Faculty Orientation Program on *Natural Language Processing*

[Elective V : 410252 A]
BE Computer Engineering 2019 Course

Organized By
S.T.E.S.'s Sinhgad Institute of Technology
in association with
BOS Computer Engineering, SPPU, Pune
(20th January 2023)

Prof. Deptii Chaudhari
Assistant Professor, Department of Computer Engineering
Hope Foundation's International Institute of Information Technology, Hinjawadi, Pune
deptiic@isquareit.edu.in, www.isquareit.edu.in

Course Objectives

Natural Language Processing

410252(A)

01 Introduction !

To be familiar with fundamental concepts and techniques of natural language processing (NLP)

02 Language Syntax and Semantics: Core Knowledge

To acquire the knowledge of various morphological, syntactic, and semantic NLP tasks

03 Language Modelling: Illustrations

To develop the various language modeling techniques for NLP

04 Integrate

To use appropriate tools and techniques for processing natural languages

05 Recent Advances: Tools and Techniques

To comprehend the advance real world applications in NLP domain.

06 Applications

To describe Applications of NLP and Machine Translations.

Course Outcomes

Natural Language Processing

410252(A)

01 Introduction !
Describe the fundamental concepts of NLP, challenges and issues in NLP

02 Language Syntax and Semantics: Core Knowledge
Analyze Natural languages morphologically, syntactical and semantically

03 Language Modelling: Illustrations
Illustrate various language modelling techniques

04 Integrate
Integrate the NLP techniques for the information retrieval task

05 Recent Advances: Tools and Techniques
Demonstrate the use of NLP tools and techniques for text-based processing of natural languages

06 Applications
Develop real world NLP applications

CO-PO Mapping

@The CO-PO Mapping Matrix

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	2	2	1	-	-	-	-	-	-	-	-	-
CO2	3	3	2	2	2	-	-	-	-	-	-	1
CO3	2	3	3	2	2	-	-	-	-	-	-	2
CO4	2	2	3	3	3	-	2	2	-	-	-	3
CO5	2	2	3	3	3	-	-	-	-	-	-	3
CO6	3	3	3	3	3	2	1	1	-	-	-	3

Sample Questions

Question	CO	BTL
Differentiate between natural languages and programming languages.	CO1	BTL 2
Explain the various types of ambiguities in natural languages.	CO1	BTL 2
Discuss the linguistic levels and stages in NLP.	CO1	BTL 2
Illustrate the working of FST for morphological analysis with an example.	CO2	BTL 4
Select the best parsing tree. (Provide rules along with probabilities)	CO2	BTL 4
Break down the given words into morphemes.	CO2	BTL 4

Teaching Methologies

Live Demos

Prepare small working examples for each concept

Case Studies

Related to Indian or Reginal Languages

Assignments

Based on problem solving
ex: creating parse trees or identifying morphemes

Self Learning

Online Courses NPTEL, Udemy

Mini Projects

Covering all units

Research Papers

Identify research papers & ask students to read & present

Beyond Syllabus

Generative Models, Transformers for NLP

Learning Resources

Text Books:

1. Jurafsky, David, and James H. Martin, ―Speech and Language Processing: An Introduction to Natural Language Processing‖, Computational Linguistics and Speech Recognition‖, , PEARSON Publication

2. Manning, Christopher D., and nrich Schütze , ―Foundations of Statistical Natural Language Processing‖, Cambridge, MA: MIT Press

Reference Books:

1. Steven Bird, Ewan Klein, Edward Loper, ―Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit‖, O'Reilly Publication

2. Dipanjan Sarkar , ―Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data‖, Apress Publication ISBN: 9781484223871

3. Alexander Clark, Chris Fox, and Shalom Lappin, ―The Handbook of Computational Linguistics and Natural Language Processing‖, Wiley Blackwell Publications

4. Jacob Eisenstein, ―Natural Language Processing‖, MIT Press

5. Jacob Eisenstein, ―An Introduction to Information Retrieval‖, Cambridge University Press

Unit I: Introduction to Natural Language Processing

Introduction:

- ✓ What is Natural Language Processing? Why NLP is hard?
- ✓ Programming languages Vs Natural Languages
- ✓ Are natural languages regular?
- ✓ Finite automata for NLP
- ✓ Stages of NLP
- ✓ Challenges and Issues(Open Problems) in NLP

Basics of text processing:

- ✓ Tokenization
- ✓ Stemming,
- ✓ Lemmatization,
- ✓ Part of Speech Tagging

Case Study: Why English is not a regular language

Mapping to CO: CO1

What is Natural Language Processing?

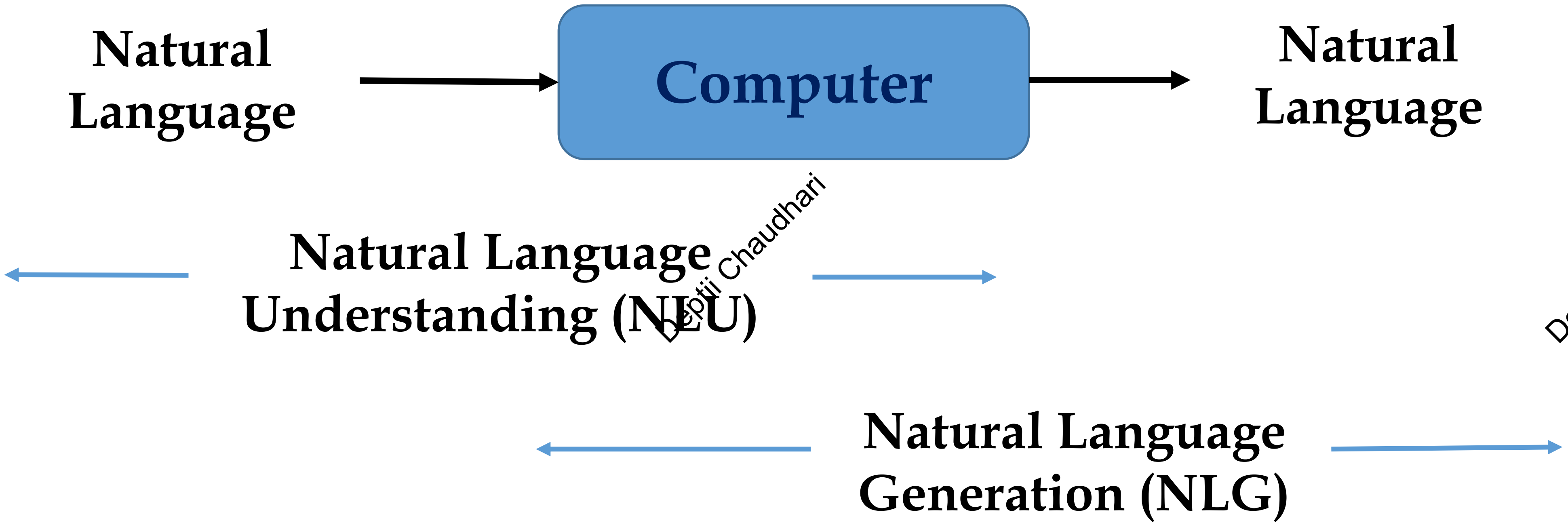
Natural language processing is an area of research in computer science and artificial intelligence (AI) concerned with processing natural languages such as English or Spanish or Hindi or Marathi.

Natural language processing is a process of automating language analysis, generation, acquisition.

- **Analysis ('understanding' or 'processing'):** Input is language, output is some representation that supports useful action.
- **Generation:** Input is some representation, output is language.
- **Acquisition:** Obtaining the representation and necessary algorithms, from knowledge and data

What is Natural Language Processing?

- Computers are using natural languages as input and / or output.


Natural Language
Computer
Natural Language
Natural Language Understanding (NLU)
Natural Language Generation (NLG)

What is Natural Language Processing?

- Computers are using natural languages as input and / or output.

Natural Language Understanding (NLU) | NLP | Natural Language Generation (NLG)

Natural Language Understanding (NLU)

- Take sentences as input and interpret it.
- Map input into useful interpretation.

Analysis

Morphological
Syntactic
Semantic
Discourse

Natural Language Generation (NLG)

- Taking some representation what you want to say and working out a way to express it in natural language.

Analysis

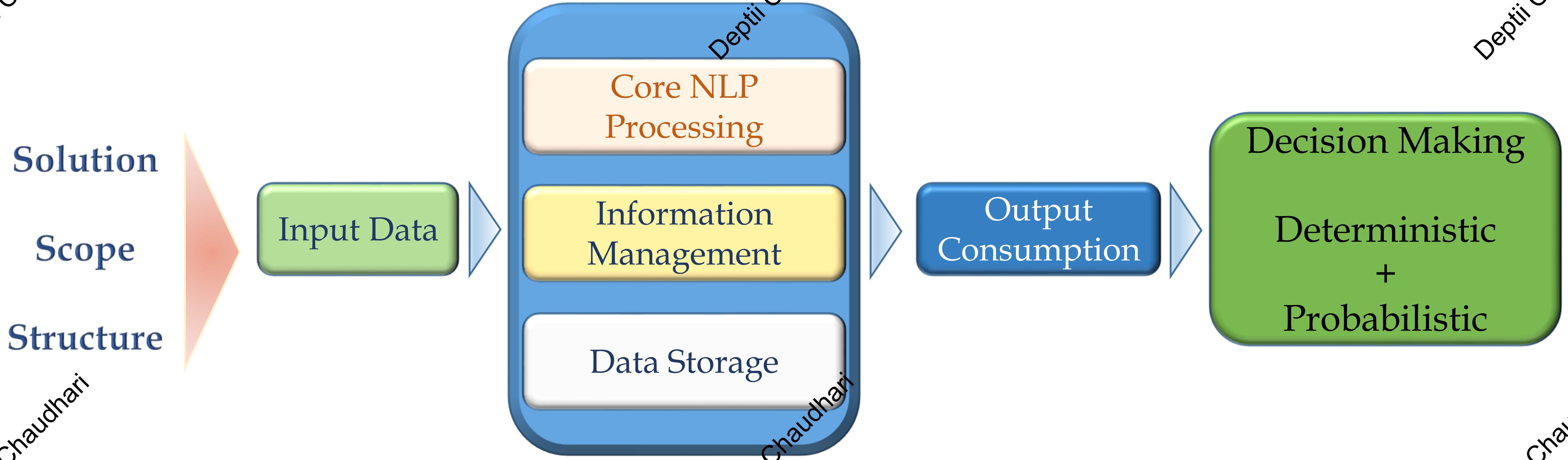
Deep Planning
Syntactic Parsing

Core NLP Pipeline

Text Pre-processing
- Noise Removal
- Lexicon Normalization
- Object Standardization

Feature Engineering
- Syntactic Parsing
- Part of Speech Tagging
- Entity Extraction
- Statistical Features
- Word Embeddings (Text Vectors)

NLP Tasks
- Text Classification
- Text Matching/Similarity
- Coreference Resolution
- Temporal Sequencing
- NLU
- NLG



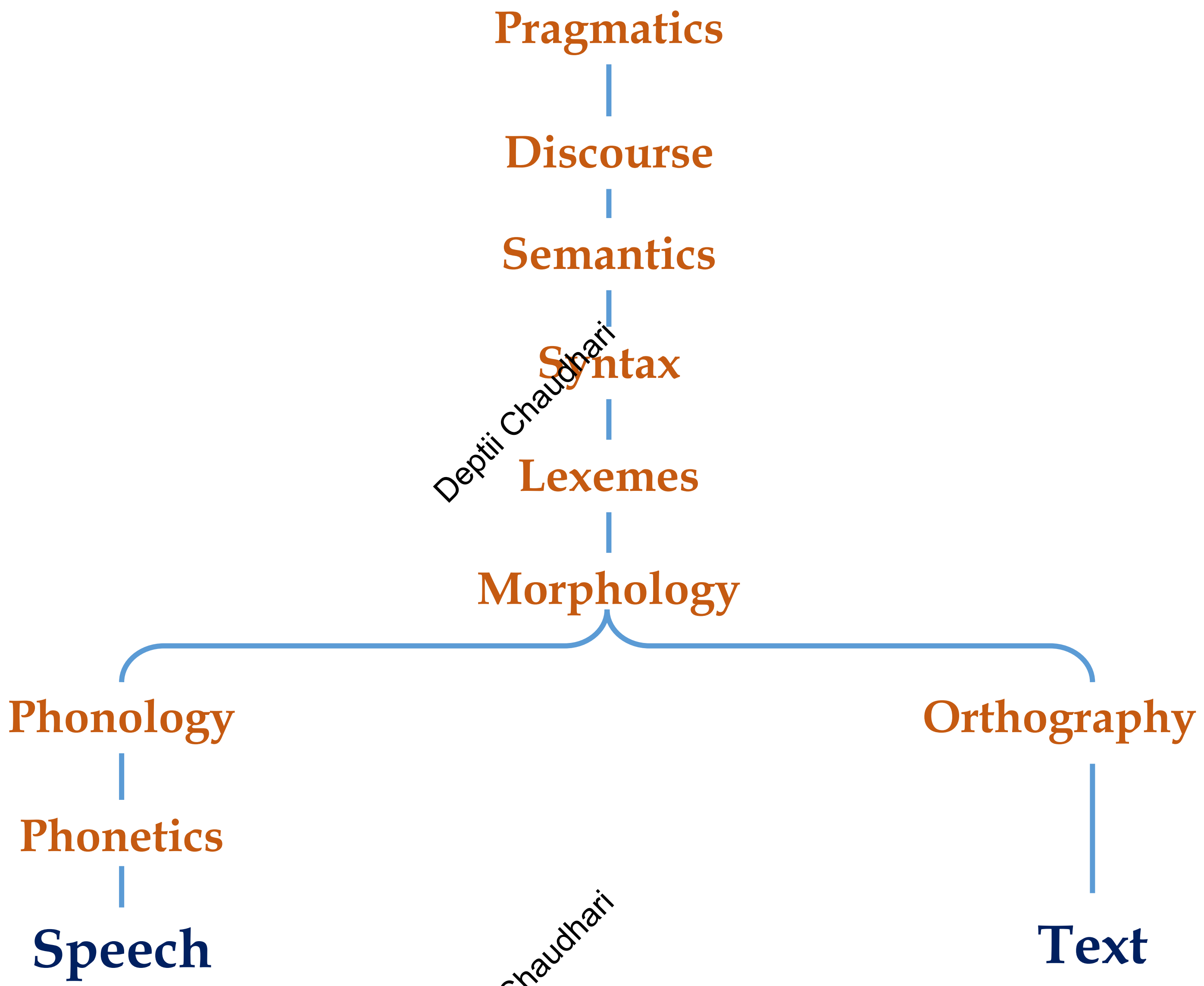
Alternative Views of NLP

- **Computational models of human language processing**
  - Programs that operate internally the way humans do
- **Computational models of human communication**
  - Programs that interact like humans
- **Computational systems that efficiently process text and speech**

Goals of NLP

- **Science Goal:** Understand the way language operates
- **Engineering Goal:** Build systems that analyze and generate language; reduce the man machine gap

Levels of Linguistic Representation

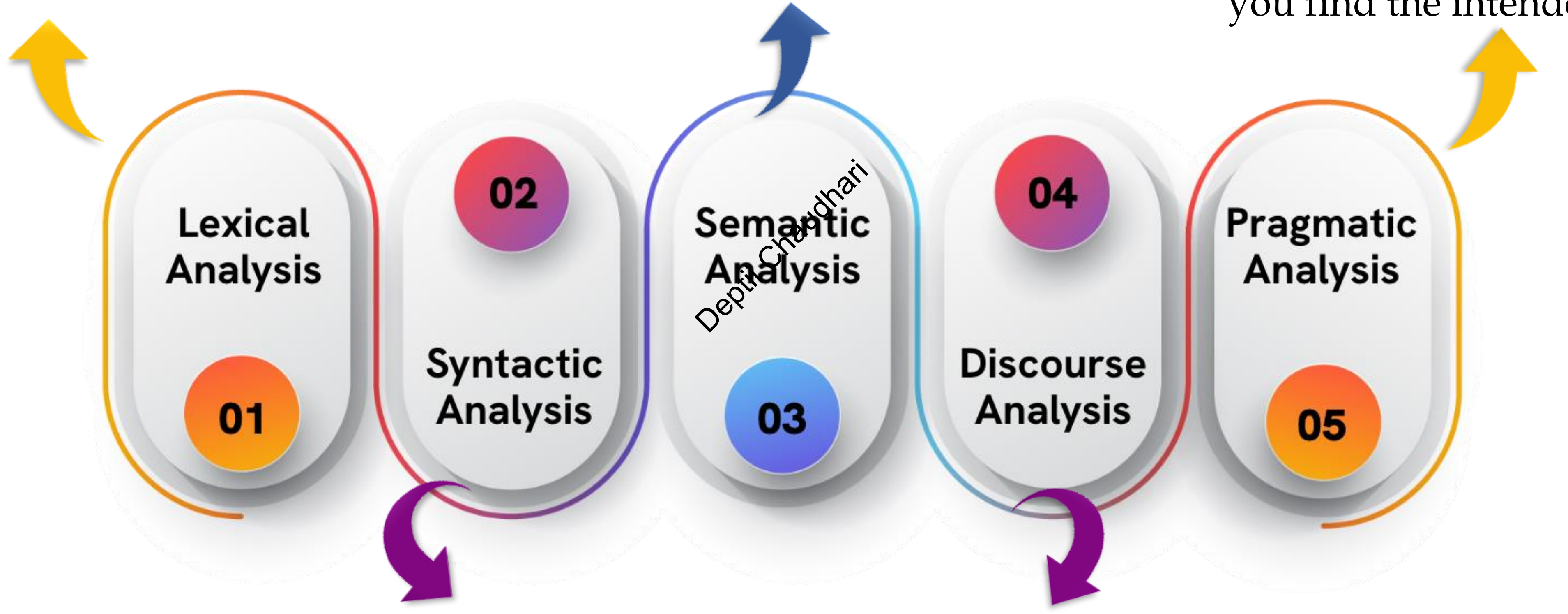


Stages of NLP

Process of converting a sequence of characters into a sequence of tokens

Process of looking for meaning in a statement

Uses a set of rules that describe cooperative dialogues to help you find the intended result



Process for checking grammar, arranging words, and displaying relationships between them.

Deals with the effect of a previous sentence on the sentence in consideration.

Why NLP is hard?

"At last a computer that understands you like your mother."

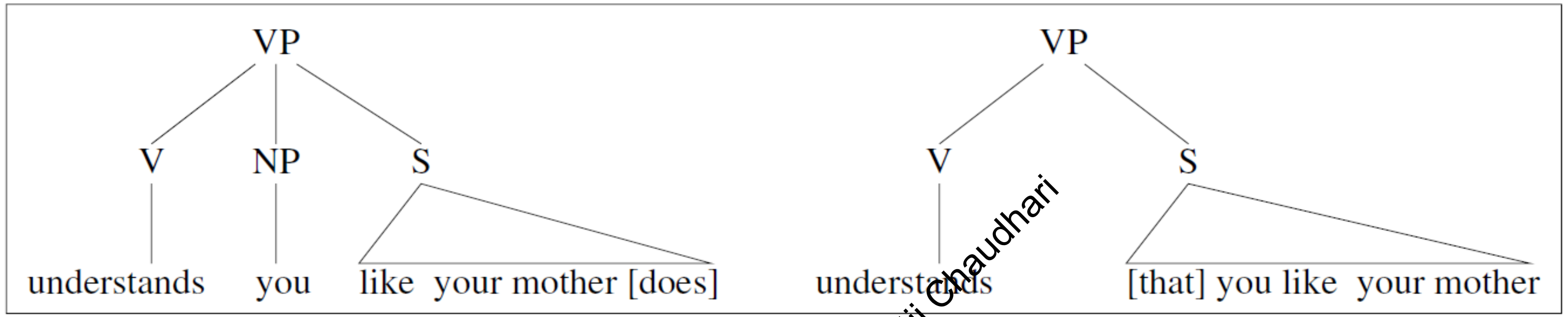
1. It understands (that) you like your mother.

2. It understands you as well as it understands your mother.

Ambiguity

Ambiguity at Many Levels

- **At acoustic level**
  - **Homophones:** words that sound similar but means different.
    - E.g. I am going to buy an apple. "Apple" (Company) Vs. "apple" (Fruit)
  - **Word Boundary:** 'Aajayenge' (Will come) Vs. 'Aaj ayenge' (Will come today)
  - **Phrase Boundary:** Sentence "I got a plate" can be broken up in two different ways
    - Either to mean "I got up late", which means I woke up late
    - Or "I got a plate", I have a plate with me
  - **Disfluency:** concerned with how a speaker intersperses his sentences with meaningless sounds just to be able to organize his/her thought.
    - E.g.: ummm, ahhh, ahem etc
- **At Lexical level**
  - **Part of speech:** Dog (as noun) Vs. Dog (as verb)
  - **Sense:** Dog (as animal) Vs. Dog (as a very detestable person)
- **At the syntactic level**



Different structures lead to different interpretations.

Ambiguity at Many Levels

- **Structural**
  - “The camera man shot the man with the gun when he was near Tendulkar.”
  - “Aid for kins of cops, killed in terrorist attacks”
- **At Semantic Level**
  - Word Sense Ambiguity
  - They put money in the ***bank***.
  - = buried in mud?
  - I saw a boy with a telescope.
- **At Discourse (multi-clause) level**
  - Alice says they’ve built a computer that understands you like your mother
  - But she . . .
  - . . . doesn’t know any details . . .
  - ….. doesn’t understand me at all

Programming Languages Vs Natural Languages

Programming Languages	Natural Languages
Simple and unambiguous	Complex, open to interpretation and evolve
Intended to be translated into a finite set of mathematical operations	Not intended to be translated into a finite set of mathematical operations
Tells a machine exactly what to do using a compiler or interpreter	No compilers or interpreters for natural languages
None of these applies to programming languages.	Communicate is both logical and emotional ways. Involves body language, intonation, volume and many other nonverbal clues. Defined by the physical attributes of human bodies (eyes, tongue, hands), and are for that reason unique to humans.
Follow very strict set of rules, hence they can't evolve and develop the same way human languages do (although we could say that programming languages evolve through various libraries).	Evolves over the period of time. E.g. Slangs used on social media, new words are added to webster dictionary every year

Programming Languages Vs Natural Languages

Programming Languages	Natural Languages
No room for errors or improvisation	Human languages are full of imperfections
No variation or different aspects of same programming language	Multiple aspects of human languages: Dialects, slang, jargon, argot (secret language used by a certain group that's incomprehensible to outsiders), namesake, accents, mispronounced words, typos, irregular punctuation Don't disrupt the message we're trying to communicate
Artificial creations: Rules and definitions were designed beforehand, which allows for them to be fully described and studied in their entirety. Self-defining grammar which doesn't change depending on the context.	Natural creations, grammar changes as per context.
Programming languages don't really have morphology, at least not the same way human languages do	Morphology is very important

Challenges and Issues (Open Problems) in NLP

01 Context of words, phrases and Homonyms

02 Large set of morphological variants

03 Irony, Sarcasm, Emotions, Intentions

04 Ambiguity at many levels

05 Errors in text, speech: Misspellings, Unstructured data

06 Colloquialisms and slang e.g. gonna, wanna, granny, granma

07 Domain specific / Low Resource Languages

08 Lack of data, benchmarks, standards

Basics of Text Processing

Process of breaking down a text into tokens or given paragraph into a list of sentences or words

Tokenization

- Process of reducing the inflectional words to their root forms
- Maps the word to a same stem even if the stem is not a valid word in the language

Stemming

Lemmatization, unlike stemming reduces the inflected words properly ensuring that the root word belongs to the language

Lemmatization

A process that attaches each word in a sentence with a suitable tag from a given set of tags

Part of Speech Tagging

Tokenization

Why Tokenize?

- Unstructured data and natural language text is broken into chunks of information that can be understood by machine.
- Converts an unstructured string (text document) into a numerical data structure suitable for machine learning. which allows the machines to understand each of the words by themselves, as well as how they function in the larger text.
- First crucial step of the NLP process as it converts sentences into understandable bits of data for the program to work with.
- Without a proper / correct tokenization, the NLP process can quickly devolve into a chaotic task.

Challenges

- Dealing with segment words when spaces or punctuation marks define the boundaries of the word. For example: donâ€™t
- Dealing with symbols that might change the meaning of the word significantly. For example: ₹100 vs 100
- Contractions such as 'you're' and 'I'm'
- Not applicable for symbol based languages like Chinese, Japanese, Korean Thai, Hindi, Urdu, Tamil, and others

Types of Tokenization

1. Word Tokenization

- Most common way of tokenization, uses natural breaks, like pauses in speech or spaces in text, and splits the data into its respective words using delimiters (characters like ‘,’ or ‘;’ or ‘“,”’).
- Word tokenization’s accuracy is based on the vocabulary it is trained with. Unknown words or Out Of Vocabulary (OOV) words cannot be tokenized.

2. White Space Tokenization

- Simplest technique, Uses white spaces as basis of splitting.
- Works well for languages in which the white space breaks apart the sentence into meaningful words.

3. Rule Based Tokenization

- Uses a set of rules that are created for the specific problem.
- Rules are usually based on grammar for particular language or problem.

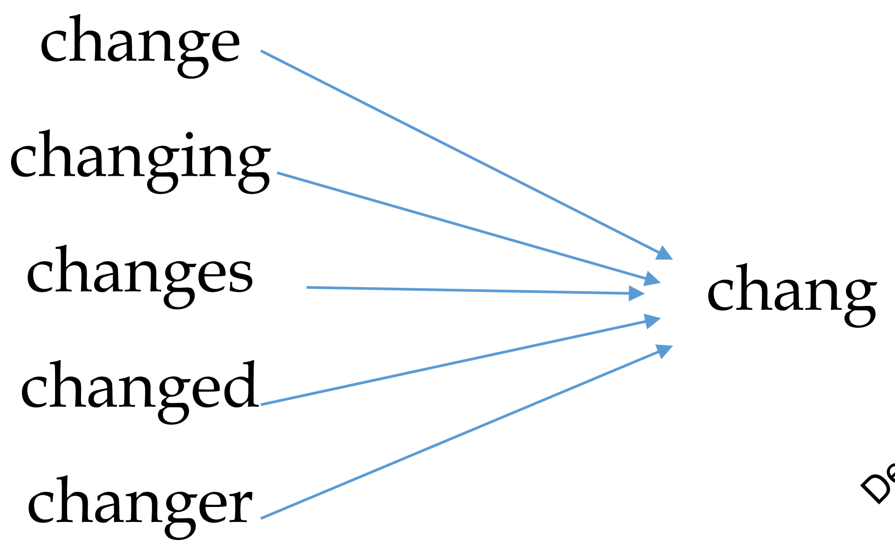
4. Regular Expression Tokenizer

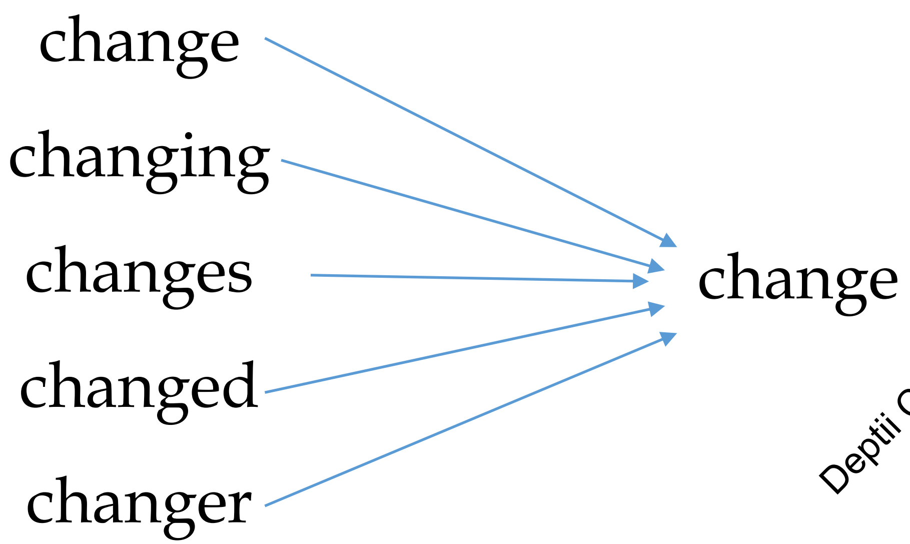
- Type of Rule based tokenizer
- Uses regular expression to control the tokenization of text into tokens.

5. Penn Treebank Tokenizer

- Penn Treebank is a corpus maintained by the University of Pennsylvania containing over four million and eight hundred thousand annotated words in it, all corrected by humans
- Uses regular expressions to tokenize text as in Penn Treebank

Stemming Vs Lemmatization


$$\begin{tikzcd}
\text{change} \arrow[dr] & \\
\text{changing} \arrow[r] & \text{chang} \\
\text{changes} \arrow[u] & \\
\text{changed} \arrow[uu] & \\
\text{changer} \arrow[uuu] &
\end{tikzcd}$$


$$\begin{tikzcd}
\text{change} \arrow[dr] & \\
\text{changing} \arrow[r] & \text{change} \\
\text{changes} \arrow[u] & \\
\text{changed} \arrow[uu] & \\
\text{changer} \arrow[uuu] &
\end{tikzcd}$$

Stemming

- **Porter Stemming:** Uses suffix stripping to produce stems
- **Lancaster Stemming:** Works with a table containing about 120 rules indexed by the last letter of a suffix.

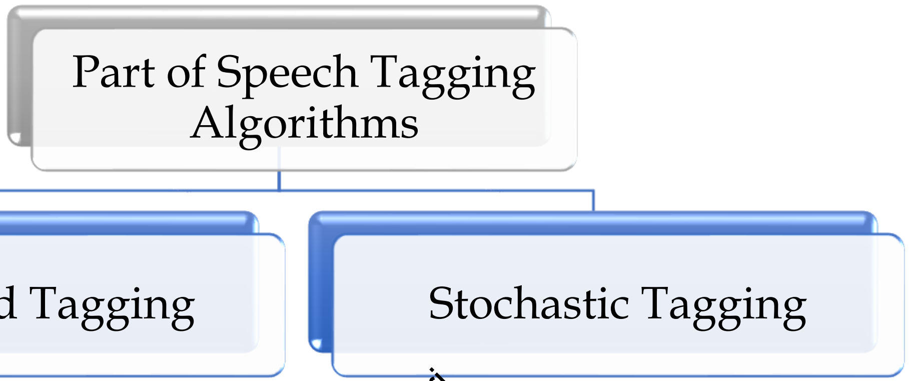
Lemmatization

- Wordnet Lemmatization: Uses WordNet database to lookup lemmas of the words.

Parts of Speech Tagging

- Part of Speech tagging (or just tagging for short) is the process of assigning a part of- speech or other syntactic class marker to each word in a corpus.
- Because tags are generally also applied to punctuation, tagging requires that the punctuation marks (period, comma, etc) be separated off of the words.
- Thus tokenization is usually performed before, or as part of, the tagging process, separating commas, quotation marks, etc., from words, and disambiguating end-of-sentence punctuation (period, question mark, etc) from part-of-word punctuation (such as in abbreviations like e.g. and etc.)
- The input to a tagging algorithm is a string of words and a specified tagset of the kind. The output is a single best tag for each word.
- Automatically assigning a tag to each word is not trivial because of the ambiguity.
- For example:
  - ***Book*** that flight OR ***Book*** that suspect. book --> verb.
  - Hand me that book. book --> noun.

Types of POS Taggers



- Involves a large database of hand-written disambiguation rules
- Disambiguation is done by analyzing the linguistic features of the word, its preceding word, its following word, and other aspects.
- Example of a rule:
- If an ambiguous/unknown word X is preceded by a determiner and followed by a noun, tag it as an adjective.
- Example of Rule-based tagger is Brill's Tagger.

- Any model which somehow incorporates frequency or probability may be properly labelled stochastic.
- The simplest stochastic taggers disambiguate words based solely on the probability that a word occurs with a particular tag.
- The problem with this approach is that while it may yield a valid tag for a given word, it can also yield inadmissible sequences of tags.
- An alternative approach is to calculate the probability of a given sequence of tags occurring known as n-gram approach, referring to the fact that the best tag for a given word is determined by the probability that it occurs with the n previous tags.

Hidden Markov Model Tagging

- Section 5.5 - HMM PART-OF-SPEECH TAGGING - Jurafsky, David, and James H. Martin, ―Speech and Language Processing: An Introduction to Natural Language Processing , Computational Linguistics and Speech Recognition
- https://www.freecodecamp.org/news/an-introduction-to-part-of-speech-tagging-and-the-hidden-markov-model-953d45338f24/
- https://www.freecodecamp.org/news/a-deep-dive-into-part-of-speech-tagging-using-viterbi-algorithm-17c8de32e8bc
- https://medium.com/data-science-in-your-pocket/pos-tagging-using-hidden-markov-models-hmm-viterbi-algorithm-in-nlp-mathematics-explained-d43ca89347c4

Unit II: Language Syntax and Semantics

Morphological Analysis:

- ✓ What is Morphology?
- ✓ Types of Morphemes
- ✓ Inflectional morphology & Derivational morphology
- ✓ Morphological parsing with Finite State Transducers (FST)

Syntactic Analysis:

- ✓ Syntactic Representations of Natural Language,
- ✓ Parsing Algorithms,
- ✓ Probabilistic context-free grammars, and Statistical parsing

Semantic Analysis:

- ✓ Lexical Semantic,
- ✓ Relations among lexemes & their senses –Homonymy, Polysemy, Synonymy, Hyponymy, WordNet, Word Sense Disambiguation (WSD)
- ✓ Dictionary based approach
- ✓ Latent Semantic Analysis

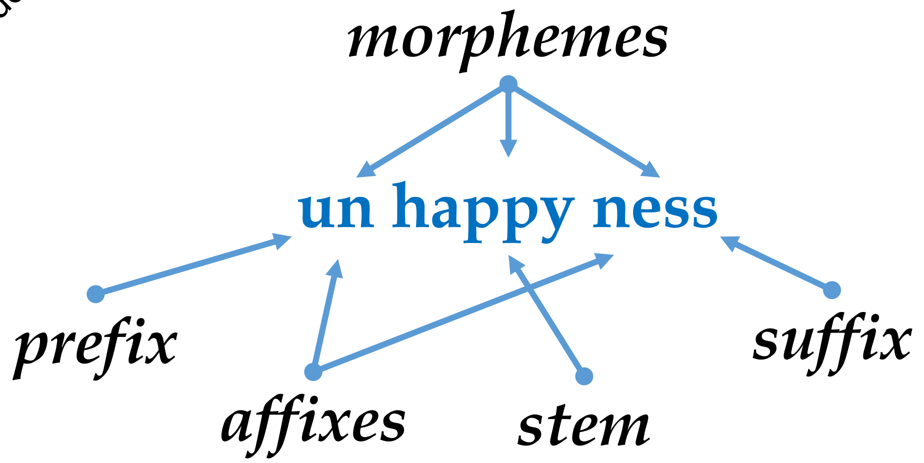
Case Studies: Study of Stanford Parser and POS Tagger https://nlp.stanford.edu/software/lex-parser.html, https://nlp.stanford.edu/software/tagger.html

Mapping to CO: CO2

Morphological Analysis

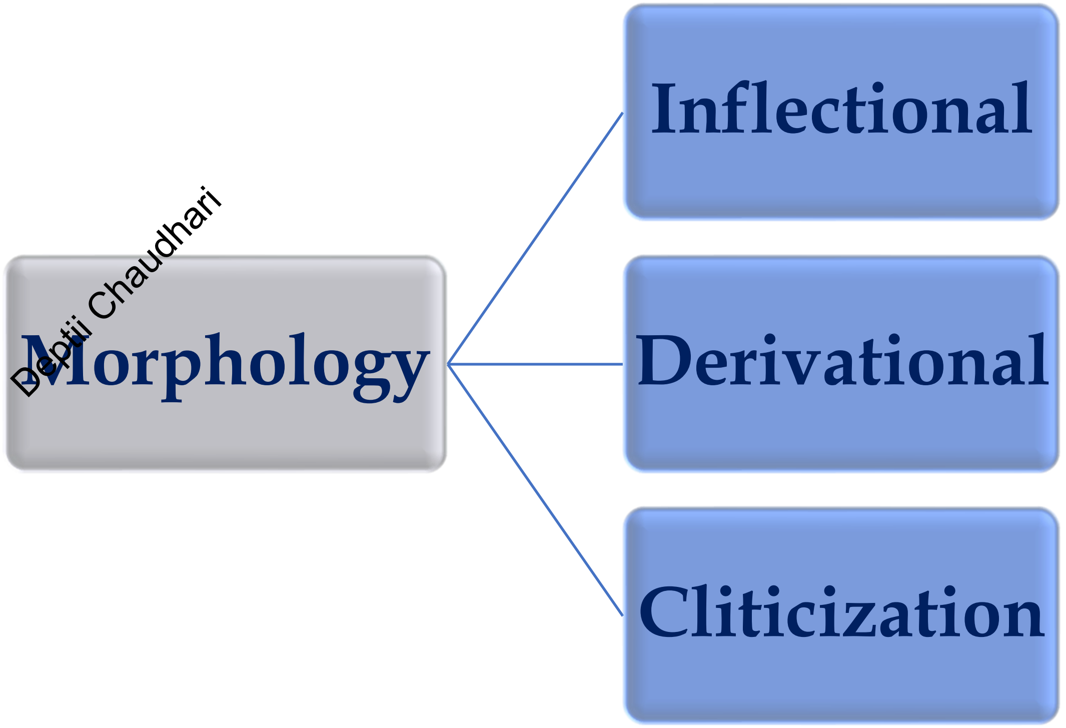
Morphology is the study of the structure and formation of words.

Morpheme is the important unit of morphology, which is defined as the "minimal unit of meaning“ or “the minimal unit of grammatical analysis"



- There are three morphemes, each carrying a certain amount of meaning. *un* means "not", while *ness* means "being in a state or condition".
- *Happy* is a *free morpheme* because it can appear on its own (as a "word" in its own right).
- *Bound morphemes* have to be attached to a free morpheme, and so cannot be words in their own right.
- Thus, you can't have sentences in English such as "Jason feels very un ness today".

Types of Morphology



Inflection is the process of changing the form of a word so that it expresses information such as number, person, case, gender, tense, mood and aspect, but the **syntactic category of the word remains unchanged.**
Examples: car / cars , table / tables, dog / dogs

Creation of a new word from existing word by **changing grammatical category**.
Example: happiness, brotherhood etc.

Cliticization is the combination of a word stem with a clitic. Clitic is a morpheme that **acts syntactically like a word, but is reduced in form** and attached (phonologically and sometimes orthographically) to another word.
For example the English morpheme ’ve in the word I’ve is a clitic.

Morphological Parsing

In order to build a morphological parser, we'll need at least the following:

1. lexicon: the list of stems and affixes, together with basic information about them (whether a stem is a Noun stem or a Verb stem, etc.).

2. morphotactics: the model of morpheme ordering that explains which classes of morphemes can follow other classes of morphemes inside a word.

3. orthographic rules: these spelling rules are used to model the changes that occur in a word, usually when two morphemes combine (e.g., the y→ie spelling rule discussed above that changes city + -s to cities rather than citys)

Syntactic Analysis

- The study of grammar has an ancient lineage; Panini’s grammar of Sanskrit was written over two thousand years ago, and is still referenced today in teaching Sanskrit.
- The word syntax comes from the Greek s´yntaxis, meaning “setting out together or arrangement”, and refers to the way words are arranged together.
- Syntax refers to **the set of rules, principles, processes that govern the structure of sentences in a natural language**.
- Syntactic analysis, also referred to as syntax analysis or parsing, is the process of analyzing natural language with the rules of a formal grammar.

Notions of Syntax and Grammar

- **Constituency**
  - Groups of words may behave as a single unit or phrase, called a constituent.
  - Example: ***On September seventeenth***, I'd like to fly from Pune to Delhi.
- **Grammatical relations**
  - A formalization of ideas from traditional grammar such as SUBJECTS and OBJECTS, and other related notions.
  - Example: She ate a mammoth breakfast. Here noun phrase She is the SUBJECT and a mammoth breakfast is the OBJECT.
- **Subcategorization and dependency relations**
  - Refer to certain kinds of relations between words and phrases.
  - For example: The verb ***want*** can be followed by an infinitive, as in I want to fly to Delhi, or a noun phrase, as in I want a flight to Delhi. But the verb ***find*** cannot be followed by an infinitive (*I found to fly to Delhi).
  - These are called facts about the subcategorization of the verb.

Context Free Grammars

- A context-free grammar consists of a set of rules or productions, each of which expresses the ways that symbols of the language can be grouped and ordered together, and a lexicon of words and symbols.
- The symbols that are used in a CFG are divided into two classes.
- The symbols that correspond to words in the language are called **terminal symbols**; the lexicon is the set of rules that introduce these terminal symbols.
- The symbols that express clusters or generalizations of these are called **non-terminals**.
- In each context free rule, the item to the right of the arrow (→) is an ordered list of one or more terminals and non-terminals, while to the left of the arrow is a single non-terminal symbol expressing some cluster or generalization.

Context Free Grammars

- A context-free grammar G is defined by four parameters N, S, P, S (technically "is a 4-tuple"):
  - N a set of non-terminal symbols (or variables)
  - ∑ a set of terminal symbols (disjoint from N)
  - R a set of rules or productions, each of the form A→b , where A is a nonterminal, β is a string of symbols from the infinite set of strings $(\sum \cup N)*$
  - S a designated start symbol

Context Free Grammars

Noun	$\rightarrow$	*flights* | *breeze* | *trip* | *morning* | …
Verb	$\rightarrow$	*is* | *prefer* | *like* | *need* | *want* | *fly*
Adjective	$\rightarrow$	*cheapest* | *non – stop* | *first* | *latest*
		| *other* | *direct* | …
Pronoun	$\rightarrow$	*me* | *I* | *you* | *it* | …
Proper-Noun	$\rightarrow$	*Alaska* | *Baltimore* | *Los Angeles*
		| *Chicago* | *United* | *American* | …
Determiner	$\rightarrow$	*the* | *a* | *an* | *this* | *these* | *that* | …
Preposition	$\rightarrow$	*from* | *to* | *on* | *near* | …
Conjunction	$\rightarrow$	*and* | *or* | *but* | …

The lexicon for L_0.

S	$\rightarrow$	*NP VP*	I + want a morning flight
NP	$\rightarrow$	*Pronoun*	I
	|	*Proper-Noun*	Los Angeles
	|	*Det Nominal*	a + flight
Nominal	$\rightarrow$	*Nominal Noun*	morning + flight
	|	*Noun*	flights
VP	$\rightarrow$	*Verb*	do
	|	*Verb NP*	want + a flight
	|	*Verb NP PP*	leave + Boston + in the morning
	|	*Verb PP*	leaving + on Thursday
PP	$\rightarrow$	*Preposition NP*	from + Los Angeles

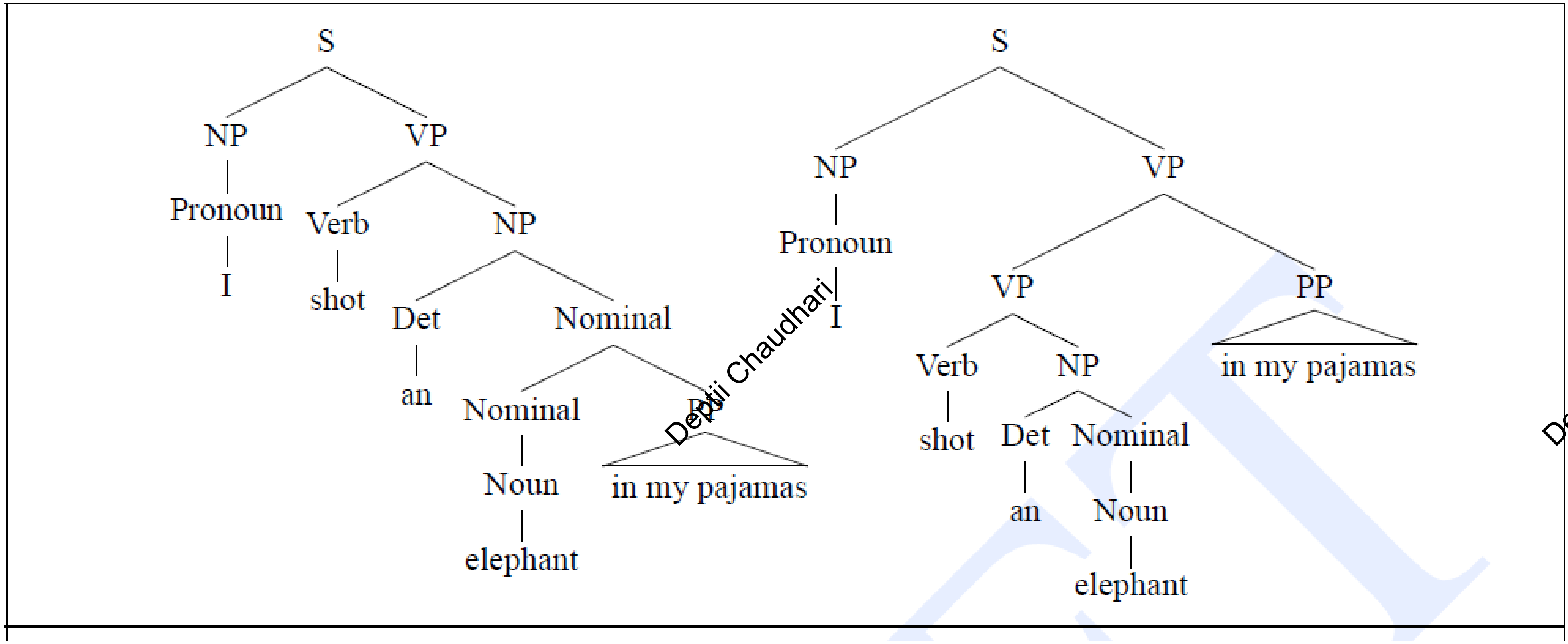
The grammar for L_0

Example phrases for each rule

Parsing Algorithms

- **Top Down Parsing**
  - Goal driven searching
  - Gives importance to textual precedence (rule precedence)
  - Problems with Top Down Parsing
    1. Only judges grammaticality.
    2. Stops when it finds a single derivation.
    3. No semantic knowledge employed.
    4. No way to rank the derivations.
    5. Problems with left-recursive rules.
    6. Problems with ungrammatical sentences.
- **Bottom-up Parsing**
  - Starts with the words of input and tries to build trees from words up, again by applying rules from the grammar one at a time.

Parsing Algorithms - Ambiguity



Dependency Parsing

- Dependency Parsing is the process to analyze the grammatical structure in a sentence and find out related words as well as the type of the relationship between them.
- Each relationship:
  - Has one head and a dependent that modifies the head.
  - Is labeled according to the nature of the dependency between the head and the dependent. These labels can be found at Universal Dependency Relations.
- Dependency parsing is the task of extracting a dependency parse of a sentence that represents its grammatical structure and defines the relationships between “head” words and words, which modify those heads.

- Read more about Dependency Parsing in Chapter 15 Speech and Language Processing. Daniel Jurafsky & James H. Martin

Probabilistic Context Free Grammars

- The simplest augmentation of the context-free grammar is the **Probabilistic Context-Free Grammar (PCFG)**, also known as the **Stochastic Context-Free Grammar (SCFG**
- A probabilistic context-free grammar augments each rule in *P* with a conditional probability.
- A PCFG is thus defined by the following components

N	a set of non-terminal symbols
∑	a set of terminal symbols
R	a set of rule production, each of the form A --> β where A is a non-terminal, β is a string of symbols from the infinite set of strings (∑ ∪ N)* and p is a number between 0 and 1 expressing P(β |A)
S	a designated start symbol

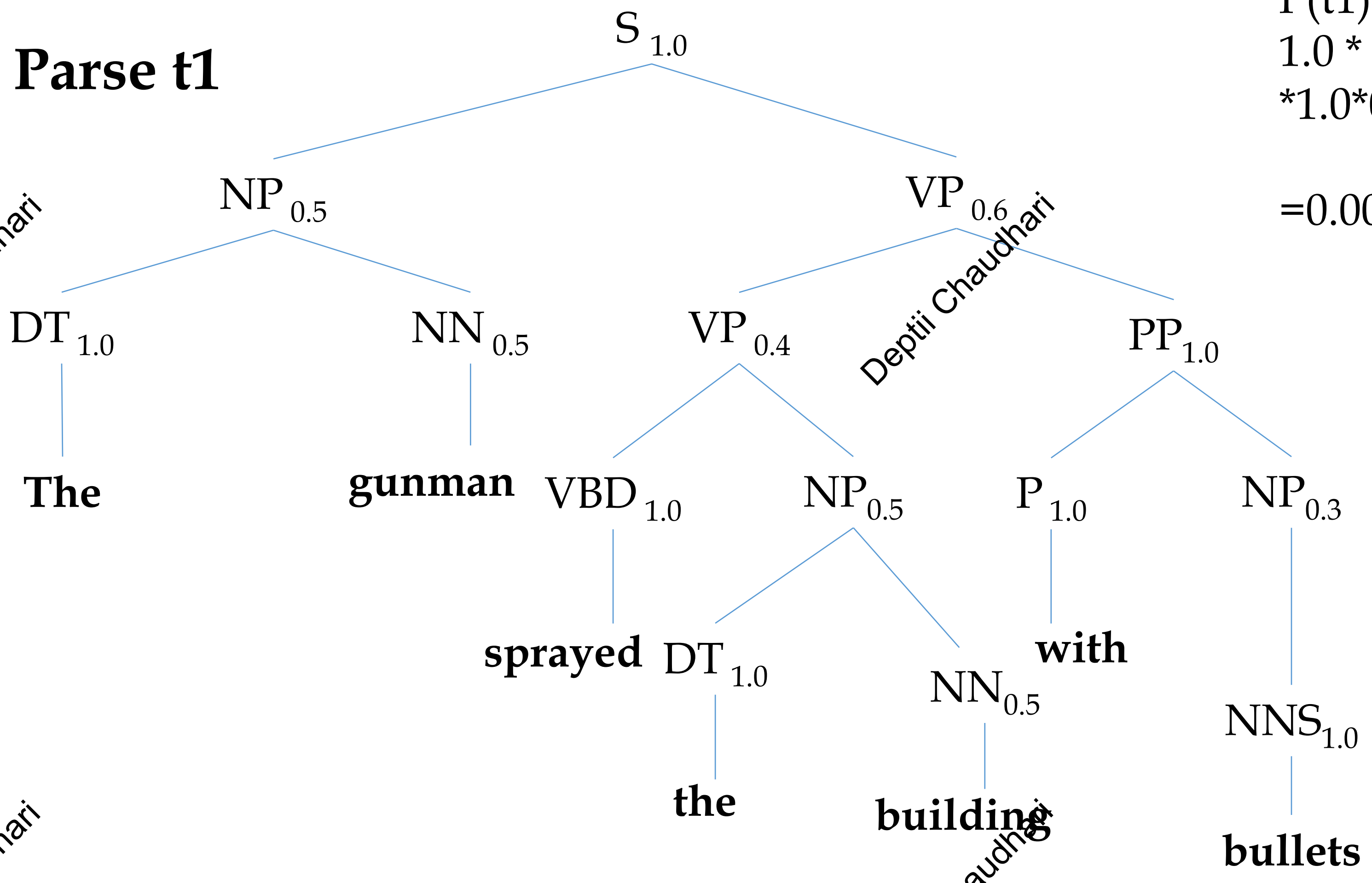
Probabilistic Context Free Grammars

$S \rightarrow NP\ VP$	0.1	$DT \rightarrow$ the	0.1
$NP \rightarrow DT\ NN$	0.5	$NN \rightarrow$ gunman	0.5
$NP \rightarrow NNS$	0.3	$NN \rightarrow$ building	0.5
$NP \rightarrow NP\ PP$	0.2	$VBD \rightarrow$ sprayed	1.0
$PP \rightarrow P\ NP$	1.0	$NNS \rightarrow$ bullets	1.0
$VP \rightarrow VP\ PP$	0.6		
$VP \rightarrow VBD\ NP$	0.4		

Probabilistic Context Free Grammars

The gunman sprayed the building with bullets.

Parse t1



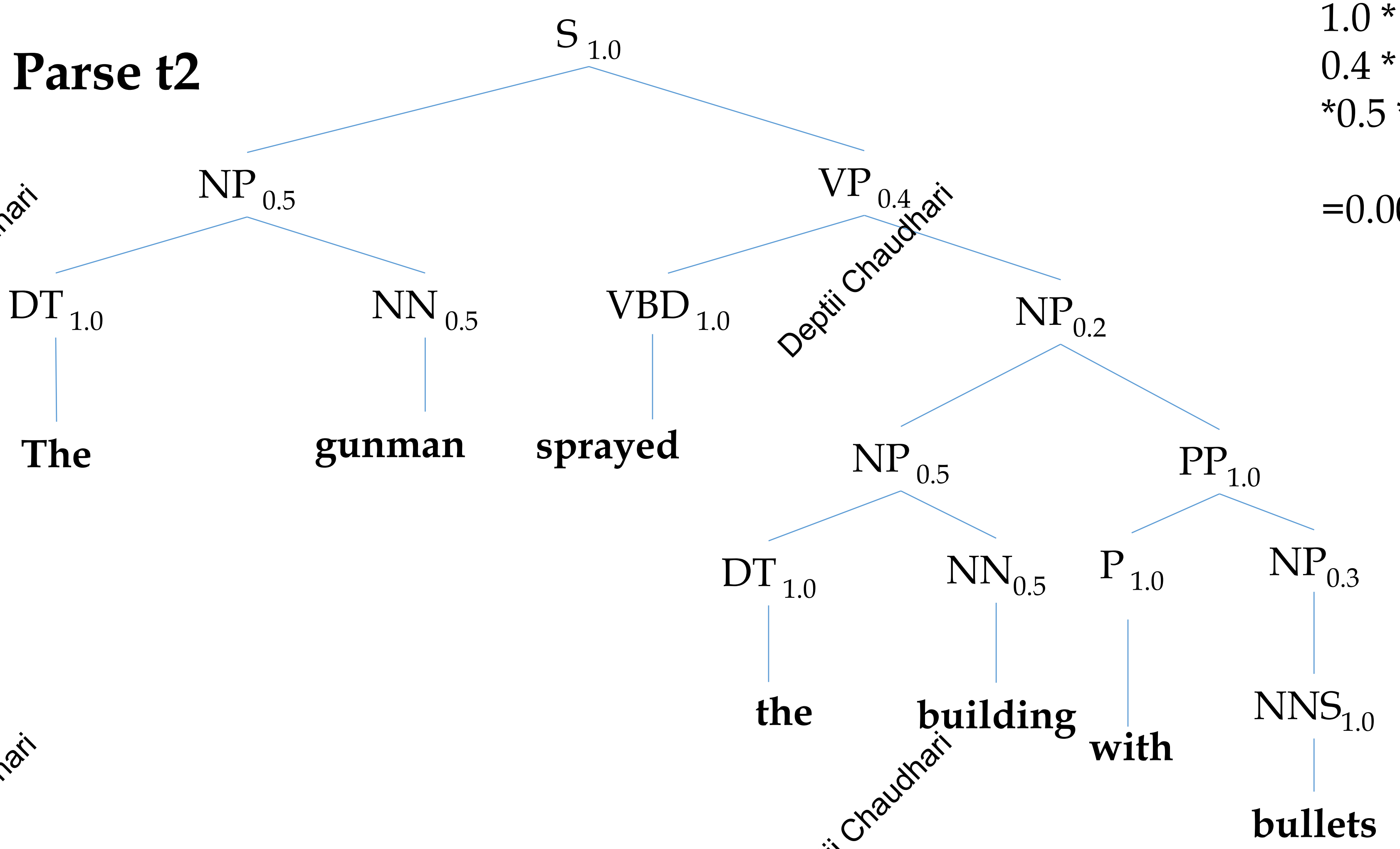
P(t1) =
1.0 * 0.5*1.0*0.5*0.6*0.4 *1.0 *0.5 *1.0*0.5*1.0*1.0*0.3 *0.1

=0.00225

Probabilistic Context Free Grammars

The gunman sprayed the building with bullets.

Parse t2



P(t2) =
1.0 * 0.5* 1.0* 0.5*
0.4 * 1.0 * 0.2 * 0.5* 1.0
*0.5 *0.1 *0.1 *0.3 *1.0

=0.0015

Lexical Semantic

- **Lexeme** is a pairing of a particular form (orthographic or phonological) with its meaning, and a **lexicon** is a finite list of lexemes.
- A **lemma** or **citation form** is the grammatical form that is used to represent a lexeme. Example: The lemma or citation form for sing, sang, sung is sing.
- The process of mapping from a wordform to a lemma is called **lemmatization**.
- Lexical Semantics is the relationship of lexical meaning to sentence meaning and syntax.
- Lexical semantics is concerned with the intrinsic characteristics of word meaning, semantic relationships between words, and how word meaning is related to syntactic structure.

Relations between lexical items

- **Hyponymy and hypernymy**
  - Hyponymy and hypernymy refers to a relationship between a general term and the more specific terms that fall under the category of the general term.
  - For example: the colors red, green, blue and yellow are hyponyms. They fall under the general term of color, which is the hypernym.
- **Synonymy** refers to the words that are pronounced and spelled differently but contain the same meaning.
  - When the meaning of two senses of two different words (lemmas) are identical or nearly identical we say the two senses are synonyms.
  - Synonyms include such pairs as: couch/sofa , car/automobile
- **Antonyms**, by contrast, are words with opposite meaning such as the following: long/short, big/little, fast/slow, cold/hot, dark/light

Relations between lexical items

- **Homonymy** refers to the relationship between words that are spelled or pronounced the same way but hold different meanings.
  - For Example: bank (of river) , bank (financial institution)
- **Polysemy** refers to a word having two or more related meanings.
  - For Example: bright (shining), bright (intelligent)
- Lexical semantics also explores whether the meaning of a lexical unit is established by looking at its neighborhood in the semantic net, (words it occurs with in natural sentences), or whether the meaning is already locally contained in the lexical unit.
- In English, **WordNet** is an example of a semantic network.
- It contains English words that are grouped into synsets. Some semantic relations between these synsets are meronymy, hyponymy, synonymy, and antonymy.

NLP Online Courses

https://lenavoita.github.io/nlp_course/word_embeddings.html

https://www.fast.ai/posts/2019-07-08-fastai-nlp.html

https://www.udemy.com/course/natural-language-processing-with-bert/

Write to Me..

deptiic@isquareit.edu.in

Thank You!