Clustering is typically used to automatically group semantically similar user utterances together to accelerate the derivation and verification of an underneath common user intent (notice derivation of a new class, not classification into an existing class).

Unsupervised Learning is also used for association rules mining which aims at discovering relationships between features directly from data. This technique is typically used to automatically extract existing dependencies between named entities from input user utterances, dependencies of intents across a set of user utterances part of the same user/system session, or dependencies of questions and answers from conversational logs capturing the interactions between users and live agents during the problem troubleshooting process.

2.Scope:

a.**Domain Dependent**: These systems are specific to certain domains, such as air travel reservations or simulated football coaching.

b.**Domain Independent**: These systems are general enough that the techniques can be applicable to multiple domains without little or no change.

3.Coverage:

a.**Shallow**: These systems tend to produce an intermediate representation that can then beconverted to one that a machine can base its action on.

b. **Deep**: These systems usually create a terminal representation that is directly consumed by a machine or application.

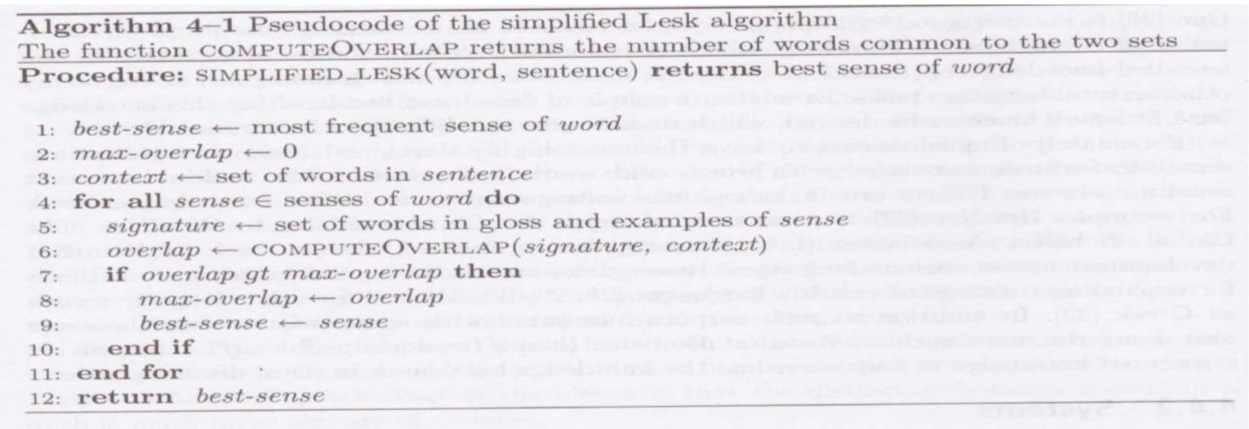
1.Knowledge based Or Rule based

rule based or knowledge based there use dictionary definitions of senses of words.

Much of this information is historical and cannot readily be translated and made available for building systems today. But some of techniques and algorithms are still available.

The simplest and oldest dictionary based sense disambiguation algorithm was introduced byleak author .

The core of the algorithm is that the dictionary sense whose terms most closely overlap withthe terms in the context.



Algorithm 4–1 Pseudocode of the simplified Lesk algorithm

The function COMPUTEOVERLAP returns the number of words common to the two sets

Procedure: SIMPLIFIED_LESK(word, sentence) **returns** best sense of *word*

```
best-sense ← most frequent sense of word
max-overlap ← 0
context ← set of words in sentence
for all sense ∈ senses of word do
    signature ← set of words in gloss and examples of sense
    overlap ← COMPUTEOVERLAP(signature, context)
    if overlap gt max-overlap then
        max-overlap ← overlap
        best-sense ← sense
    end if
end for
return best-sense
```

The word is being analysed and best sense of the word is compared to the database(signature) of the word is being matched with the exact gloss(scenario) of the word. And the word is being sensed and retrieved accordingly.
Example of gloss referring to correct context

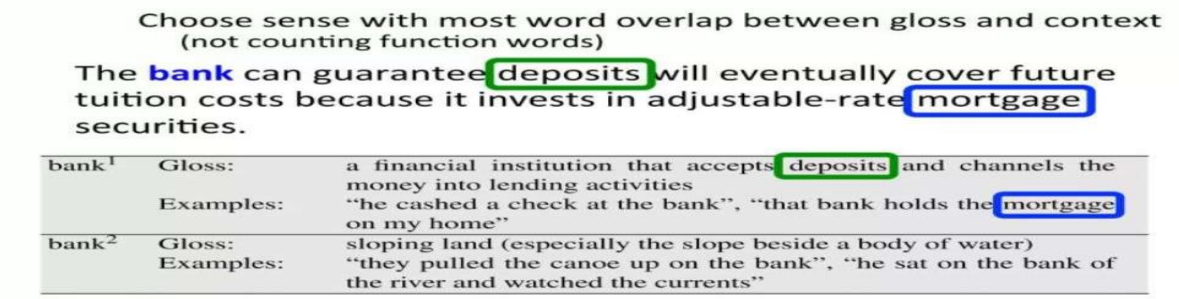
The word is mapped to the signature file structure present in the database..given below is signature file structure.

Signature file structure

- The coding is based upon words in the code.
- The words are mapped into word signatures .
- A word signature is fixed length code with a fixed number of bits set to 1.
- The bit positions that are set to one are determined via a hash function of the word.
- The word signatures are Ored together to create signature of an item..
- Partitioning of words is done in block size ,Which is nothing but set of words, Code length is 16 bits .

Search is accomplished by template matching on the bit position

- The block size is set at five words, the code length is 16 bits and the number of bits that are allowed to be "1" for each word is five.
- TEXT: Computer Science graduate students study (assume block size is five words)



Choose sense with most word overlap between gloss and context (not counting function words)

The **bank** can guarantee deposits will eventually cover future tuition costs because it invests in adjustable-rate mortgage securities.

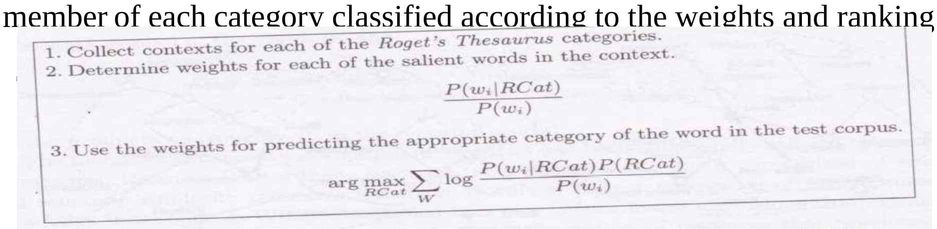
bank^1	Gloss:	a financial institution that accepts deposits and channels the money into lending activities
	Examples:	"he cashed a check at the bank", "that bank holds the mortgage on my home"
bank^2	Gloss:	sloping land (especially the slope beside a body of water)
	Examples:	"they pulled the canoe up on the bank", "he sat on the bank of the river and watched the currents"

graduate	1000 0101 0100 0010
students	0000 0111 1000 0100
study	0000 0110 0110 0100
--	
Block Signature	1001 0111 1110 0110

Application(s)/Advantage(s)

- Signature files provide a practical solution for storing and locating information in a number of different situations.
- Signature files have been applied as medium size databases, databases with low frequency of terms, WORM devices, parallel processing machines, and distributed environments

- Another dictionary based algorithm was suggested Yarowsky author .
- This study used Roget's Thesaurus categories and classified unseen words into one of these 1042 categories based on a statistical analysis of 100 word concordances for each member of each category classified according to the weights and ranking .



1. Collect contexts for each of the *Roget's Thesaurus* categories.
2. Determine weights for each of the salient words in the context.

$$\frac{P(w_i|RCat)}{P(w_i)}$$

3. Use the weights for predicting the appropriate category of the word in the test corpus.

$$\arg\max_{RCat} \sum_{W} \log \frac{P(w_i|RCat)P(RCat)}{P(w_i)}$$

Figure 4–2: Algorithm for disambiguatimg words into *Roget's Thesaurus* categories

- The first step is a collection of contexts.

The second step computes weights for each of the salient words.
P(w|Rcat) is the probability of a word w occurring in the context of a Roget's Thesaurus category Rcat.
P(w|Rcat) |Pr(w) , the probability of a word (w) appearing in the context of a Roget category divided by its overall probability in the corpus.

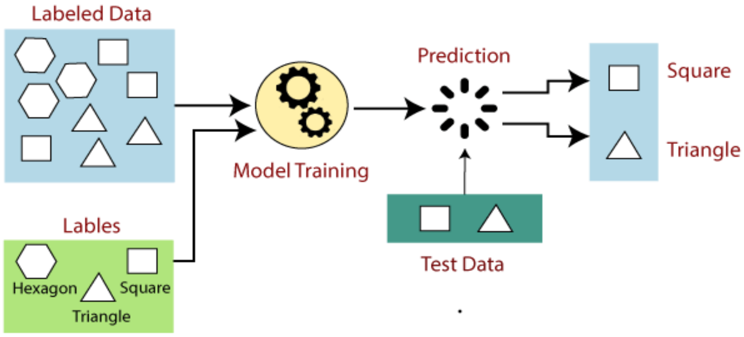
Finally, in third step, the unseen words in the test set are classified into the classified into the category that has the maximum weight and according to the Rank the information is retrieved.

The disambiguating which word to retrieved is categorized according to the weight and ranking which is being calculated with rogets formula and concept /thesaurus is generated for the specific word

2.Supervised Learning

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

The working of Supervised learning can be easily understood by the below example and diagram:



Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a **Square**.
- If the given shape has three sides, then it will be labelled as a **triangle**.
- If the given shape has six equal sides then it will be labelled as **hexagon**.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape.

The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset, test dataset, and validation dataset**.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Text Classification is an automated process of classification of text into predefined categories. We can classify Emails into spam or non-spam, news articles into different categories and clusters like Politics, Stock Market, Sports, etc.

This can be done with the help of Natural Language Processing and different Classification Algorithms like Naive Bayes, **Support Vector Machine** and even Neural Networks in Python.These are the common attributes in supervised learning for nlp woed sensing

a)Features: Here we discuss a more commonly found subset of features that have been useful in supervised learning of word sense.

b)**Lexical context**: The feature comprises the words and lemma of words occurring in the entireparagraph or a smaller window of usually five words.

Form	Stem	Lemma

Studies	Studi	Study
Studying	Study	Study
Beautiful	Beauty	beautiful
Beautifully	beauti	beautiful

c)Parts of speech : the feature comprises the POS information for words in the surrounding the word that is being sense tagged according to the suitable parts of speech .

d)Bag of words context: this feature comprises using an unordered set of words in the context
Window which is being properly classified.

e)Local Collections : Local collections are an ordered sequence of phrases near the target word that provide semantic context for disambiguation. Usually, a very small window of about three tokens on each side of the target word, most often in contiguous pairs or triplets, are added as a list of features.

f)Syntactic relations: if the parse of the sentence containing the target word is available, then we can use syntactic features.

g)**Topic features**: The board topic, or domain, of the article that word belongs to is also a good indicator of what sense of the word might be most frequent to that specific domain.

The **chen and palmer author** noticed the word sensed creates diambigutiy and confusion in sensing the word when it is unable identify the

1.user voice for the sentence whether is a passive or active
Example
active voice: She prepares dinner.
Passive voice: The dinner is prepared by her.

Active voice: She knows him.
Passive voice: He is known to her.

2.. Presence of subject/ object: unble to identify the subject when given large amount of training data

3.Sentential complement: Sentential complementation is a kind of sentence in which one of the arguments of a verb is a clause. That clausal argument is called a complement clause.
examples
The term *complement clause* is extended by some analysts to include clauses selected by nouns or adjectives.

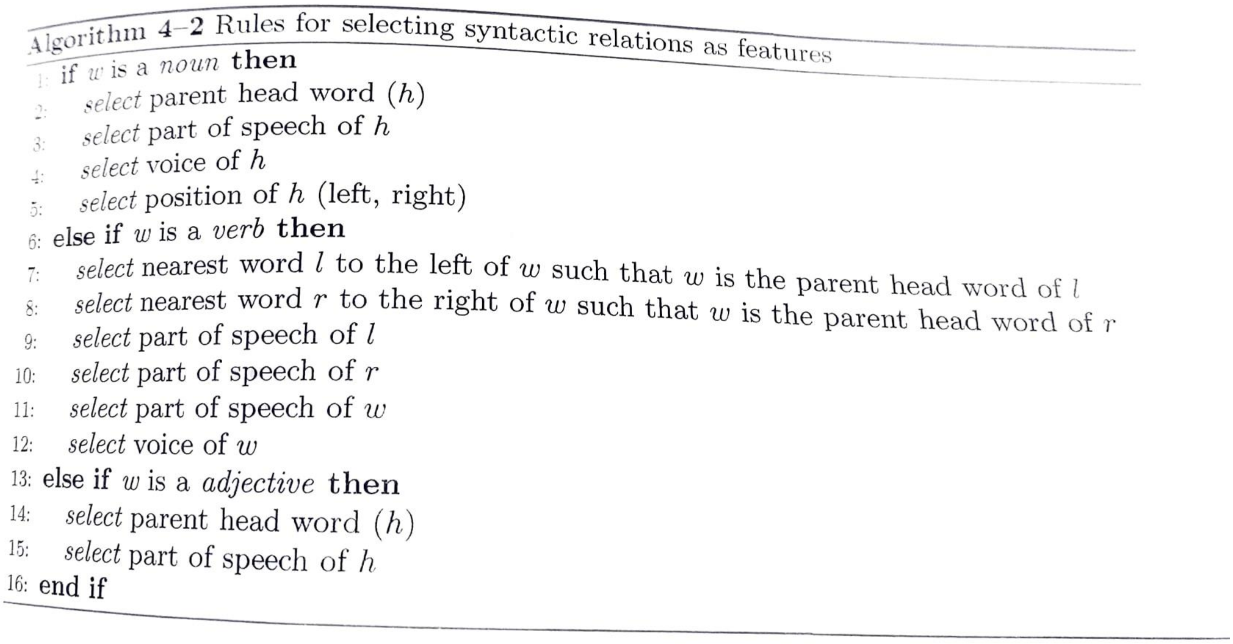
Examples:

- I heard the evidence *that he did it*.
- I am sure *that he did it*.
- I am not certain *what we did*.

4.Prepostional Pharse Adjunct:
An adjunct is any adverb, adverbial clause, adverbial phrase or prepositional phrase that gives more information primarily about the action in the sentence.

5.Name Entity: Identifying the correct domain for the word entity

6.Wordnet: Refering proper wordnet for the specific word..Ex hymonys,antonym and memonym etc



Algorithm 4–2 Rules for selecting syntactic relations as features

```
if $w$ is a noun then
    select parent head word ($h$)
    select part of speech of $h$
    select voice of $h$
    select position of $h$ (left, right)
else if $w$ is a verb then
    select nearest word $l$ to the left of $w$ such that $w$ is the parent head word of $l$
    select nearest word $r$ to the right of $w$ such that $w$ is the parent head word of $r$
    select part of speech of $l$
    select part of speech of $r$
    select part of speech of $w$
    select voice of $w$
else if $w$ is a adjective then
    select parent head word ($h$)
    select part of speech of $h$
end if
```

Above mention are the rules for selecting the domain of the word and concept thesuaras is generated accordingly with pos tags and proper word is being retrieved.

3.Unsupervised learning:

Unsupervised Learning which promises effective learning using unlabeled data (no labeled data is required for training) and no human supervision (no data scientist or high-technical expertise is required). This is an important advantage compared to Supervised Learning, as unlabeled text in digital form is in abundance, but labeled datasets are usually expensive to construct or acquire, especially for common NLP tasks like PoS tagging or Syntactic Parsing.

Unsupervised Learning models are equipped with all the needed intelligence and automation to work on their own and automatically discover information, structure, and

Advancing AI AND NLP with Unsupervised Learning

The most popular applications of Unsupervised Learning in advanced AI chatbots / AI Virtual Assistants are clustering (like K-mean, Mean-Shift, Density-based, Spectral clustering, etc patterns from the data itself. This allows for the Unsupervised NLP to shine and association rules methods.

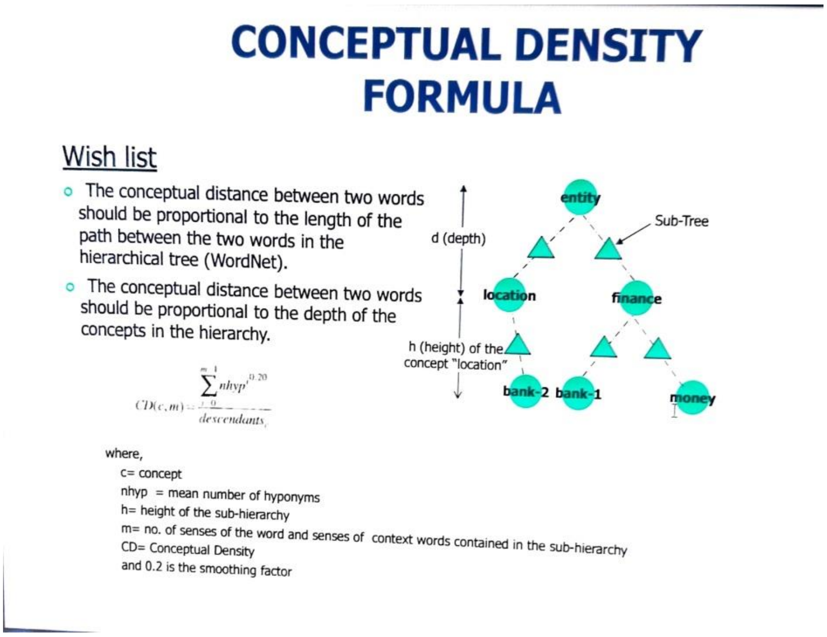
Clustering is typically used to automatically group semantically similar user utterances together to accelerate the derivation and verification of an underneath common user intent (notice derivation of a new class, not classification into an existing class).

Unsupervised Learning is also used for association rules mining which aims at discovering relationships between features directly from data. This technique is typically used to automatically extract existing dependencies between named entities from input user utterances, dependencies of intents across a set of user utterances part of the same user/system session, or dependencies of questions and answers from conversational logs capturing the interactions between users and live agents during the problem troubleshooting process.

Even though the benefits and level of automation brought by Unsupervised Learning are large and technically very intriguing, Unsupervised Learning, in general, is less accurate and trustworthy compared to Supervised Learning. Indeed, the most advanced AI Chatbot / AI Virtual Assistant technologies in the market strive by achieving the right level of balance between the two technologies, which when exploited correctly can deliver the accuracy and precision of Supervised Learning (tasks for which labeled data is available) coupled with the self-automation of unsupervised learning (tasks for which no labeled data is available).

In Unsupervised learning for the specific topic domain the depth of tree is calculated by concept density (CD) or conceptual density

The formula is given below



CONCEPTUAL DENSITY FORMULA

Wish list

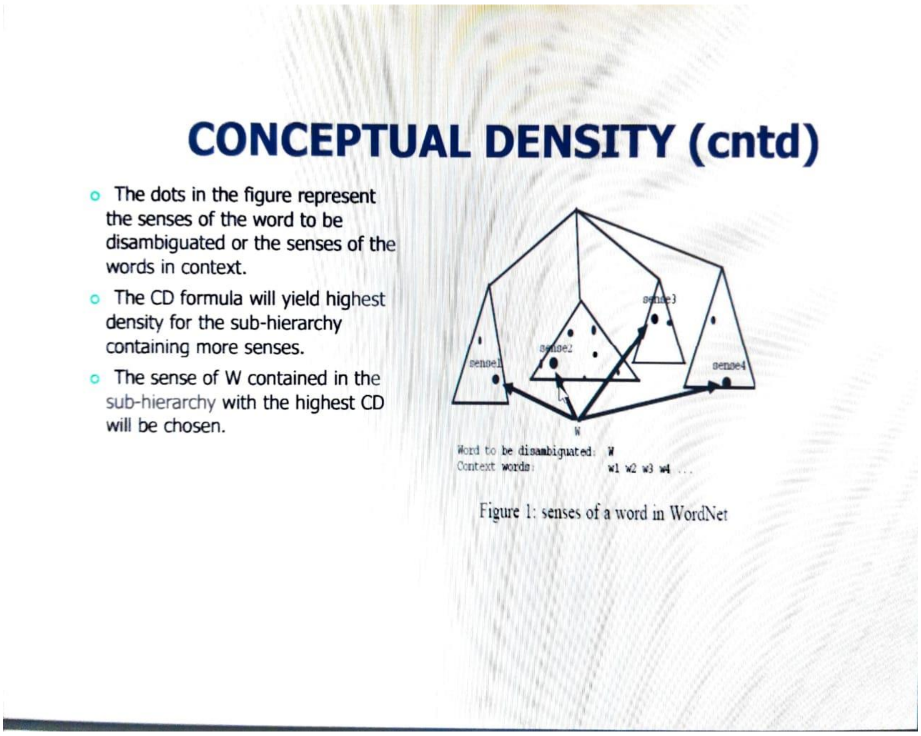
- The conceptual distance between two words should be proportional to the length of the path between the two words in the hierarchical tree (WordNet).
- The conceptual distance between two words should be proportional to the depth of the concepts in the hierarchy.

$$CD(c,m) = \frac{\sum_{i=0}^{m-1} nhyp^{i^{0.20}}}{descendants_c}$$

where,

c= concept
nhyp = mean number of hyponyms
h= height of the sub-hierarchy
m= no. of senses of the word and senses of context words contained in the sub-hierarchy
CD= Conceptual Density
and 0.2 is the smoothing factor

The depth of the tree is being examined and best word sense is being retrieved and the disambiguity is cleared by examining the depth of the word by refereeing to sources like word net(antonyms synonyms and metonyms).

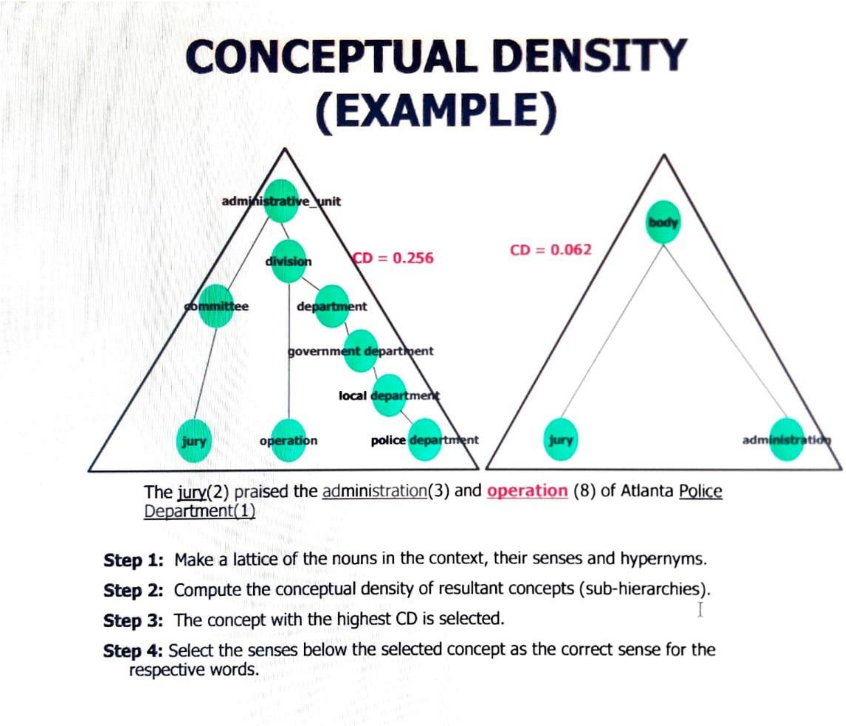


CONCEPTUAL DENSITY (cntd)

- The dots in the figure represent the senses of the word to be disambiguated or the senses of the words in context.
- The CD formula will yield highest density for the sub-hierarchy containing more senses.
- The sense of W contained in the sub-hierarchy with the highest CD will be chosen.

Figure 1: senses of a word in WordNet

Conceptual density is calculated by above formula . The tree which is having highest density is being referred and best sense of the word is being retrieved



CONCEPTUAL DENSITY (EXAMPLE)

The jury(2) praised the administration(3) and operation (8) of Atlanta Police Department(1)

Step 1: Make a lattice of the nouns in the context, their senses and hypernyms.

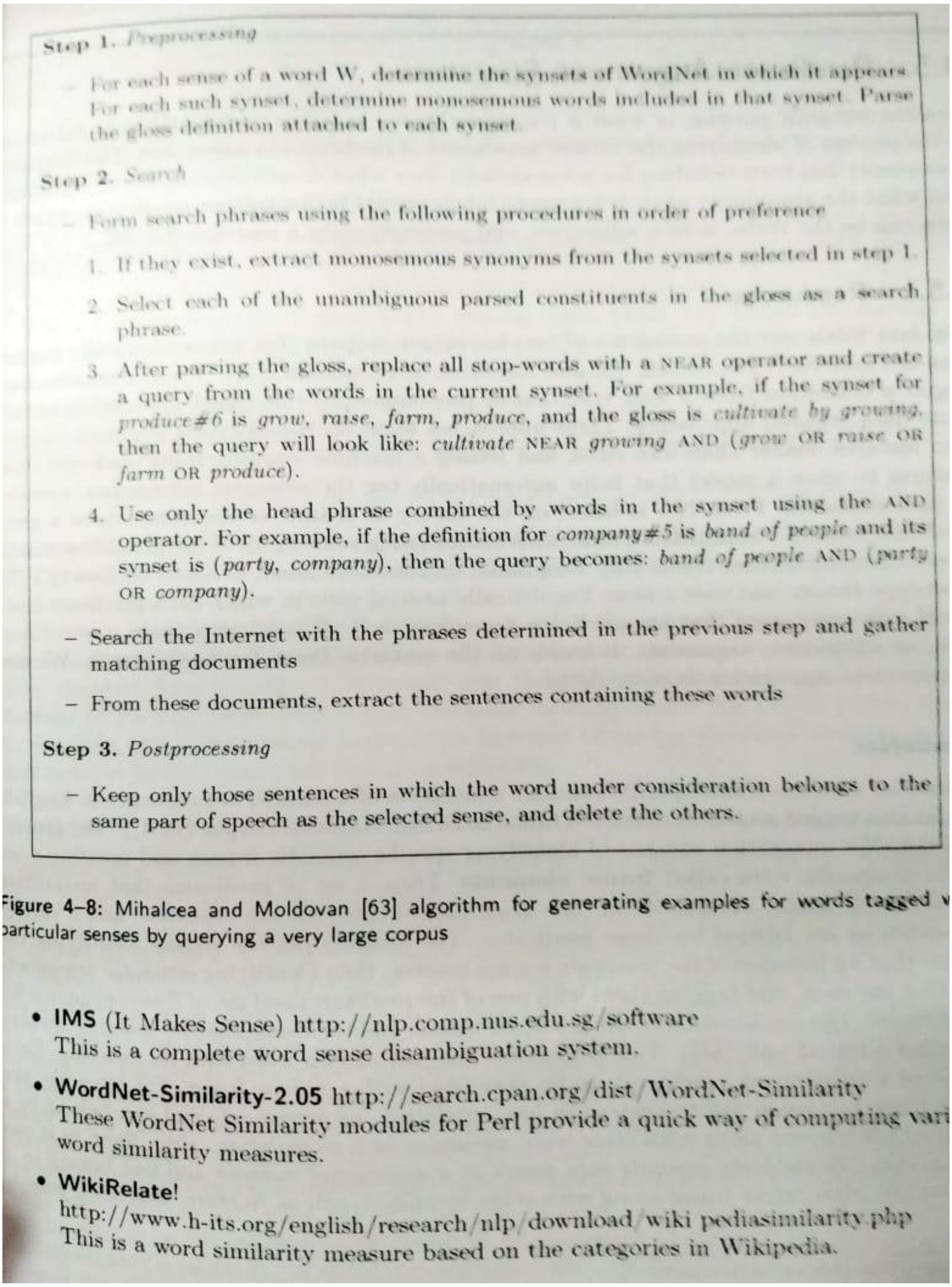
Step 2: Compute the conceptual density of resultant concepts (sub-hierarchies).

Step 3: The concept with the highest CD is selected.

Step 4: Select the senses below the selected concept as the correct sense for the respective words.

Software:

Several software programs are made available by the research community for word sense disambuguity. Few are listed below



Step 1. *Preprocessing*

- For each sense of a word W, determine the synsets of WordNet in which it appears. For each such synset, determine monosemous words included in that synset. Parse the gloss definition attached to each synset.

Step 2. *Search*

- Form search phrases using the following procedures in order of preference
  1. If they exist, extract monosemous synonyms from the synsets selected in step 1.
  2. Select each of the unambiguous parsed constituents in the gloss as a search phrase.
  3. After parsing the gloss, replace all stop-words with a NEAR operator and create a query from the words in the current synset. For example, if the synset for *produce#6* is *grow, raise, farm, produce*, and the gloss is *cultivate by growing*, then the query will look like: *cultivate* NEAR *growing* AND (*grow* OR *raise* OR *farm* OR *produce*).
  4. Use only the head phrase combined by words in the synset using the AND operator. For example, if the definition for *company#5* is *band of people* and its synset is (*party*, *company*), then the query becomes: *band of people* AND (*party* OR *company*).
- Search the Internet with the phrases determined in the previous step and gather matching documents
- From these documents, extract the sentences containing these words

Step 3. *Postprocessing*

- Keep only those sentences in which the word under consideration belongs to the same part of speech as the selected sense, and delete the others.

Figure 4–8: Mihalcea and Moldovan [63] algorithm for generating examples for words tagged w
particular senses by querying a very large corpus

- **IMS** (It Makes Sense) http://nlp.comp.nus.edu.sg/software
  This is a complete word sense disambiguation system.
- **WordNet-Similarity-2.05** http://search.cpan.org/dist/WordNet-Similarity
  These WordNet Similarity modules for Perl provide a quick way of computing vari
  word similarity measures.
- **WikiRelate!**
  http://www.h-its.org/english/research/nlp/download/wikipediasimilarity.php
  This is a word similarity measure based on the categories in Wikipedia.

Above table listed are the source and software's to analysis the word sensing to the best

Predicate Argument Structure:

A thing that refers to the type of event or state we are dealing with is termed a predicate, while the things that refer to the participants in the event/state are called the arguments of the predicate.

Predicates can be divided into two main categories: action and state of being. Predicates that describe an action can be simple, compound, or complete. A simple predicate is a verb or verb phrase without any modifiers or objects

Predicate argument structure is based on the function features of lexical items (most often verbs). The function features determine the thematic roles to be played by the other words in the sentence. However, function features and thematic roles don't always coincide

Shallow semantics parsing or semantic role labelling, is the process of identifying the variousarguments of predicates in a sentence.
In linguistics, *predicate* refers to the main verb in the sentence. Predicate takes arguments.
The role of **Semantic Role Labelling (SRL)** is to determine how these arguments are semantically related to the predicate.

Examples

- The sun (subject) / was shining brightly (predicate).
- The dogs (subject) / were barking loudly (predicate).
- The pretty girl (subject) / was wearing a blue frock (predicate).
- My younger brother (subject) / serves in the army (predicate).
- The man and his wife (subject) / were working in their garden (predicate).

Generally, this process can be defined as the identification of who did what to whom, where, why and how. This is shown with help of diagram

Bell Atlantic Corp. *said* it will *acquire* one of Control Data Corp.'s computer maintenance businesses.

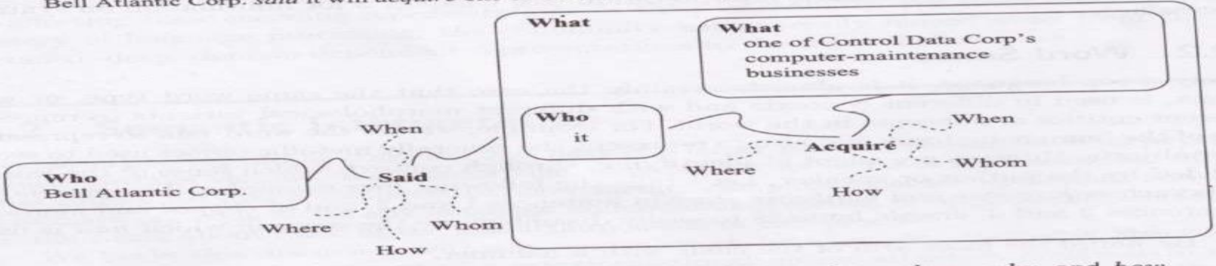
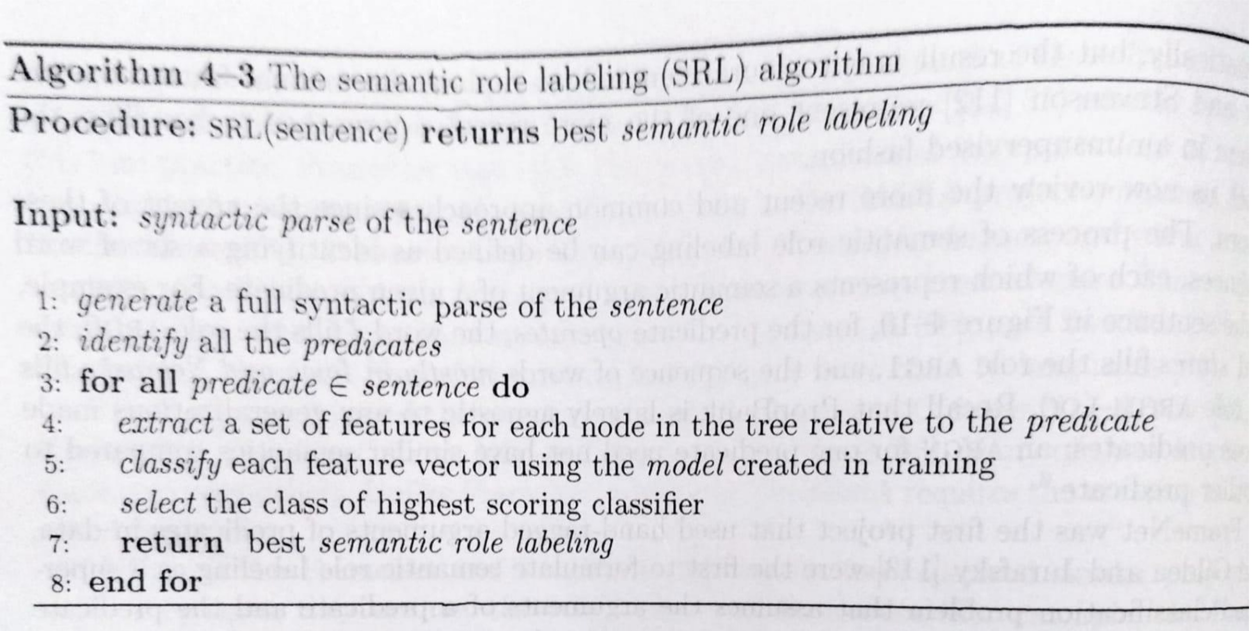


Figure 4–1: A representation of *who* did *what* to *whom*, *when*, *where*, *why*, and *how*

These two grammar structures are used to identify semantic role Labeling (subject and predicate):
Phrase structure grammar, also known as constituency grammar, is a way of representing the syntactic structure of natural language sentences using hierarchical trees(refer to unit 2)
In natural language processing (NLP), phrase structure grammar can be used to analyze, parse, and generate natural language texts and semantic role labeling uses this structure

Combinatory categorical Grammer:
The term "categorical grammar" refers to a variety of approaches to syntax and semantics in which the best word is being sensed.



Algorithm 4–3 The semantic role labeling (SRL) algorithm

Procedure: SRL(sentence) **returns** best *semantic role labeling*

Input: *syntactic parse* of the *sentence*

```
generate a full syntactic parse of the sentence
identify all the predicates
for all predicate ∈ sentence do
    extract a set of features for each node in the tree relative to the predicate
    classify each feature vector using the model created in training
    select the class of highest scoring classifier
    return best semantic role labeling
end for
```

The above method is used to predict the best semantic word labeling and determines subject and predicate of the sentence.

There are two subsets of predicate argument structure
1.frame net 2.propbank(propostional bank)

FrameNet : is a linguistic knowledge graph containing information about lexical and predicate argument semantics of the English language. FrameNet contains two distinct entity classes: frames and lexical units, where a frame is a meaning and a lexical unit is a single meaning for a word.

FrameNet is based on the theory of frame semantics, where a given predicate invokes a semantic frame, this instantiating some or all of the possible semantic roles belonging to that frame. FrameNet contains frame-specific semantic annotation of a number of predicates in English. It contains tagged sentences extracted from British National Corpus (BNC: Search the **British National Corpus** online. Various online services offer the possibility to search and explore the **BNC** via different interfaces which is a trusted search platform).

The process of FrameNet annotation consists of identifying specific semantic frames and creating a set of frame-specific roles called frame elements.

Then, a set of predicates that instantiate the semantic frame, irrespective of their grammatical category, are identified, and a variety of sentences are labelled for those predicates.

The labelling process entails identifying the frame that an instance of the predicate lemma invokes, then identifying semantic arguments for that instance, and tagging them with one of the predetermined ser of frame elements for that frame.

The combination of the predicate lemma and the frame that its instance invokes is called a lexical unit (LU).
This is therefore the pairing of a word with its meaning.

Working of Frames(meaning and Representation)

A Frame is a script-like conceptual structure that describes a particular type of situation, object, or event along with the participants and props that are needed for that Frame. For example, the “Apply_heat” frame describes a common situation involving a Cook, some Food, and a Heating_Instrument, and is evoked by words such as bake, blanch, boil, broil, brown, simmer, steam, etc.

We call the roles of a Frame “frame elements” (FEs) and the frame-evoking words are called “lexical units” (LUs).

FrameNet includes relations between Frames. Several types of relations are defined, of which the most important are:

- Inheritance: An IS-A relation. The child frame is a subtype of the parent frame, and each FE in the parent is bound to a corresponding FE in the child. An example is the “Revenge” frame which inherits from the “Rewards_and_punishments” frame.
- Using: The child frame presupposes the parent frame as background, e.g the “Speed” frame “uses” (or presupposes) the “Motion” frame; however, not all parent FEs need to be bound to child FEs.
- Subframe: The child frame is a subevent of a complex event represented by the parent, e.g. the “Criminal_process” frame has subframes of “Arrest”, “Arraignment”, “Trial”, and “Sentencing”.
- Perspective_on: The child frame provides a particular perspective on an un-perspectivized parent frame. A pair of examples consists of the “Hiring” and “Get_a_job” frames, which perspectivize the “Employment_start” frame from the Employer’s and the Employee’s point of view, respectively.

- Each LU is linked to a Frame, and hence to the other words which evoke that Frame. This makes the FrameNet database similar to a thesaurus, grouping together semantically similar words.

Resources of framenet

The Berkeley FrameNet project is creating an on-line lexical resource for English, based on frame semantics and supported by corpus evidence. The aim is to document the range of semantic and syntactic combinatory possibilities (valences) of each word in each of its senses, through computer-assisted annotation of example sentences and automatic tabulation and display of the annotation results. The major product of this work, the FrameNet lexical database, currently contains more than 10,000 lexical units (defined below), more than 6,100 of which are fully annotated, in more than 825 semantic frames, exemplified in more than 135,000 annotated sentences. It has gone through three releases, and is now in use by hundreds of researchers, teachers, and students around the world (see FrameNet Users). Active research projects are now seeking to produce comparable frame-semantic lexicons for other languages and to devise means of automatically labeling running text with semantic frame information.

2.Propbank:(Propostional bank)

PropBank was developed with the idea of serving as training data for machine learning-based semantic

role labeling systems in mind.

Propbank (proposition bank) is a digital collection of parsed sentences – a treebank – based on the Penn TreeBank, with other treebanks for languages other than English. The sentences are parsed and annotated with the semantic roles described in Verbnet, another major resource. Each sentence in Propbank is linked to a list of its numbered arguments, each with a semantic (thematic) role and selection restrictions, and labeled with its Verbnet class. Verb adjuncts such as adverbials are also labeled with roles, such as temporal and manner. Propbank was made primarily for the training of automatic semantic role labelers through machine learning.

Representation of Propbank

Each propbank instance defines the following member variables:

- Inflection information: *inflection*
- Roleset identifier: *roleset*
- Verb location: *predicate*
- Argument locations and types: *arguments*

It is represented by argument structure for sentence in which which relates to specific frame and connects to the verb

Example : frame :Commerce Propstionalbank orPropbank can connect to the following verbs in different scenarios

Arg0:He can buy in the goods in ecommerce

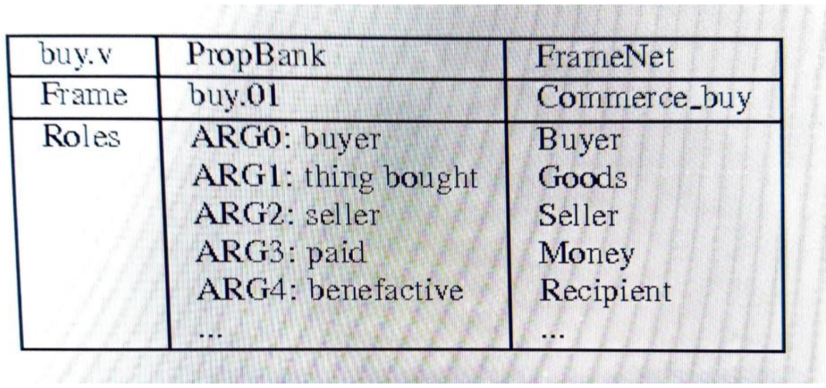
Arg1:He bought things

Arg2:He can sell goods

Arg3:He can make payment of money

Arg 4He can be benefective from ecommerce source

Probable verb to which it can cannot and concept thesaurus of that frame is generated connected to different argument structures



buy.v	PropBank	FrameNet
Frame	buy.01	Commerce_buy
Roles	ARG0: buyer ARG1: thing bought ARG2: seller ARG3: paid ARG4: benefactive ...	Buyer Goods Seller Money Recipient ...

Many of the framset for a particular concept is identified and associated verb referring to propbank is retrived according to the argument structure it will hit on to the proper treebank and information is retrived.

Software used which support NlP which are rule based supports framnet and propbank and are available for download

1.WASP: (welnberg AI autonomous system for software program)

https://wasp-sweden.org/ai-graduate-school-courses/

2.krisper: https://www.cs.utexas.edu/~ml/krisp/

3.chill:[http://www.cs.utexas.edu/ml/hill

Vyasapuri, Bandlaguda, Post:Keshavgiri
Hyderabad-500005,Telangana, India
Tel:040-29880079, 86,8978380692, 9642703342
9652216001, 9550544411, Website:www.mist.ac.in
Email:principal@mist.ac.in
principal.mahaveer@gmail.com
Counseling code:MHVR, University Code:E3

MAHAVEER
INSTITUTE OF SCIENCE & TECHNOLOGY
(AN UGC AUTONOMOUS INSTITUTION)
Approved by AICTE, Affiliated to JNTU,Hyderabad
Accredited by NAAC with 'A' Grade
Recognized Under 2(f) of UGC Act 1956,ISO 9001:2015 Certified

ESTD : 2001

Department of Computer Science and Engineering (AIML)

(R18)

Natural Language Processing

Lecture Notes

B. Tech III YEAR – II SEM

Prepared by

Mrs.Swapna
(Professor&HOD-CSM)
Dept. CSE(AIML)

Syllabus

NATURAL LANGUAGE PROCESSING

B.Tech. III Year II Sem.

L	T	P	C
3	1	0	4

Prerequisites: Data structures, finite automata and probability theory

Course Objectives:

- Introduce to some of the problems and solutions of NLP and their relation to linguistics and statistics.

Course Outcomes:

- Show sensitivity to linguistic phenomena and an ability to model them with formal grammars.
- Understand and carry out proper experimental methodology for training and evaluating empirical NLP systems
- Able to manipulate probabilities, construct statistical models over strings and trees, and estimate parameters using supervised and unsupervised training methods.
- Able to design, implement, and analyze NLP algorithms
- Able to design different language modeling Techniques.

UNIT - I

Finding the Structure of Words: Words and Their Components, Issues and Challenges, Morphological Models

Finding the Structure of Documents: Introduction, Methods, Complexity of the Approaches, Performances of the Approaches

UNIT - II

Syntax Analysis: Parsing Natural Language, Treebanks: A Data-Driven Approach to Syntax, Representation of Syntactic Structure, Parsing Algorithms, Models for Ambiguity Resolution in Parsing, Multilingual Issues

UNIT - III

Semantic Parsing: Introduction, Semantic Interpretation, System Paradigms, Word Sense Systems, Software.

UNIT - IV

Predicate-Argument Structure, Meaning Representation Systems, Software.

UNIT - V

Discourse Processing: Cohension, Reference Resolution, Discourse Cohension and Structure **Language Modeling:** Introduction, N-Gram Models, Language Model Evaluation, Parameter Estimation, Language Model Adaptation, Types of Language Models, Language-Specific Modeling Problems, Multilingual and Crosslingual Language Modeling

TEXT BOOKS:

1. Multilingual natural Language Processing Applications: From Theory to Practice – Daniel M. Bikel and Imed Zitouni, Pearson Publication

Natural Language Processing and Information Retrieval: Tanvier Siddiqui, U.S. Tiwary

Unit 5

UNIT 5

Discourse Processing:

Cohension, Reference Resolution, Discourse Cohension and Structure

Language Modeling:

Introduction, N-Gram Models, Language Model Evaluation, Parameter Estimation, Language Model Adaptation, Types of Language Models, Language-Specific Modeling Problems, Multilingual and Crosslingual Language Modeling

Cohesion is a term in linguistics that refers to how the structure and content of a sentence or text is linked together to create meaning.

Cohesion needs to be achieved in a sentence, within a paragraph and across paragraphs for a text to make sense.

Why is cohesion important?

Cohesion means that writing is well structured with linked ideas that follow a logical pattern. Sentences and paragraphs flow smoothly and are written in the same tense, meaning the piece of writing as a whole is fluid and makes sense.

Cohesion is important as:

- It teaches how to order and structure sentences and paragraphs
- It means related ideas are kept together and flow logically from one to another
- It helps to express their ideas in a way that the reader will clearly understand

How can cohesion be achieved?

To achieve **cohesion**, we must be able to select appropriate nouns, verbs, adjectives and adverbs in order for a sentence to make sense.

They must then write appropriate sentences, which organise their ideas and follow a logical sequence.

There are 4 main types of sentences that can choose from:

- **Statements -** convey information
- **Questions -** ask something and usually end with a question mark
- **Commands -** give instructions or tell you to do something

- **Exclamations** - usually begin with 'What' or 'How,' include a verb and can end with an exclamation mark

For more help, try these Types of Sentences Display Posters.

What are cohesive devices?

Cohesive devices are techniques that can be used to help create **cohesion**. Some examples of these are:

Adverbials
Expressing place, time or manner, adverbials can help create **cohesion** in or across paragraphs. Take a look at this example:

Without adverbials:

"Tim left home. He realised he had forgotten his homework."

With adverbials:

"Tim left home early in the morning. Later that day, he realised he had forgotten his homework."

The adverbial of time keeps the sentences in a logical order and makes it easier for the reader to follow the text.

Ellipsis
Ellipsis is another cohesive device that can improve the flow of a sentence. Ellipsis refers to the removal of superfluous words, as the meaning can be inferred from the preceding clause. For example:

Without ellipsis:

"If James listens to music, he will have to dance to the music."

With ellipsis:

"If James listens to music, he will have to dance."

In this case, the sentence with ellipsis avoids repeating words, but still makes sense.

Repetition
Repetition can also be used as a cohesive device to highlight or emphasise important information. For example:

"Peter was running late. He had promised not to be late this time. If only he could get to places on time!"

In this example, the repetition of the adjective/adverb 'late' helps create a sense of urgency across the sentences.

Pronouns
Using pronouns in a sentence allows you to refer back to a noun without having to repeat it. For example:

Without pronoun:

“Billy listened to music while sitting in Billy’s car.”

With pronoun:

“Billy listened to music while sitting in his car.”

In this example, using a pronoun helps the sentence flow more smoothly, while still making sense.

This is also known as an anaphoric reference. A cataphoric reference is the opposite of this, referring to something later in a text. For example:

“Jess couldn’t wait to see him, but Jack would not be back until next week.”

In this case, the pronoun 'him' refers to Jack.

Remember...

It is also important to use the same tense throughout a text in order for it to achieve **cohesion**.

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. If we talk about 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.

Concept of Coherence

Coherence and discourse structure are interconnected in many ways. Coherence, along with property of good text, is used to evaluate the output quality of natural language generation system. The question that arises here is what does it mean for a text to be coherent? Suppose we collected one sentence from every page of the newspaper, then will it be a discourse? Of-course, not. It is because these sentences do not exhibit coherence. The coherent discourse must possess the following properties −

Coherence relation between utterances

The discourse would be coherent if it has meaningful connections between its utterances. This property is called coherence relation. For example, some sort of explanation must be there to justify the connection between utterances.

Relationship between entities

Another property that makes a discourse coherent is that there must be a certain kind of relationship with the entities. Such kind of coherence is called entity-based coherence.

Discourse structure

An important question regarding discourse is what kind of structure the discourse must have. The answer to this question depends upon the segmentation we applied on discourse. Discourse segmentations may be defined as determining the types of structures for large discourse. It is quite difficult to implement discourse segmentation, but it is very important for **information retrieval, text summarization and information extraction** kind of applications.

Algorithms for Discourse Segmentation

In this section, we will learn about the algorithms for discourse segmentation. The algorithms are described below −

Unsupervised Discourse Segmentation

The class of unsupervised discourse segmentation is often represented as linear segmentation. We can understand the task of linear segmentation with the help of an example. In the example, there is a task of segmenting the text into multi-paragraph units; the units represent the passage of the original text. These algorithms are dependent on cohesion that may be defined as the use of certain linguistic devices to tie the textual units together. On the other hand, lexicon cohesion is the cohesion that is indicated by the relationship between two or more words in two units like the use of synonyms.

Supervised Discourse Segmentation

The earlier method does not have any hand-labeled segment boundaries. On the other hand, supervised discourse segmentation needs to have boundary-labeled training data. It is very easy to acquire the same. In supervised discourse segmentation, discourse marker or cue words play an important role. Discourse marker or cue word is a word or phrase that functions to signal discourse structure. These discourse markers are domain-specific.

Text Coherence

Lexical repetition is a way to find the structure in a discourse, but it does not satisfy the requirement of being coherent discourse. To achieve the coherent discourse, we must focus on coherence relations in specific. As we know that coherence relation defines the possible connection between utterances in a discourse. Hebb has proposed such kind of relations as follows −

We are taking two terms $\mathbf{S_0}$ and $\mathbf{S_1}$ to represent the meaning of the two related sentences −

Result

It infers that the state asserted by term $\mathbf{S_0}$ could cause the state asserted by $\mathbf{S_1}$. For example, two statements show the relationship result: Ram was caught in the fire. His skin burned.

Explanation

It infers that the state asserted by $\mathbf{S_1}$ could cause the state asserted by $\mathbf{S_0}$. For example, two statements show the relationship − Ram fought with Shyam’s friend. He was drunk.

Parallel

It infers p(a1,a2,…) from assertion of $\mathbf{S_0}$ and p(b1,b2,…) from assertion $\mathbf{S_1}$. Here ai and bi are similar for all i. For example, two statements are parallel − Ram wanted car. Shyam wanted money.

Elaboration

It infers the same proposition P from both the assertions − $\mathbf{S_0}$ and $\mathbf{S_1}$ For example, two statements show the relation elaboration: Ram was from Chandigarh. Shyam was from Kerala.

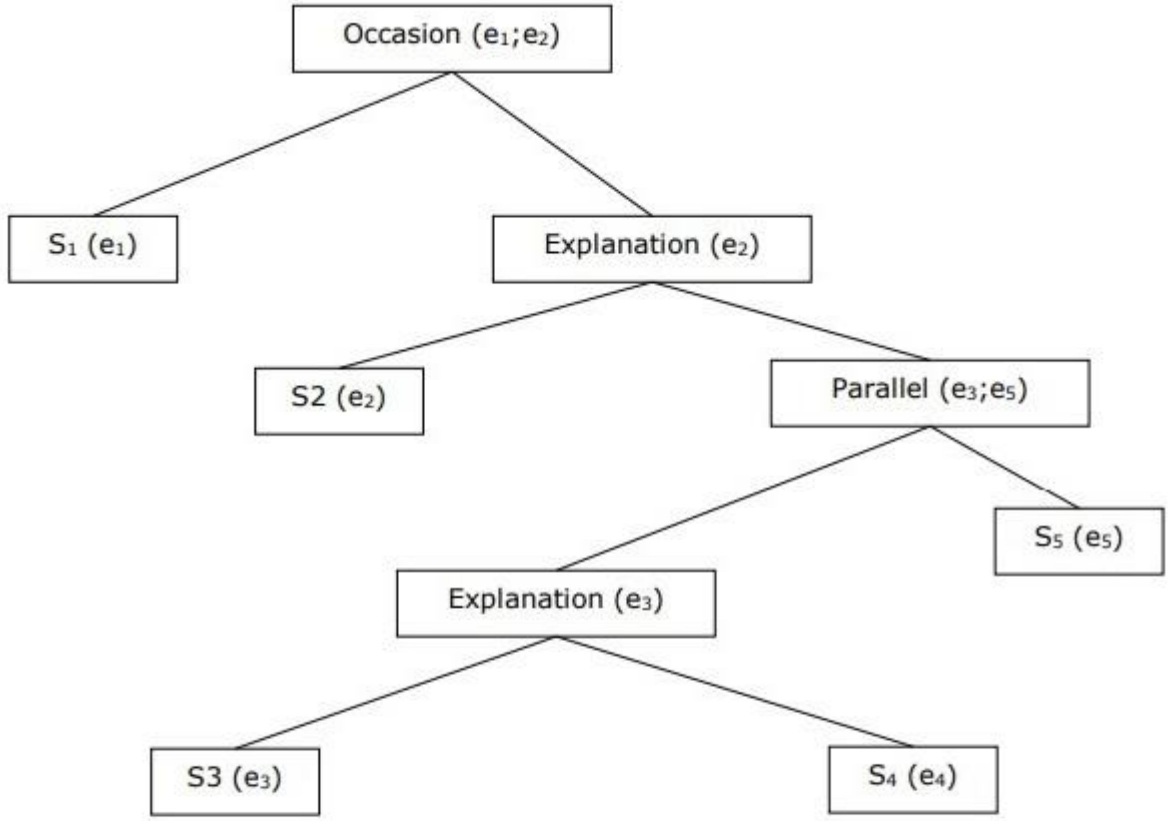
Occasion

It happens when a change of state can be inferred from the assertion of $\mathbf{S_0}$, final state of which can be inferred from **S_1** and vice-versa. For example, the two statements show the relation occasion: Ram picked up the book. He gave it to Shyam.

Building Hierarchical Discourse Structure

The coherence of entire discourse can also be considered by hierarchical structure between coherence relations. For example, the following passage can be represented as hierarchical structure −

- **S_1** − Ram went to the bank to deposit money.
- **S_2** − He then took a train to Shyam's cloth shop.
- **S_3** − He wanted to buy some clothes.
- **S_4** − He do not have new clothes for party.
- **S_5** − He also wanted to talk to Shyam regarding his health



Reference Resolution

Interpretation of the sentences from any discourse is another important task and to achieve this we need to know who or what entity is being talked about. Here, interpretation reference is the key element. **Reference** may be defined as the linguistic expression to denote an entity or individual. For example, in the passage, Ram, the manager of ABC bank, saw his friend Shyam at a shop. He went to meet him, the linguistic expressions like Ram, His, He are reference.

On the same note, **reference resolution** may be defined as the task of determining what entities are referred to by which linguistic expression.

Terminology Used in Reference Resolution

We use the following terminologies in reference resolution −

- **Referring expression** − The natural language expression that is used to perform reference is called a referring expression. For example, the passage used above is a referring expression.
- **Referent** − It is the entity that is referred. For example, in the last given example Ram is a referent.
- **Corefer** − When two expressions are used to refer to the same entity, they are called corefers. For example, ***Ram*** and ***he*** are corefers.
- **Antecedent** − The term has the license to use another term. For example, ***Ram*** is the antecedent of the reference ***he***.
- **Anaphora & Anaphoric** − It may be defined as the reference to an entity that has been previously introduced into the sentence. And, the referring expression is called anaphoric.
- **Discourse model** − The model that contains the representations of the entities that have been referred to in the discourse and the relationship they are engaged in.

Types of Referring Expressions

Let us now see the different types of referring expressions. The five types of referring expressions are described below −

Indefinite Noun Phrases

Such kind of reference represents the entities that are new to the hearer into the discourse context. For example − in the sentence Ram had gone around one day to bring him some food − some is an indefinite reference.

Definite Noun Phrases

Opposite to above, such kind of reference represents the entities that are not new or identifiable to the hearer into the discourse context. For example, in the sentence - I used to read The Times of India – The Times of India is a definite reference.

Pronouns

It is a form of definite reference. For example, Ram laughed as loud as he could. The word **he** represents pronoun referring expression.

Demonstratives

These demonstrate and behave differently than simple definite pronouns. For example, this and that are demonstrative pronouns.

Names

It is the simplest type of referring expression. It can be the name of a person, organization and location also. For example, in the above examples, Ram is the name-refereeing expression.

Reference Resolution Tasks

The two reference resolution tasks are described below.

Coreference Resolution

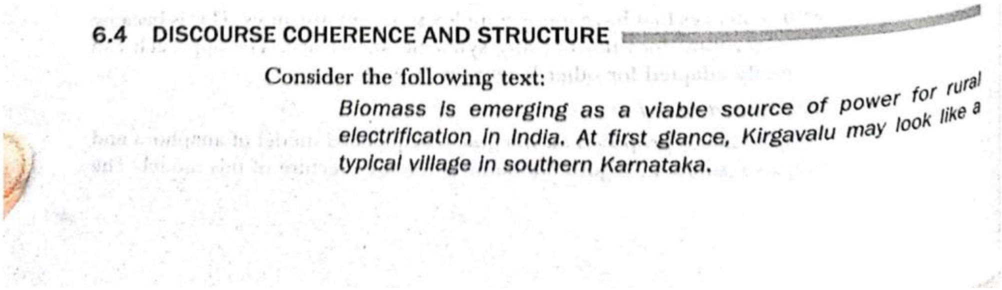
It is the task of finding referring expressions in a text that refer to the same entity. In simple words, it is the task of finding corefer expressions. A set of coreferring expressions are called coreference chain. For example - He, Chief Manager and His - these are referring expressions in the first passage given as example.

Constraint on Coreference Resolution

In English, the main problem for coreference resolution is the pronoun it. The reason behind this is that the pronoun it has many uses. For example, it can refer much like he and she. The pronoun it also refers to the things that do not refer to specific things. For example, It’s raining. It is really good.

Pronominal Anaphora Resolution

Unlike the coreference resolution, pronominal anaphora resolution may be defined as the task of finding the antecedent for a single pronoun. For example, the pronoun is his and the task of pronominal anaphora resolution is to find the word Ram because Ram is the antecedent



6.4 DISCOURSE COHERENCE AND STRUCTURE

Consider the following text:

Biomass is emerging as a viable source of power for rural electrification in India. At first glance, Kirgavalu may look like a typical village in southern Karnataka.

Both sentences in the passage are well formed and independently interpretable. But the passage seems a bit odd. The reason is that we try to establish a connection between the first and the second sentence. We raise questions such as how the Kirgavalu village is connected with biomass. In this case, we find it difficult to understand the connection. By raising such questions, we point out that the discourse is not coherent. In order to make the text coherent, we might build an understanding that perhaps the Kirguvalu village has a biomass plant. The attempt on the part of hearer (reader) to establish a connection between a pair of sentences suggests that merely grouping well-formed, independently interpretable sentences, does not yield meaningful passages; coherence is required to produce a meaningful composition. It is also needed for discourse comprehension. Coherence is different from cohesion. *Cohesion* refers to the grammatical relationship between words, referring forwards or backwards to other words, or substituting words or phrases, within the text.

There are a number of relations that connect utterances (sentences). Consider the following text:

Section 5.2 deals with sentence level meaning representation. (6.18a)

In particular, we discuss the general characteristics of meaning representation languages (Section 5.2.1), (6.18b′) *and computational approaches to semantic analysis (syntax-driven semantic analysis and semantic grammars in Section 5.2.3).* (6.18b″)

Next, we discuss the internal structure of words, their relationships, and their meanings in Section 5.3. (6.18c)

Sentence (6.18a) introduces a topic. The next sentence elaborate on that by breaking it into subtopics, clauses (6.18b′) and (6.18b″). A temporal relation exists between (6.18b) and (6.18c) indicated by 'next', which links the topics we intend to discuss. Figure 6.3 illustrates these relations.

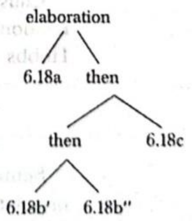
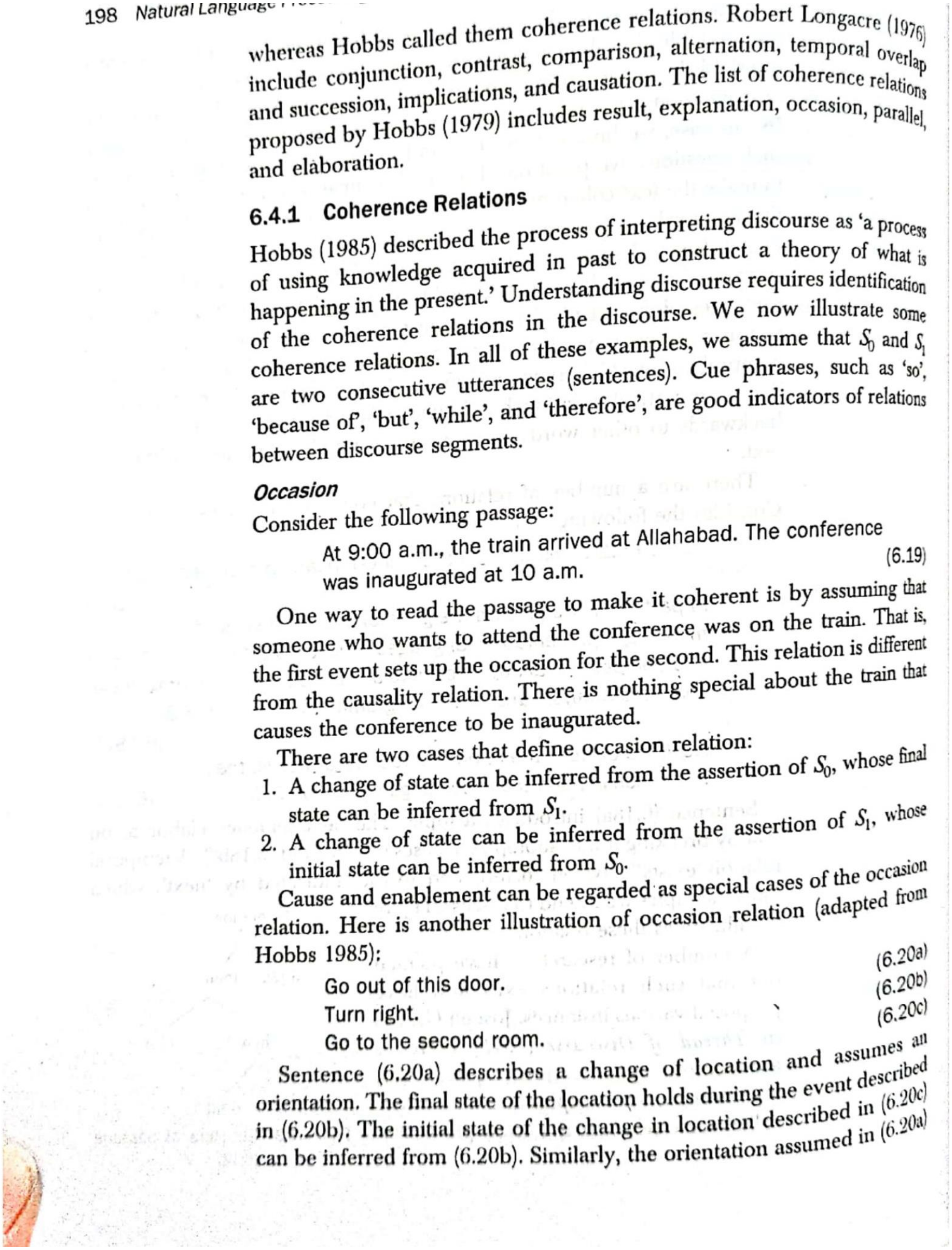


Figure 6.3 Structure of passage (6.18)

A number of researchers have pointed out that such relations exist and have proposed various instances. Joseph Grimes in *Thread of Discourse* (1975) includes alternation, specification, equivalence, attributions, and explanations. Grimes called these relations rhetorical predicates,



whereas Hobbs called them coherence relations. Robert Longacre (1976) include conjunction, contrast, comparison, alternation, temporal overlap and succession, implications, and causation. The list of coherence relations proposed by Hobbs (1979) includes result, explanation, occasion, parallel, and elaboration.

6.4.1 Coherence Relations

Hobbs (1985) described the process of interpreting discourse as 'a process of using knowledge acquired in past to construct a theory of what is happening in the present.' Understanding discourse requires identification of the coherence relations in the discourse. We now illustrate some coherence relations. In all of these examples, we assume that S_0 and S_1 are two consecutive utterances (sentences). Cue phrases, such as 'so', 'because of', 'but', 'while', and 'therefore', are good indicators of relations between discourse segments.

Occasion

Consider the following passage:

At 9:00 a.m., the train arrived at Allahabad. The conference was inaugurated at 10 a.m. (6.19)

One way to read the passage to make it coherent is by assuming that someone who wants to attend the conference was on the train. That is, the first event sets up the occasion for the second. This relation is different from the causality relation. There is nothing special about the train that causes the conference to be inaugurated.

There are two cases that define occasion relation:

1. A change of state can be inferred from the assertion of S_0, whose final state can be inferred from S_1.
2. A change of state can be inferred from the assertion of S_1, whose initial state can be inferred from S_0.

Cause and enablement can be regarded as special cases of the occasion relation. Here is another illustration of occasion relation (adapted from Hobbs 1985):

Go out of this door. (6.20a)
Turn right. (6.20b)
Go to the second room. (6.20c)

Sentence (6.20a) describes a change of location and assumes an orientation. The final state of the location holds during the event described in (6.20b). The initial state of the change in location described in (6.20c) can be inferred from (6.20b). Similarly, the orientation assumed in (6.20a)

is the initial state for the change in state described in 6.20b, and its final state is assumed in 6.20c. Figure 6.4 shows the inferences that need to be drawn to satisfy the definition. It is possible to find more than one relation between a pair of sentences, provided they do not involve inconsistent assumptions.

Type	6.20a	6.20b	6.20c
1	loc 1 → loc 2	loc2	
2		loc 2	loc 2 → loc 3
2	angle 1	angle 1 → angle 2	
1		angle 1 → angle 2	angle 2

Figure 6.4 Occasion relation in sentences (6.20)

Here is another example of occasion relation:

Increment the counter by one.
If it is 100, reset it to zero. (6.21)

Here, the value of the counter is changed, which is presupposed in the second sentence.

Explanation

The segment S_1 is an explanation of S_0 if S_1 describes an event or state that could cause the state or event described in S_0. Explanation is a relation that relates a segment of discourse to the listener's prior knowledge. It is formally defined as follows:

Infer that the state or event asserted by S_1 causes, or could cause, the state or event asserted by S_0.

Suha ate all the rice in bowl. She was very hungry. (6.22)

In this passage, S_1 explains the event asserted by S_0.

Causality may sometimes be explicitly stated as in the following statements:

Suha ate all the rice in the bowl because she was very hungry.
I get late because of the procession on the roads.

Elaboration

The definition of the elaboration relation involves identical entities. It is defined formally as follows:

Infer the same proposition from the assertions of S_0 and S_1.

A simple example of elaboration relation is

Saif scored an unbeaten century today. He was in full
swing and made 108 not out on 87 balls. (6.23)

From the first sentence, and from what we know about an unbeaten century, we infer that Saif made more than 100 runs and he remains not out. By assuming that 'he' refers to Saif, we infer the same proposition from S_1 and thus establish the elaboration relation.

Parallel

The parallel relation is based on the similarity of entities. In this context, we say that two entities are similar if they share some property. More formally, the parallel relation is defined as follows:

Infer $p(x_1, x_2,\ldots)$ from the assertion of S_0 and $p(y_1, y_2, \ldots)$ from the assertion of S_1, where x_1 and y_2 are similar, for all i.

Suha likes reading novels. Zuha enjoys reading science fiction. (6.24)

For each of the segments, we infer that a person likes reading books. Zuha and Suha are similar in that they are both people, reading novels, and reading science fiction. The predicate in this case may be hobby.

Contrast

There are two cases that define contrast relations:

1. Infer $p(x)$ from the assertion of S_0 and $\neg\, p(y)$ from the assertion of S_1, where x and y are similar.
2. Infer $p(x)$ from the assertion of S_0 and $\neg\, p(y)$ from the assertion of S_1, where there is some property q such that $q(x)$ and $\neg\, q(y)$.

Here is a simple example of the first case:

Suha does not like cricket. But she likes cricket more than any other game. (6.25)

Exemplification

Infer $p\ (X)$ from the assertion of S_0 and infer $p(x)$ from the assertion of S_1, where x is a member or subset of X. This relation is illustrated as follows:

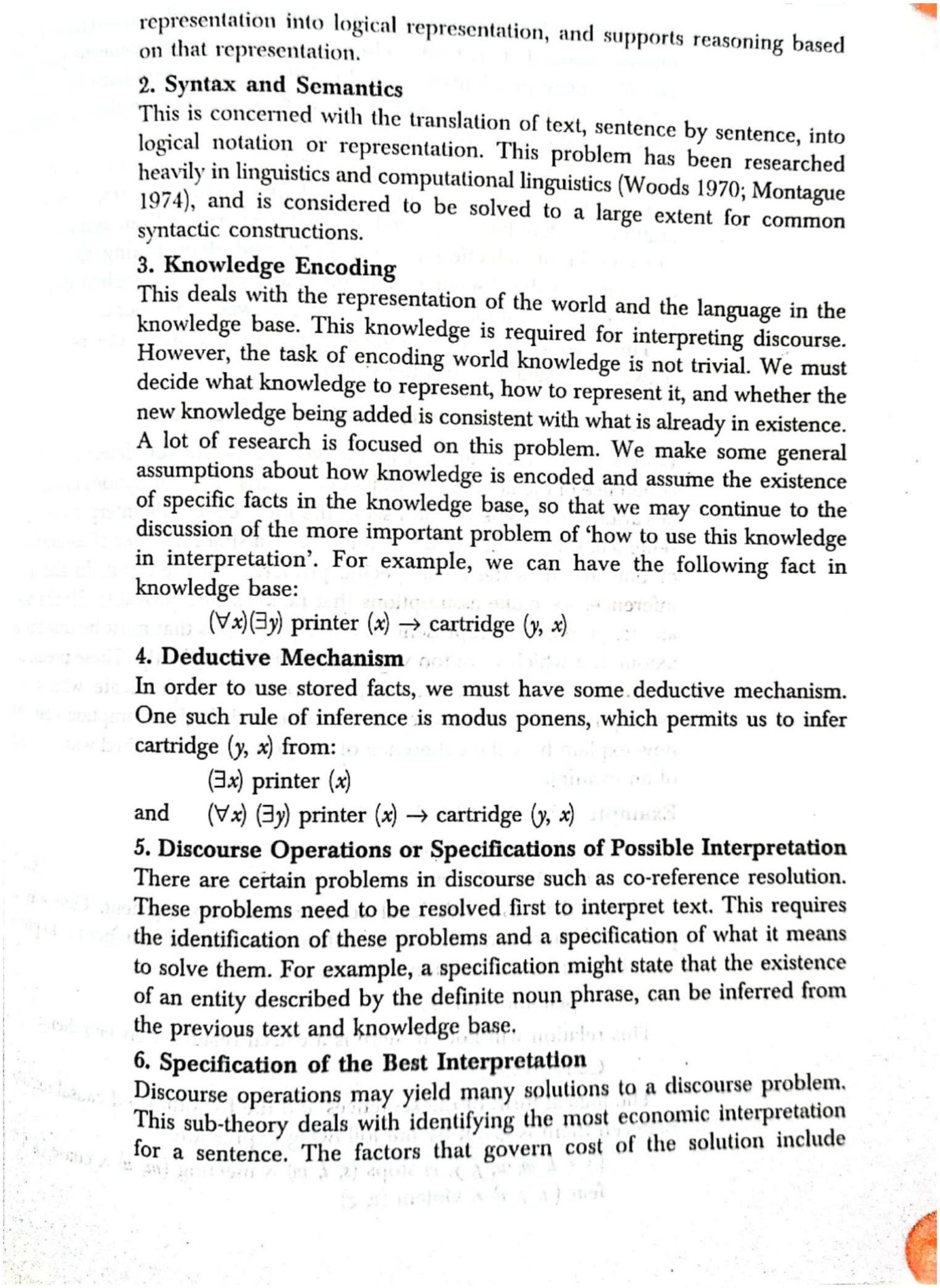
Suha bought a printer today. It is a laser printer. (6.26)

6.4.2 Discourse Interpretation

Hobbs (1985) suggested that the problem of discourse interpretation can be solved by decomposing it into six sub-problems.

1. Logical Notation or Knowledge Representation

The first sub-problem deals with the problem of representation. In order to interpret discourse, a logical representation of natural language sentences is required. First order predicate logic representation is one such representation that has been used to translate natural language



representation into logical representation, and supports reasoning based on that representation.

2. Syntax and Semantics

This is concerned with the translation of text, sentence by sentence, into logical notation or representation. This problem has been researched heavily in linguistics and computational linguistics (Woods 1970; Montague 1974), and is considered to be solved to a large extent for common syntactic constructions.

3. Knowledge Encoding

This deals with the representation of the world and the language in the knowledge base. This knowledge is required for interpreting discourse. However, the task of encoding world knowledge is not trivial. We must decide what knowledge to represent, how to represent it, and whether the new knowledge being added is consistent with what is already in existence. A lot of research is focused on this problem. We make some general assumptions about how knowledge is encoded and assume the existence of specific facts in the knowledge base, so that we may continue to the discussion of the more important problem of 'how to use this knowledge in interpretation'. For example, we can have the following fact in knowledge base:

$$(\forall x)(\exists y)\ \text{printer}\ (x) \rightarrow \text{cartridge}\ (y, x)$$

4. Deductive Mechanism

In order to use stored facts, we must have some deductive mechanism. One such rule of inference is modus ponens, which permits us to infer cartridge (y, x) from:

$$(\exists x)\ \text{printer}\ (x)$$

and $(\forall x)\ (\exists y)\ \text{printer}\ (x) \rightarrow \text{cartridge}\ (y, x)$

5. Discourse Operations or Specifications of Possible Interpretation

There are certain problems in discourse such as co-reference resolution. These problems need to be resolved first to interpret text. This requires the identification of these problems and a specification of what it means to solve them. For example, a specification might state that the existence of an entity described by the definite noun phrase, can be inferred from the previous text and knowledge base.

6. Specification of the Best Interpretation

Discourse operations may yield many solutions to a discourse problem. This sub-theory deals with identifying the most economic interpretation for a sentence. The factors that govern cost of the solution include

complexity of proof, salience of axiom used, and redundancy in the interpretation. Let us take a close look at what the discourse problems are. Discourse problems can be divided into those problems that can be solved using information within the sentence, and those that involve the relation of the sentence to its context.

The within-sentence problems include problems of co-reference resolution, e.g., resolving pronouns, definite noun phrases, and missing arguments; identifying intended predicates where predicate is non-specific; satisfaction of selectional constraints; and determining the internal coherence of the discourse. The internal coherence problem deals with inferring relationship, such as causality, between sentences.

The second type of discourse problem considers the relationship between the sentence and the world.

6.4.3 Abductive Interpretation of Local Coherence

Hobbs (1985) presented an abductive framework for determining local coherence of the utterance. 'Abductive' means that assumptions are allowed at various costs. The method seeks the most economic interpretation of a sentence, such as an explanation that uses a small number of assumptions or one that uses the most specific properties of the input. In abductive inference, we make assumptions that need not be provable. Hobbs used an etc. predicate to represent all other properties that must be true for an axiom, but which were too vague to be stated explicitly. These predicates are assumed at a certain cost, not proved. A predicate with a low assumption cost will be preferred to one with high assumption cost. We now explain how the coherence of a segment is established with the help of an example.

Example 6.3 Consider the following text:

The local administration stopped the trade union from meeting. They feared violence. (6.27)

We need to establish local coherence in this segment. One way to prove that there is a coherence relation between the sentences is to prove that there is an explanation relation, i.e.,

Explanation $(e1, e2)$

This relation will hold if there is a causal relation between them:

Cause $(e2, e1)$

The logical form of the sentences and the hypothesized causal relation between them is given by the following expression.

$(\exists\ s, l, m, u, f, y, v)$ stops $(s, l, m) \wedge$ meeting $(m, u) \wedge$ cuase $(f, s) \wedge$ fear $(f, y, v) \wedge$ violent (v, z)

To prove this expression, we require axioms representing world knowledge in addition to axioms about coherence relation. The world knowledge needed for establishing coherence in this example is as follows:

1. If there is a fear event *f* imposed by someone, say *y*, of violence *v*, it means that *y* does not want violence (*v*).
2. A meeting *m*, by trade union *u*, causes violence.
3. If someone *y*, does not want (diswant) event *v*, and *v* is caused by *m*, then that will also cause *y* to diswant *m*.
4. If the local administration does not want something, then they will stop it.
5. And finally, cause is transitive, i.e., if *e*1 causes *e*2 and *e*2 causes *e*3, then *e*1 causes *e*3.

The axioms representing these sentences are as follows:

$(\forall f, y, v)$ fear $(f, y, v) \rightarrow \exists\ d)$ diswant $(d, y, v) \wedge$ cause (f, d)

$(\forall m, u)$ meeting $(m, u) \rightarrow (\exists\ v, z)$ cause $(m, v) \wedge$ violent (v, z)

$(\forall m, v, d, y)$ cause $(m, v) \wedge$ diswant $(d, y, v) \rightarrow (\exists\ d1)$ diswant $(d1, y, m) \wedge$ cause $(d, d1)$

$(\forall d1, l, m)$ diswant $(d1, l, m) \wedge$ localadministration $(l) \rightarrow (\exists s)$ stop $(s, l, m) \wedge$ cause $(d1, s)$

$(\forall e1, e2, e3)$ cause $(e1, e2) \wedge$ cause $(e2, e3) \rightarrow$ cause $(e1, e3)$

The derivation is shown in Figure 6.5. During the derivation, we also unify *y* with *l*.

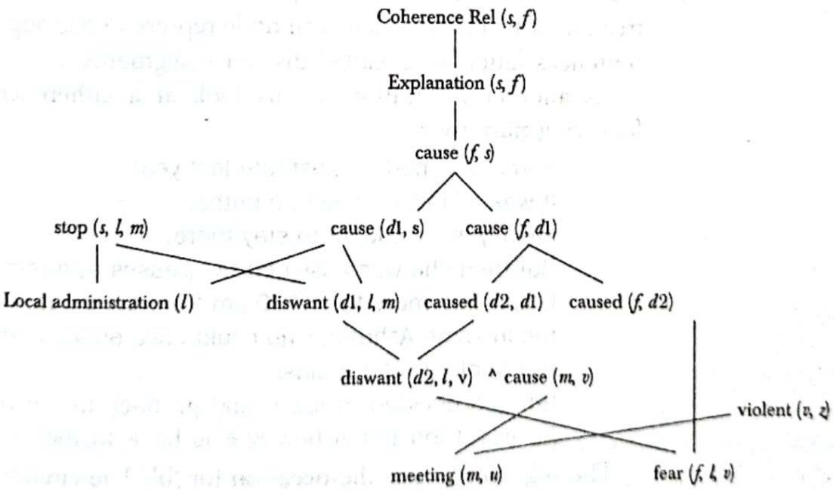


Figure 6.5 Interpretation of sentences (6.27)

Sometimes, the speaker uses a connective, which makes coherent relations explicit. For example, the use of 'because' to join the two sentences in segment (6.27), resulting in *The local administration stopped trade union from meeting because they feared violence*, would make the coherence relation explicit. The literal cause (*f, s*) becomes part of the logical form of the sentence and we need not to assume it.

We can extend this framework to establish the coherence of larger discourse.

6.4.4 Discourse Structure

So far we have discussed only the relations between a pair of sentences. In fact, it is possible to establish such relations in longer discourse. A discourse has a structure. For example, sentences (6.18b′) and (6.18b″) are related by an occasion relation. They combine to give a segment, which is linked with (6.18c) by an occasion relation. The resulting composed segment is related to (6.18a) by an elaboration relation. The coherence relationships between these sentences assign a 'coherence structure' to the discourse, as shown in Figure 6.6. It is a tree-like structure in which each node represents a group of locally coherent sentences (utterances) called discourse segments.

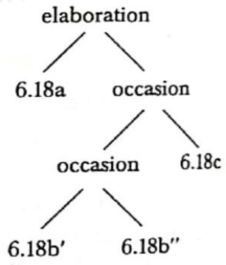


Figure 6.6 Coherence structure of text (6.18)

As another illustration, let us look at a coherence structure of the following narrative:

Sumitha joined the institute last year. (6.28a)
It was all okay for eight months, (6.28b)
Initially, she thought to stay there. (6.28c)
But then she was assigned UG classes at a remote centre.(6.28d)
Um, it was more than 100 km from the institute and that
too in some Ashram, who could have enough time to waste
the whole day for a class. (6.28e)
So, she decided to leave and go back to her parent (6.28f)
institute|and that's how she is back to the university. (6.28g)

The segment 'a' sets the occasion for 'b'. The circumstance of segment 'd'–'e' causes and thus occasions the events of 'f'. The segments 'c' and 'f' are contrasting relations. 'a'–'e' and 'f' are related by a set of events and its outcome. Figure 6.7 illustrates the structure.

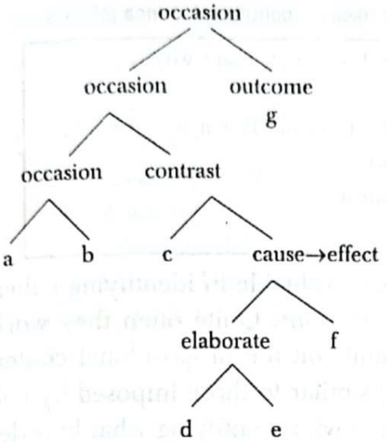


Figure 6.7 Structure of text (6.28)

Such a structure of discourse is useful in explaining classical problems of 'topic', 'genre', and 'coherence drift' that occur in ordinary conversation.

Hobbs proposed a four-step procedure for analysing discourse. As we move from one step to another, difficulty increases. The steps involved in the procedure are discussed below.

1. Identify one or two major breaks in the text and divide the text into two or three segments. This division corresponds to most natural division that one can carry out intuitively. This process is repeated for each segments obtained at first place. The iteration continues until we get single clause. The output of this step is a tree structure of the text. For example, in text (6.18), the major break comes between sentences (6.18a) and (6.18c). Within sentence (6.19b), there is a break between the first clause and the second clause of the sentence. This gives the tree structure of Figure 6.5.
2. In step 2, the non-terminal nodes of the tree are labelled with coherence relations. We follow a bottom-up approach to get an understanding of what is being representing by the composed segment. Thus, in Figure 6.5, the node linking (6.18b′) and (6.18b″) is labelled with the occasion relation. The node linking the resulting segment and (6.18c) is labelled with occasion relation and son. This step requires an understanding of different types of relations. Some simple heuristics based on what conjunctions need to be inserted might help identifying coherence relation. For example, if we can insert 'then' between S0 and S1 and reversing the order of the segment changes the sense, then the occasion relation is quite likely. If 'because' seems to be appropriate between S0 and S1, then explanation relation is preferred candidate. Table 6.3 lists useful conjuncts to identify coherence relation.

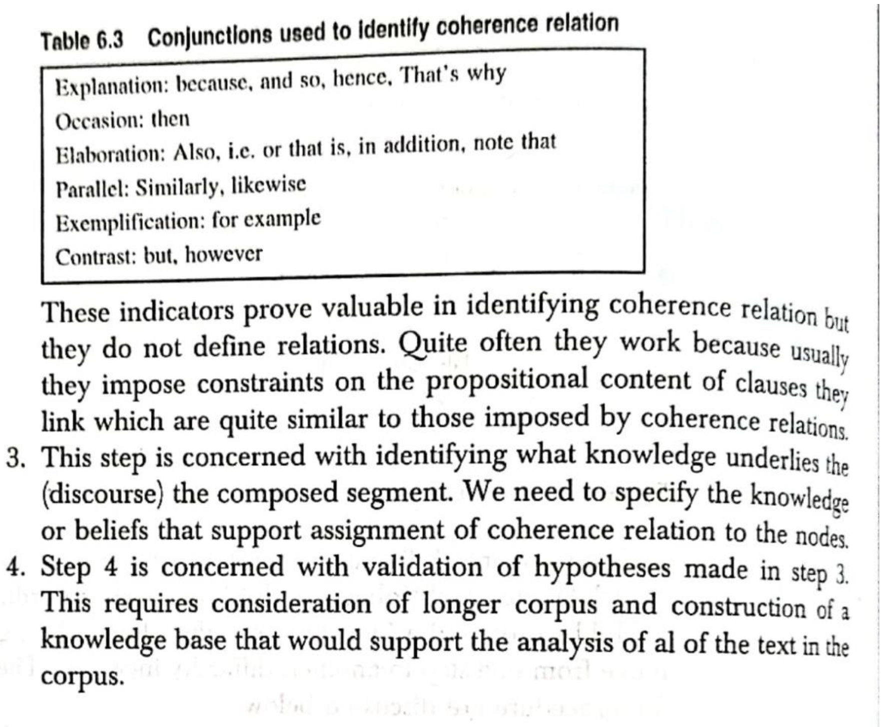


Table 6.3 Conjunctions used to identify coherence relation

Explanation:	because, and so, hence, That's why
Occasion:	then
Elaboration:	Also, i.e. or that is, in addition, note that
Parallel:	Similarly, likewise
Exemplification:	for example
Contrast:	but, however

These indicators prove valuable in identifying coherence relation but they do not define relations. Quite often they work because usually they impose constraints on the propositional content of clauses they link which are quite similar to those imposed by coherence relations.

3. This step is concerned with identifying what knowledge underlies the (discourse) the composed segment. We need to specify the knowledge or beliefs that support assignment of coherence relation to the nodes.
4. Step 4 is concerned with validation of hypotheses made in step 3. This requires consideration of longer corpus and construction of a knowledge base that would support the analysis of al of the text in the corpus.

Language modeling is the way of determining the probability of any sequence of words. Language modeling is used in a wide variety of applications such as Speech Recognition, Spam filtering, etc. In fact, language modeling is the key aim behind the implementation of many state-of-the-art Natural Language Processing models.

Methods of Language Modelings:

Two types of Language Modelings:

- **Statistical Language Modelings**: Statistical Language Modeling, or Language Modeling, is the development of probabilistic models that are able to predict the next word in the sequence given the words that precede. Examples such as N-gram language modeling.
- **Neural Language Modelings**: Neural network methods are achieving better results than classical methods both on standalone language models and when models are incorporated into larger models on challenging tasks like speech recognition and machine translation. A way of performing a neural language model is through word embeddings.

N-gram

N-gram can be defined as the contiguous sequence of n items from a given sample of text or speech. The items can be letters, words, or base pairs according to the application. The N-grams typically are collected from a text or speech corpus (A long text dataset).

Applications of N-gram Language Model

The N-gram language model has numerous applications in Natural Language Processing. Some of the popular applications are:

Speech Recognition

In speech recognition, the N-gram language model is used to predict the next word in a spoken sentence. This helps in improving the accuracy of the transcription.

Machine Translation

In machine translation, the N-gram language model is used to predict the most likely translation of a sentence based on its context.

Text Classification

In text classification, the N-gram language model is used to classify a text document into different categories based on its content.

Information Retrieval

In information retrieval, the N-gram language model is used to rank search results based on their relevance to the query.

N-gram Language Model EVALUATION AND ESTIMATION

An N-gram language model predicts the probability of a given N-gram within any sequence of words in the language. A good N-gram model can predict the next word in the sentence i.e the value of p(w|h)

Example of N-gram such as unigram (“This”, “article”, “is”, “on”, “NLP”) or bi-gram (‘This article’, ‘article is’, ‘is on’,’on NLP’).

Now, we will establish a relation on how to find the next word in the sentence using

. We need to calculate p(w|h), where is the candidate for the next word. For example in the above example, lets’ consider, we want to calculate what is the probability of the last word being “NLP” given the previous words:

$$p(NLP | this\, article\, is\, on)$$

After generalizing the above equation can be calculated as:

$$p(w_5 | w_1, w_2, w_3, w_4) \, or \, P(W)$$

$$= p(w_n | w_1, w_2...w_n)$$

But how do we calculate it? The answer lies in the chain rule of probability:

$$P(A|B) = \frac{P(A,B)}{P(B)}\\ P(A,B) = P(A|B)P(B)\\$$

Now generalize the above equation:

$$P(X_1,X_2, ...,X_n) = P(X_1) P(X_2 | X_1) P(X_3 | X_1, X_2) .... P(X_n | X_1, X_2,...X_n)\\ P(w_1 w_2 w_3 ...w_n) =\prod_i P(w_i | w_1 w_2 ... w_n)$$

Simplifying the above formula using Markov assumptions:

$P(w_i | w_1, w_2, ...w_{i-1}) \approx P(w_i | w_{i-k},... w_{i-1})$

- **For unigram:**

$P(w_1 w_2, ... w_n) \approx \prod_i P(w_i)$

- **For Bigram:**

$P(w_i | w_1 w_2, ..w_{i-1}) \approx P(w_i | w_{i-1})$

Methods of the Language Modeling

Language modelings are classified as follows:

Statistical language modelings: In this modeling, there is the development of probabilistic models. This probabilistic model predicts the next word in a sequence. For example N-gram language modeling. This modeling can be used for disambiguating the input. They can be used for selecting a probable solution. This modeling depends on the theory of probability. Probability is to predict how likely something will occur.

Neural language modelings: Neural language modeling gives better results than the classical methods both for the standalone models and when the models are incorporated into the larger models on the challenging tasks i.e. speech recognitions and machine translations. One method of performing neural language modeling is by word embedding [1].

N-gram Modelling in NLP

N-gram is a sequence of the N-words in the modeling of NLP. Consider an example of the statement for modeling. “I love reading history books and watching documentaries”. In one-gram or unigram, there is a one-word sequence. As for the above statement, in one gram it can be “I”, "love", "history", "books", "and", "watching", "documentaries". In two-gram or the bi-gram, there is the two-word sequence i.e. "I love", "love reading", or "history books". In the three-gram or the tri-gram, there are the three words sequences i.e. "I love reading", "history books," or "and watching documentaries" [3]. The illustration of the N-gram modeling i.e. for N=1,2,3 is given below in Figure 2 [5].

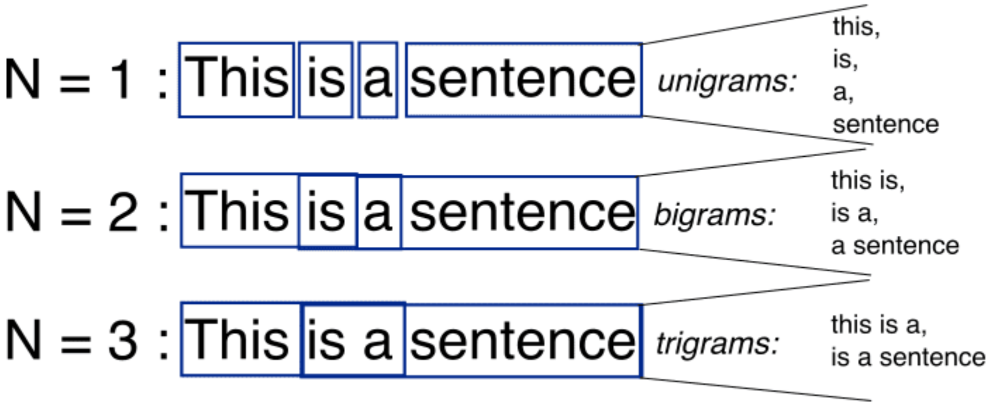


Figure 2 Uni-gram, Bi-gram, and Tri-gram Model

For N-1 words, the N-gram modeling predicts most occurred words that can follow the sequences. The model is the probabilistic language model which is trained on the collection of the text. This model is useful in applications i.e. speech recognition, and machine translations. A simple model has some limitations that can be improved by smoothing, interpolations, and back off. So, the N-gram language model is about finding probability distributions over the sequences of the word. Consider the sentences i.e. "There was heavy rain" and "There was heavy flood". By using experience, it can be said that the first statement is good. The N-gram language model tells that the "heavy rain" occurs more frequently than the "heavy flood". So, the first statement is more likely to occur and it will be then selected by this model. In the one-gram model, the model usually relies on that which word occurs often without pondering the previous words. In 2-gram, only the previous word is considered for predicting the current word. In 3-gram, two previous words are considered. In the N-gram language model the following probabilities are calculated:

P (“There was heavy rain”) = P (“There”, “was”, “heavy”, “rain”) = P (“There”) P (“was” |“There”) P (“heavy”| “There was”) P (“rain” |“There was heavy”).

As it is not practical to calculate the conditional probability but by using the “*Markov Assumptions”*, this is approximated to the bi-gram model as [4]:

P (“There was heavy rain”) ~ P (“There”) P (“was” |“'There”) P (“heavy” |“was”) P (“rain” |“heavy”)

Applications of the N-gram Model in NLP

In speech recognition, the input can be noisy. This noise can make a wrong speech to the text conversion. The N-gram language model corrects the noise by using probability knowledge. Likewise, this model is used in machine translations for producing more natural statements in target and specified languages. For spelling error corrections, the dictionary is useless sometimes. For instance, "in about fifteen minutes" 'minuets' is a valid word according to the dictionary but it is incorrect in the phrase. The N-gram language model can rectify this type of error.

The N-gram language model is generally at the word levels. It is also used at the character levels for doing the stemming i.e. for separating the root words from a suffix. By looking at the N-gram model, the languages can be classified or differentiated between the US and UK spellings. Many applications get benefit from the N-gram model including tagging of part of the speech, natural language generations, word similarities, and sentiments extraction. [4].

Limitations of N-gram Model in NLP

The N-gram language model has also some limitations. There is a problem with the out of vocabulary words. These words are during the testing but not in the training. One solution is to use the fixed vocabulary and then convert out vocabulary words in the training to pseudowords. When implemented in the sentiment analysis, the bi-gram model outperformed the uni-gram model but the number of the features is then doubled. So, the scaling of the N-gram model to the larger data sets or moving to the higher-order needs better feature selection approaches. The N-gram model captures the long-distance context poorly. It has been shown after every 6-grams, the gain of performance is limited.

Language specific modeling problems

Language modeling is a core task in natural language processing (NLP) that involves predicting the next word or phrase given some context. Language models can be used for various applications such as text generation, machine translation, speech recognition, and question answering. However, developing and evaluating language models is not a trivial problem, and there are several challenges and limitations that researchers and practitioners face. In this article, we will discuss some of the current issues related to language modeling benchmarks, which are datasets and metrics that are used to measure the performance and quality of language models.

Data quality and diversity

One of the main challenges of language modeling benchmarks is ensuring that the data used to train and test the models is of high quality and diversity. Data quality refers to the accuracy, completeness, and consistency of the data, while data diversity refers to the variety and coverage of the data in terms of domains, genres, languages, and styles. Poor data quality and diversity can lead to overfitting, bias, and generalization errors in language models. For example, some benchmarks may contain noisy, outdated, or irrelevant data that do not reflect the real-world usage of language. Some benchmarks may also be skewed towards certain topics, domains, or languages that limit the applicability and robustness of language models to other scenarios.

Evaluation metrics and criteria

Another challenge of language modeling benchmarks is choosing and designing appropriate evaluation metrics and criteria that can capture the complexity and nuances of natural language. Evaluation metrics are numerical scores that quantify the performance and quality of language models, while evaluation criteria are the standards and guidelines that define the objectives and expectations of language modeling. However, existing evaluation metrics and criteria may not be sufficient or suitable for all types of language models and tasks. For example, some metrics may only focus on the surface-level accuracy or fluency of the generated text, but neglect the deeper-level aspects such as coherence, relevance, logic, and creativity. Some criteria may also be vague, subjective, or inconsistent, making it hard to compare and interpret the results of different models.

Ethical and social implications

A third challenge of language modeling benchmarks is addressing the ethical and social implications of creating and using language models. Language models can have a significant impact on society, as they can influence how people communicate, learn, and interact with information. However, language models can also pose potential risks and harms, such as generating misleading, offensive, or harmful content, amplifying existing biases and stereotypes, or violating privacy and security. Therefore, language modeling benchmarks should consider the ethical and social dimensions of language modeling, and ensure that the data, models, and metrics are aligned with the values and norms of the intended users and stakeholders. For example, some benchmarks may need to include ethical guidelines, human evaluation, or feedback mechanisms to ensure the quality and safety of the generated content.

Multilingual and Cross lingual Languages

How does multilingual NLP work?

There are many different forms of multilingual NLP, but in general, it enables computational software to understand the language of certain texts, along with contextual nuances. Multilingual NLP is also capable of obtaining specific data and delivering key insights. In short, multilingual NLP technology makes the impossible possible which is to process and analyze large amounts of data. Without it, this kind of task can probably only be executed by employing a very labor- and time-intensive approach.

What makes multilingual NLP difficult to scale?

One of the biggest obstacles preventing multilingual NLP from scaling quickly is relating to low availability of labelled data in low-resource languages.

Among the 7,100 languages that are spoken worldwide, each of them has its own linguistic rules and some languages simply work in different ways. For instance, there are undeniable similarities between Italian, French and Spanish, whilst on the other hand, these three languages are totally different from a specific Asian language group, that is Chinese, Japanese, and Korean which share some similar symbols and ideographs.

The outcome from this leads to the need to have various techniques to generate language models that can work with all these languages. In short, different languages often require different vector spaces, even if there are existing pre-trained language embeddings.

Even though pre-trained word embeddings in different languages exist, it is possible that all of them are in different vector spaces. This means that similar words can signify different vector representations, basically due to the natural characteristics of a certain language.

This is why scaling multilingual NLP applications can be challenging. They use large amounts of labelled data, process it, learn patterns, and generate prediction models. When building NLP on a text comprising different languages, it is best to consider multilingual NLP.

When we need to build NLP on a text containing different languages, we may look at multilingual word embeddings for NLP models that have the potential to scale effectively.

Multilingual NLP Tasks

It is very important to mention Multilingual NLP Tasks. There is still a digital perception that English is the language that everyone knows, that it is innate, and that it is used worldwide. This idea leads to social inequalities and these social inequalities are reflected in the future, in the technological world.

First of all, when machines can analyze a language, they encode and decode not only the linguistic structure but also the culture that this language is connected. On the other hand, the world is far from being a place where nations live in isolation, it is very global. **The whole world interacts with each other. This increases the need for multilingual NLP** and this technology is actively used in many different fields. For example:

- Recruitment processes and analyzing resumes in different languages
- Finance, review, and analysis of financial records in different languages, credit processes
- Security, analysis of criminal records and bureaucratic documents in different languages
- Education, equal opportunities, and analysis of transcripts or texts in different languages
- Fast and secure translation services

Many more important aspects and needs, such as the use of multilingual NLP can be given as examples of Multilingual NLP tasks.

Solutions for tackling multilingual NLP challenges

1, Training specific non-English NLP models

The first suggested solution is to train an NLP model for a specific language. A well-known example would be a few new versions of Bidirectional Encoder Representations from Transformers (BERT) that have been trained in numerous languages.

However, the biggest problem with this approach is its low success rate of scaling. It takes lots of time and money to train a new model, let alone many models. NLP systems require various large models, hence the processes can be very expensive and time-consuming.

This technique also does not scale effectively in terms of inference. Using NLP in different languages means the business would have to sustain different models and provision several servers and GPUs. Again, this can be extremely costly for the business.

2, Leveraging multilingual models

The past years have seen that new emerging multilingual NLP models can be incredibly accurate, at times even more accurate than specific, dedicated non-English language models.

Whilst there are several high-quality pre-trained models for text classification, so far there has not been a multilingual model for text generation with impressive performance.

3, Utilizing translation

The last solution some businesses benefit from is to use translation. Companies can translate their non-English content to English, provide the NLP model with that English content, then translate the result back to the needed language.

What Is Cross-Lingual Language?

Although studies in the field of NLP, or Natural Language Processing, have been carried out for years, all studies in the field have been in English. The vast majority of sentences that machines could understand, or rather perceive and encode, were in English. Breaking this English-dominated orientation and enabling machines to perceive and encode almost every language that exists in a global manner is called Cross-Lingual Language studies.

Cross-Lingual Language NLP is a very difficult and complex process**. The reason for this complexity and difficulty lies in the fundamental differences between languages**. All of the more than 5000 languages spoken around the world have different rules and vectors. So machines need to be trained to recognize these languages

Cross-Lingual Language Models in Machine Learning

We mentioned that developing Cross-Lingual Language models is a very difficult task. We also talked about the fact that developing models for each language individually is a time-consuming, expensive and time-consuming process that has been tried before and has a low success rate. So which models are currently enabling machine learning technologies that can detect different languages?

Many different models make this possible. Different approaches and diversity in the field increase the chances of success. But some models stand out because they are more popular or more successful than others. The most important factor in the popularity of these models is their ease of use. Models that can be developed without the need for too much time and financial resources are of course preferred by many people. Let's take a brief look at some of these popular models.

mBbert

Bert, or Bidirectional Encoder Representations from Transformers, is a very important model. This model works unsupervised with pre-trained data. In other words, none of the large amounts of data used to train the model is labeled. In other words, it has not been human-controlled. In this way, the model can learn and evolve on its own in deep layers.

It can work in more than 100 languages. So what makes it possible?

- **Masked Language Modeling (MLM):** Here, the model randomly masks 15% of the words in the sentence or text. And the sentence has to guess these words. This distinguishes it from other ways of working, such as RNN, because it does not learn words one after the other.

- **Next Sentence Prediction (NSP):** Here the model combines two masked sentences as input. These sentences may or may not be ordered in the text. The model then needs to predict whether these sentences follow one after the other.

Through them, the model gains insight into how languages work. It can perceive languages without human supervision.

XLM

XLM is a model that combines many different models. These are:

- Casual Language Modeling (CLM)

- Just like Bert, a Masked Language Modeling (MLM)

- Translation Language Modeling (TLM)

XLM uses two different pre-training methods. These can be divided into supervised and unsupervised. One is the source language and the other is the target language. XLM has many different checkpoints where it checks whether the correct choice has been made.

Multifit

Muliti-fit is a model that works differently than the other two models. It is based on the

tokenization of subwords, not words, and uses QRNN models.

Let's briefly explain subword tokenization. Morphology is the study of the structure, inflection, and inflection of words. Therefore, working only on "words" does not give accurate results in languages rich in this respect.

In morphologically rich languages such as Turkish, it is necessary to focus on subwords to get accurate language perception results. Because the inflections of words are very common in Turkish, these sublimes are also quite numerous.

The tokenization of these subwords allows the machines to detect words that are not very common.