Natural Language Processing Introduction

M E Palanivel
Professor
SITAMS

Course Objectives & Syllabus

- Basics of NLP, Morphology, Tokenization, N-gram Models
- POS Tagging, Parsing, Treebank's, Ambiguity Handling
- Word Sense Disambiguation, Semantic Parsing, Sentiment Analysis
- Machine Translation, Transformers, BERT/GPT, Ethical NLP
- Speech Recognition, Feature Extraction, Discourse Analysis

UNIT I: Introduction to NLP

Introduction to NLP: Origins and Challenges, Language and Grammar in NLP, Regular Expressions and Finite-State Automata, Tokenization: Text Segmentation and Sentence Splitting, Morphological Parsing: Stemming and Lemmatization, Spelling Error Detection and Correction, Minimum Edit Distance and Applications, Statistical Language Models: Unigram, Bigram, and Trigram Models, Processing Indian Languages in NLP.

UNIT II: Word-Level and Syntactic Analysis

Introduction, Part-of-Speech (POS) Tagging: Rule-Based, Stochastic and Transformation-Based Approaches, Hidden Markov Models (HMM) and Maximum Entropy Models for POS Tagging, Context-Free Grammar (CFG) and Constituency Parsing, Treebanks and Normal Forms for Grammar, Top-Down and Bottom-Up Parsing Strategies, CYK Parsing Algorithm, Probabilistic Context-Free Grammars (PCFGs), Feature Structures and Unification

UNIT III: Text Classification and Information Retrieval

Naïve Bayes Classifier for Text Classification, Training and Optimization for Sentiment Analysis, Information Retrieval: Basic Concepts and Design Features, Information Retrieval Models: Classical, Non-Classical, and Alternative Models, Cluster Model, Fuzzy Model, and LSTM-Based Information, Retrieval, Word Sense Disambiguation (WSD) Methods: Supervised and Dictionary-Based Approaches.

UNIT IV: Machine Translation and Semantic Processing

Introduction to Machine Translation (MT), Language Divergence and Typology in MT Encoder- Decoder Model for Machine Translation, Translating in Low-Resource Scenarios, MT Evaluation Metrics and Techniques, Bias and Ethical Issues in NLP and Machine Translation, Semantic Analysis and First-Order Logic in NLP, Thematic Roles and Selectional Restrictions in Semantics, Word Senses and Relations Between Senses

UNIT V: Speech Processing and Advanced NLP Models

Speech Fundamentals: Phonetics and Acoustic Phonetics, Digital Signal Processing in Speech Analysis, Feature Extraction in Speech: Short-Time Fourier Transform (STFT), Mel-Frequency Cepstral Coefficients (MFCC) and Perceptual Linear Prediction (PLP), Hidden Markov Models (HMMs) in Speech Recognition.

Course Outcomes :

On successful completion of the course the student will be		POs related to Cos
CO1	Understand morphological processing and the structure of words and documents	
CO2	Analyze syntactic structures using various parsing algorithms.	
CO3	Apply semantic parsing techniques to interpret natural language text.	
CO4	Understand predicate-argument structures and meaning representation systems.	
CO5	Apply cross-lingual language models and speech recognition techniques in NLP applications	

Textbooks (Core Learning Materials)

1. Daniel Jurafsky & James H. Martin – Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Pearson Education, 2023.

2. Tanveer Siddiqui & U.S. Tiwary – Natural Language Processing and Information Retrieval, Oxford University Press.

Reference Books (Supplementary Learning)

1. T.V. Geetha – Understanding Natural Language Processing – Machine Learning and Deep Learning Perspectives, Pearson, 2024.

2. Akshay Kulkarni & Adarsha Shivananda – Natural Language Processing Recipes - Unlocking Text Data with Machine Learning and Deep Learning using Python, A press, 2019.

Web links and Video Lectures (e-Resources):

1. https://www.youtube.com/watch?v=M7SWr5xObkA
2. https://onlinecourses.nptel.ac.in/noc23_cs45/preview
3. https://archive.nptel.ac.in/courses/106/106/106106211/

Natural Language Processing

- Natural language processing is concerned with the developments of Computational models of aspects of Human language processing.
- Reasons:

1. To develop automated tools for language processing.
2. To gain a better understanding of human communication.

- Requirements :

1. How humans acquire knowledge
2. Store
3. Process Language
4. A knowledge of the world and of language.

- Two major Approaches

1. Rationalist
2. Empiricist

Natural Language Processing

- NLP is the branch of computer science focused on developing systems that allow computers to communicate with people using everyday language.
- NLP is an exciting and rapidly growing field that deals with the interaction between computers and human language.
- In this course, you will learn about the techniques and algorithms used to analyze and understand human language, and you will have the opportunity to apply these techniques to real-world problems.

What is Natural Language Processing (NLP)?

- Natural language processing is the set of methods for making human language accessible to computers

(Jacob Eisenstein)

- Natural language processing is the field at the intersection of Computer science (Artificial intelligence) and linguistics

(Christopher Manning)

- Make computers to understand natural language to do certain task humans can do such as Machine translation, Summarization, Questions answering

(Behrooz Mansouri)

Related Areas

- Artificial Intelligence
- Formal Language (Automata) Theory
- Machine Learning
- Deep Learning
- Data Science
- Linguistics
- Psycholinguistics
- Cognitive Science
- Philosophy of Language

- "Natural language is the most important part of artificial intelligence."

  John Searle

- "Natural language processing is a cornerstone of artificial intelligence, allowing computers to read and understand human language, as well as to produce and recognize speech."

  Ginni Rometty

- "Natural language processing is one of the most important fields in artificial intelligence and also one of the most difficult."

  Dan Jurafsky

Natural Language Processing: Terms

- Natural language refers to the language that humans use to communicate with each other, such as English, Spanish, or Chinese
- Processing As distinguished from data processing
- Question: How is data processing and natural language processing different?
- Consider the Unix wc program, which counts the total number of bytes, words, and lines in a text file
- When used to count bytes and lines, wc is an ordinary data processing application
- However, when it is used to count the words in a file, it requires knowledge about what it means to be a word and thus becomes a language processing system

Applications

- Machine Translation
- Information Retrieval
- Question Answering
- Dialogue Systems
- Information Extraction
- Entity Linking
- Text Summarization
- Text Classification
- Sentiment Analysis
- Opinion mining

Sentiment analysis

- **Sentiment analysis**, also referred to as **opinion mining**, is an approach to natural language processing (NLP) that identifies the emotional tone behind a body of text.
- This is a popular way for organizations to determine and categorize opinions about a product, service or idea.
- Sentiment analysis systems help organizations gather insights into real-time customer sentiment, customer experience and brand reputation.
- Generally, these tools use text analytics to analyze online sources such as emails, blog posts, online reviews, news articles, survey responses, case studies, web chats, tweets, forums and comments.
- Sentiment analysis uses machine learning models to perform text analysis of human language. The metrics used are designed to detect whether the overall sentiment of a piece of text is positive, negative or neutral.

Machine Translation

- **Machine translation**, sometimes referred to by the abbreviation **MT**, is a sub-field of computational linguistics that investigates the use of software to translate text or speech from one language to another.
- On a basic level, MT performs mechanical substitution of words in one language for words in another, but that alone rarely produces a good translation because recognition of whole phrases and their closest counterparts in the target language is needed.
- Not all words in one language have equivalent words in another language, and many words have more than one meaning.
- Solving this problem with corpus statistical and neural techniques is a rapidly growing field that is leading to better translations, handling differences in linguistic typology, translation of idioms, and the isolation of anomalies.

Corpus: A collection of written texts, especially the entire works of a particular author.

Text Extraction

- There are a number of natural language processing techniques that can be used to extract information from text or unstructured data.
- These techniques can be used to extract information such as entity names, locations, quantities, and more.
- With the help of natural language processing, computers can make sense of the vast amount of unstructured text data that is generated every day, and humans can reap the benefits of having this information readily available.
- Industries such as healthcare, finance, and e-commerce are already using natural language processing techniques to extract information and improve business processes.
- As the machine learning technology continues to develop, we will only see more and more information extraction use cases covered.

Text Classification

- Unstructured text is everywhere, such as emails, chat conversations, websites, and social media. Nevertheless, it's hard to extract value from this data unless it's organized in a certain way
- Text classification also known as *text tagging* or *text categorization* is the process of categorizing text into organized groups. By using Natural Language Processing (NLP), text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content
- Text classification is becoming an increasingly important part of businesses as it allows to easily get insights from data and automate business processes.

Speech Recognition

- Speech recognition is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers.
- It is also known as automatic speech recognition (ASR), computer speech recognition or speech to text (STT)
- It incorporates knowledge and research in the computer science, linguistics and computer engineering fields. The reverse process is speech synthesis.

Chatbot

- Chatbots are computer programs that conduct automatic conversations with people. They are mainly used in customer service for information acquisition. As the name implies, these are bots designed with the purpose of chatting and are also simply referred to as “bots.”
- You’ll come across chatbots on business websites or messengers that give pre-scripted replies to your questions. As the entire process is automated, bots can provide quick assistance 24/7 without human intervention.

• Email Filter

- One of the most fundamental and essential applications of NLP online is email filtering. It began with spam filters, which identified specific words or phrases that indicate a spam message. But, like early NLP adaptations, filtering has been improved.
- Gmail's email categorization is one of the more common, newer implementations of NLP. Based on the contents of emails, the algorithm determines whether they belong in one of three categories (main, social, or promotional).
- This maintains your inbox manageable for all Gmail users, with critical, relevant emails you want to see and reply to fast.

Search Autocorrect and Autocomplete

- When you type 2-3 letters into Google to search for anything, it displays a list of probable search keywords. Alternatively, if you search for anything with mistakes, it corrects them for you while still returning relevant results. Isn't it incredible?
- Everyone uses Google search autocorrect autocomplete on a regular basis but seldom gives it any thought. It's a fantastic illustration of how natural language processing is touching millions of people across the world, including you and me.
- Both, search autocomplete and autocorrect make it much easier to locate accurate results

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 analyze 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.
- The NLU is harder than NLG

Core Technologies

- Language modeling
- Part-of-speech tagging
- Syntactic parsing
- Coreference resolution
- Named-entity recognition
- Word sense disambiguation
- Semantic Role Labeling

Origins and Challenges

- NLP is mistakenly called NLU
- NLU involves only Interpretation where as NLP involves Interpretation (Understanding) and Generation (production).
- NLP includes Speech Processing.
- Computational models are divided into two types.

1. Knowledge driven : the system relay on explicitly coded linguistic knowledge represented as set of rules called grammar.

2. Data driven : presume the existence of large amount of data is used to learn syntactic patterns using ML techniques. The amount of human effort is less and the quality dependents on the quality of data.

- In this course we balance between knowledge (semantic)driven and data driven in one hand and theory and practical on the other hand.
- The amount of information available in www is abundance so we study Information retrieval using NLP.

- Four eras of NLP
- 1940–1969
- Early Explorations
- 1970–1992
- Hand-built demonstration NLP systems,
- of increasing formalization
- 1993–2012
- Statistical or Probabilistic NLP and then
- more general Supervised ML for NLP
- 2013–now
- Deep Learning or Artificial Neural
- Networks for NLP. Unsupervised or Self-
- Supervised NLP. Reinforcement Learning

Early Explorations-1940–1969

- **Machine Translation: Just a Code?**

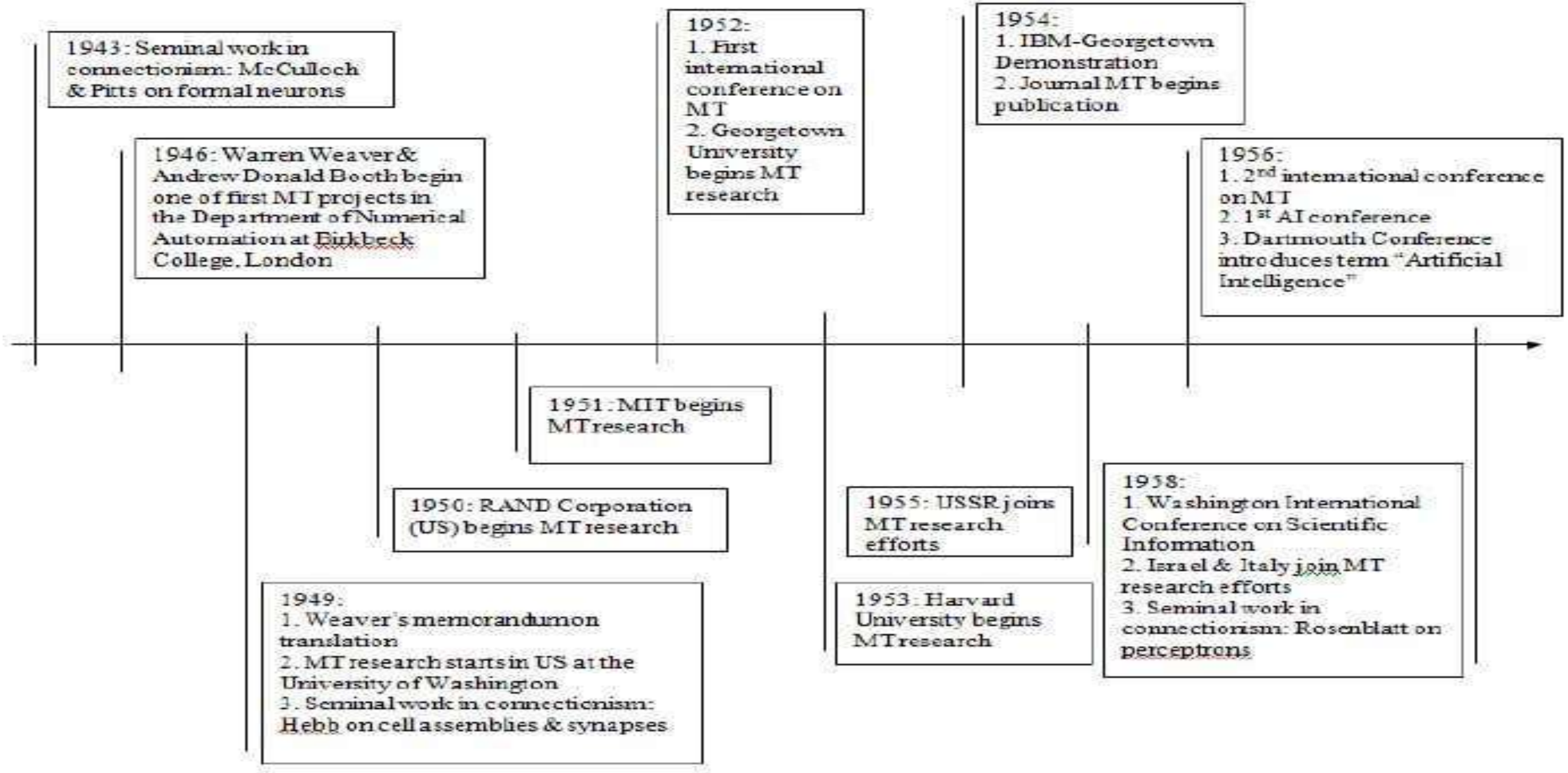
"Also knowing nothing official about, but having guessed and inferred considerable about, the powerful new mechanized methods in cryptography—methods which I believe succeed even when one does not know what

language has been coded—one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has

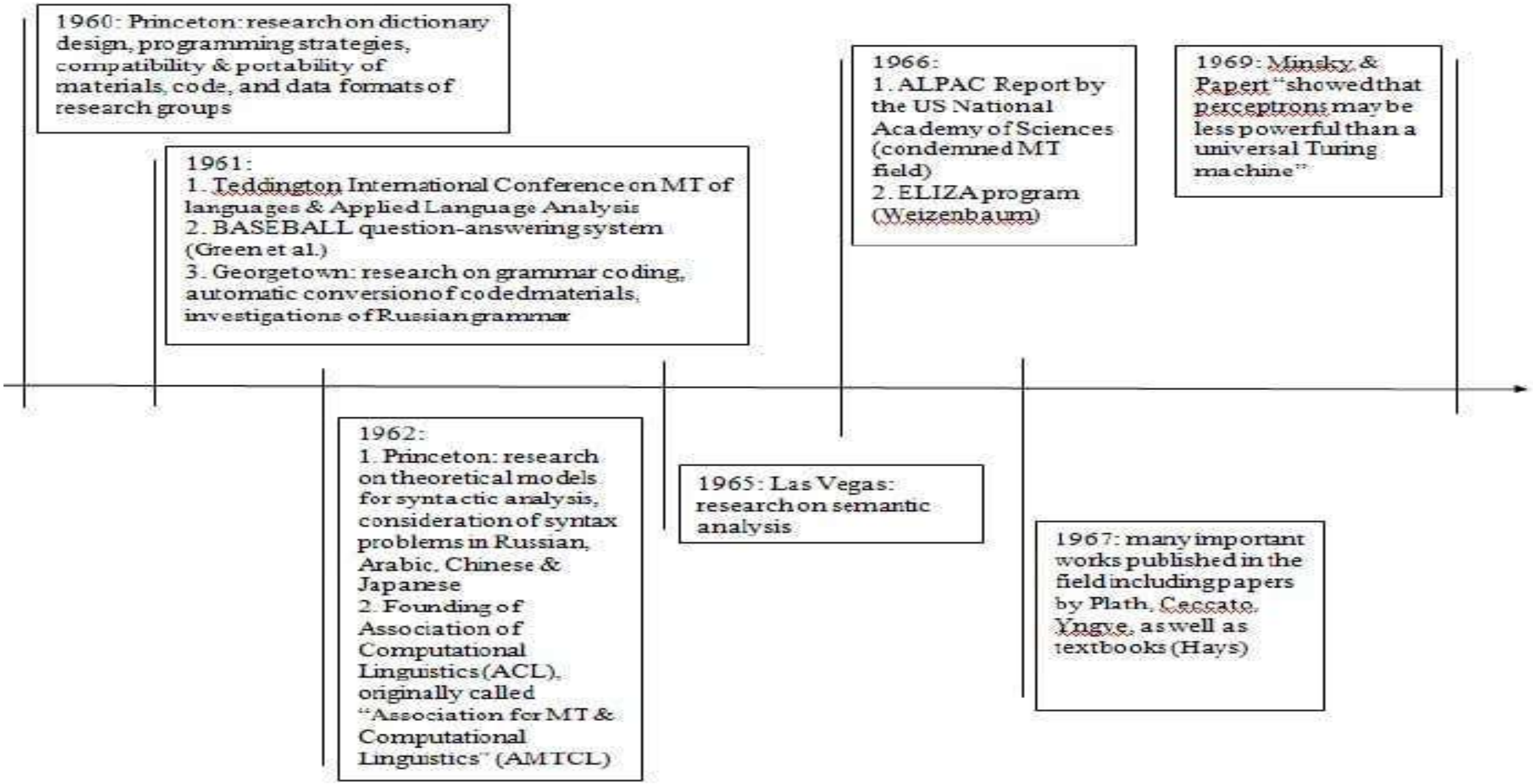
been coded in some strange symbols. I will now proceed to decode.' "

- – Warren Weaver (1955:18, quoting a letter he wrote in 1947)

Machine Translation: The origin of NLP/Computational Linguistics

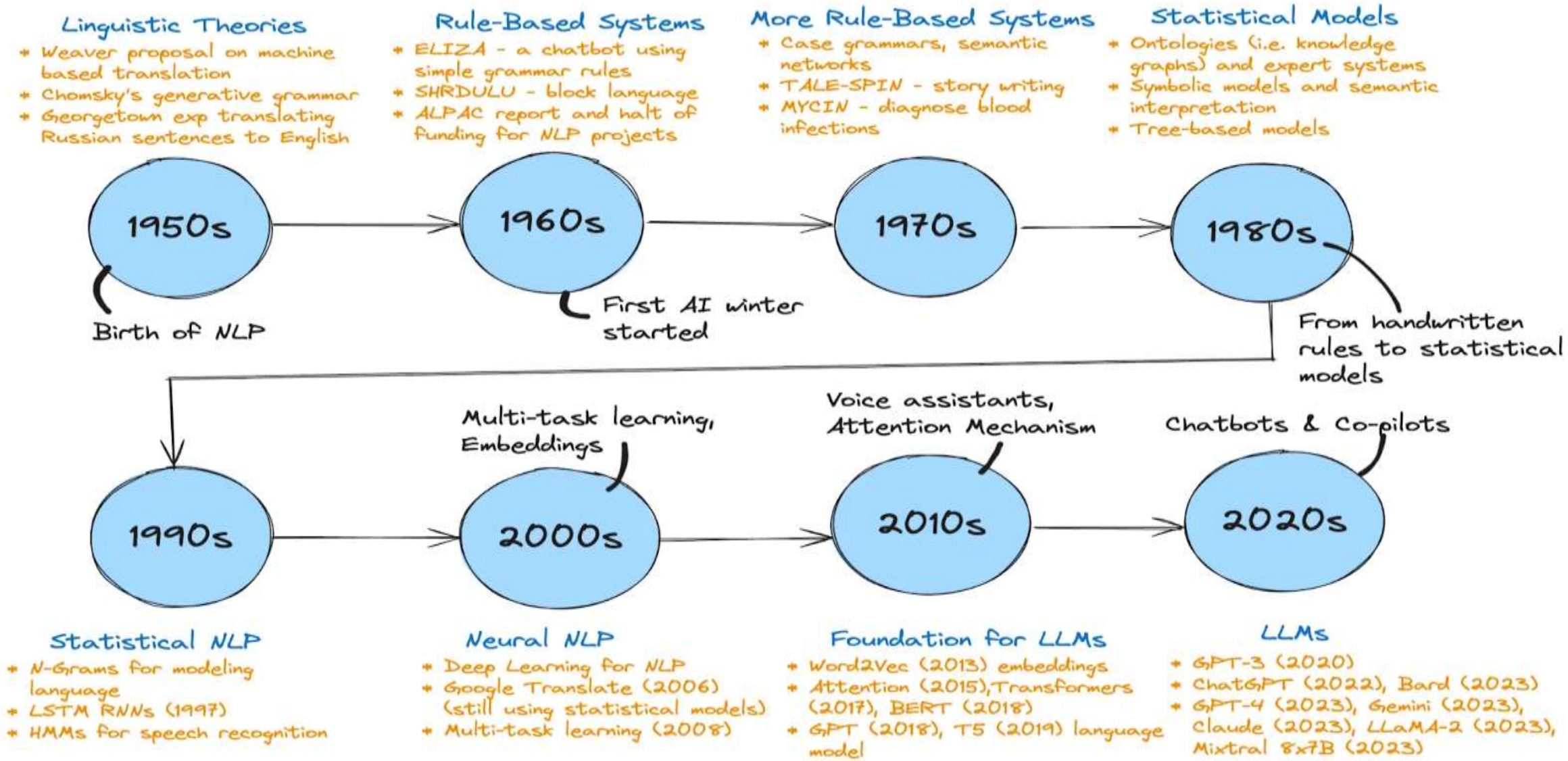


NLP in the 1960s



Deep Learning or Artificial Neural Networks for NLP 2013–present

History of NLP



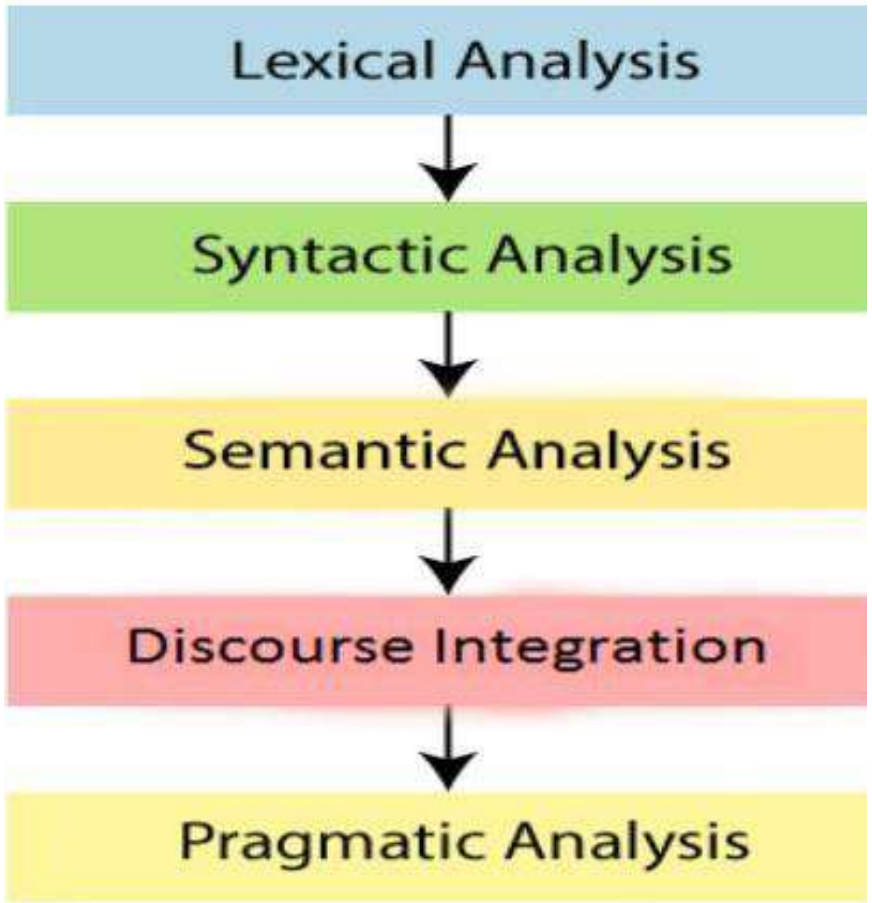
Language and Knowledge

- Language is medium of expression in which knowledge is deciphered.
- Lang. is the outer form of content it express
- To process a language means to process the content of it.
- As computers are not able to understand natural language, methods are developed to map its content in a formal language.
- Language is taken up as knowledge representation
- The lang. and speech community considers language as set of sounds however here we are representing and processing only text.
- Language processing has different levels, each involves different types of knowledge.

Steps in NLP

There are general five steps :

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



- The simplest level of analysis is Lexical analysis, which involves analysis of words.
- Words are most fundamental unit (syntactic and semantic) of any natural language text
- Word level processing require morphological knowledge i.e knowledge about structure and formation of words from basic units
- The next level of analysis is Syntactic analysis, which considers a sequence of words as unit i.e. A sentence and finds its structure.
- Syntactic analysis decomposes a sentence into words and identify how they relate each other.
- This level of processing requires syntactic knowledge, i.e knowledge about how words are combined to form larger units such as phrases and sentences and what constraints are imposed on them.
- Example: I went to the market is a valid sentence whereas went I to the market is not . She is going to the market is valid whereas she are going to the market is not.
- This require detailed knowledge about the rules of Grammar.

- The third level of analysis is Semantic analysis. Semantic associated with meaning of the language.
- Semantic analysis is concerned with creating meaning representation of linguistic input.
- The idea of Semantic interpretation is take some NL sentences and map them onto some representation of meaning.
- Grammatically valid sentences can be meaningless.
- Example colorless green ideas sleep furiously (Chomsky 1957) the sentence is well-formed i.e Syntactically correct but semantically anomalous
- Syntax is not only the component of meaning
- We feel that humans apply all sort of knowledge to arrive at the meaning of the sentence.(1,2,3,4,5)
- The semantic analyzer disregards sentence such as “hot ice-cream”.
- Another Example is “Manhattan calls out to Dave” passes a syntactic analysis because it’s a grammatically correct sentence. However, it fails a semantic analysis. Because Manhattan is a place (and can’t literally call out to people), the sentence’s meaning doesn’t make sense.

- The fourth level of analysis is Discourse analysis.
- Discourse level processing attempts to interpret the structure and meaning of even larger units example at paragraph level and document level the terms of words , phrases, clusters and sentences.
- It requires discourse knowledge i.e knowledge of how the meaning of a sentence is determined by preceding sentences.
- For instance, if one sentence reads, “Manhattan speaks to all its people,” and the following sentence reads, “It calls out to Dave,” discourse integration checks the first sentence for context to understand that “It” in the latter sentence refers to Manhattan.
- The highest level of processing is Pragmatic analysis. Which deals with the purposeful use of sentences in situations.
- It requires the knowledge of world i.e. knowledge that extends beyond the contents of the text.
- For instance, a pragmatic analysis can uncover the intended meaning of “Manhattan speaks to all its people.” Methods like neural networks assess the context to understand that the sentence isn’t literal, and most people won’t interpret it as such. A pragmatic analysis deduces that this sentence is a metaphor for how people emotionally connect with place.

Challenges of NLP

- The problem of representation and interpretation.
- Natural languages are highly ambiguous and vague achieving such representation is difficult.
- The inability to capture all required knowledge is another difficulty.
- It is almost impossible to embody all sources of knowledge that humans use to process language.
- Identifying its semantic- context in which word is used.
- Quantifier scoping is another problem. The scope of quantifier is not clear (i.e the, each, etc.)
- The ambiguity of natural language is difficult.
- Word level ambiguity- a word may be ambiguous in part of speech or ambiguous in meaning

- Sentence may be ambiguous even if the word are not.
- Structural ambiguity ex: stolen refile found by tree
- None of the words in this sentence is ambiguous but the sentence is.
- This is an example of structural ambiguity.
- A number of grammars have been proposed to describe the structure of sentences.

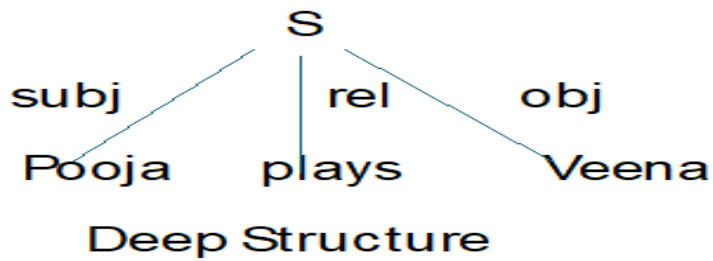
Language and Grammar

- Automatic processing of language requires the rules and exceptions of a lang. to be explained to the computer.
- Grammar defines a Language.
- It consists of a set of rules that allows us to parse and generate sentences in a language.
- Transformational grammar, Phrase structure grammar (Chomsky 1957)
- Lexical functional grammar(1982)
- Dependency grammar, Paninian grammar, tree adjoining grammar(Joshi1985)
- Phrase structure grammar focuses on derivation
- Dependency grammar, Paninian grammar, Lexical functional grammar focuses on relationships.
- Noam Chomsky proposed a hierarchy of formal grammars based on the level of complexity.
- The generative grammar refers to general frame work introduced by Chomsky

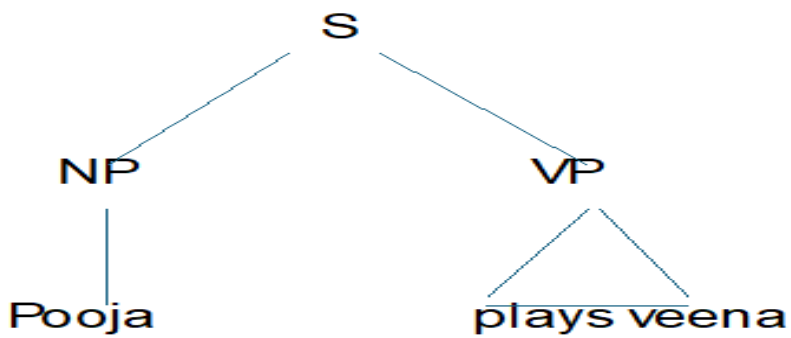
- Generative grammar refers to any grammar that uses a set of rules to specify or generate all and only grammatical(well-formed) sentences in a language.
- Chomsky proposed Syntactic structures that each sentence in a language has two levels of representation

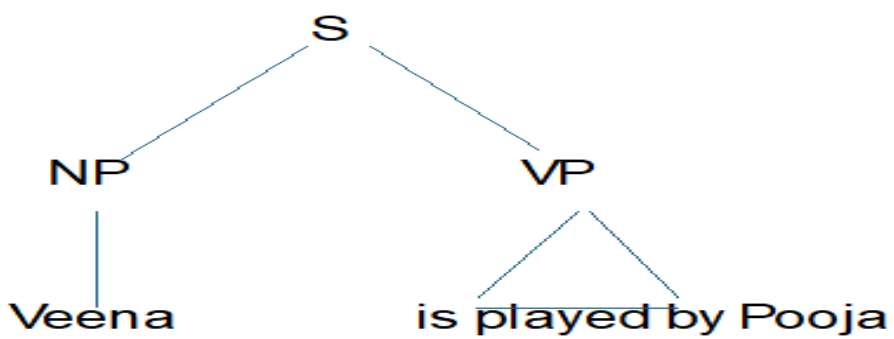
1. Deep structure
2. Surface structure

- The mapping from deep structure to surface structure is carried out by transformations.
- Deep structure can be transformed into many different level surface level repsreentations.
- Sentences with different surface level representations have the same meaning
- Example; Pooja plays veena
- Veena is played by Pooja have the same meaning
- Transformational grammar has three components
- 1. Phrase structure Grammar
- 2. Transformational Rules
- 3.Morphophonemics rules



Deep Structure





Surface Structure

- Phrase structure grammar consists of rules that generate natural language sentences.
- Example: S-> NP+VP

  VP -> V+ NP

  NP -> Det + Noun

  V -> Aux + Verb

  Det -> the, a, an,---

  Verb -> catch, eat, write, ---

  Noun -> Police, teacher, ----

  Aux -> will, can, is, ----

Transformational grammar is a set of Transformation rules , which transform one phrase-maker(underlying) into another phrase-maker (derived)

These rules are applied on the terminal string generated by phrase structure rules.

These rules are used to transform one surface representation into another.

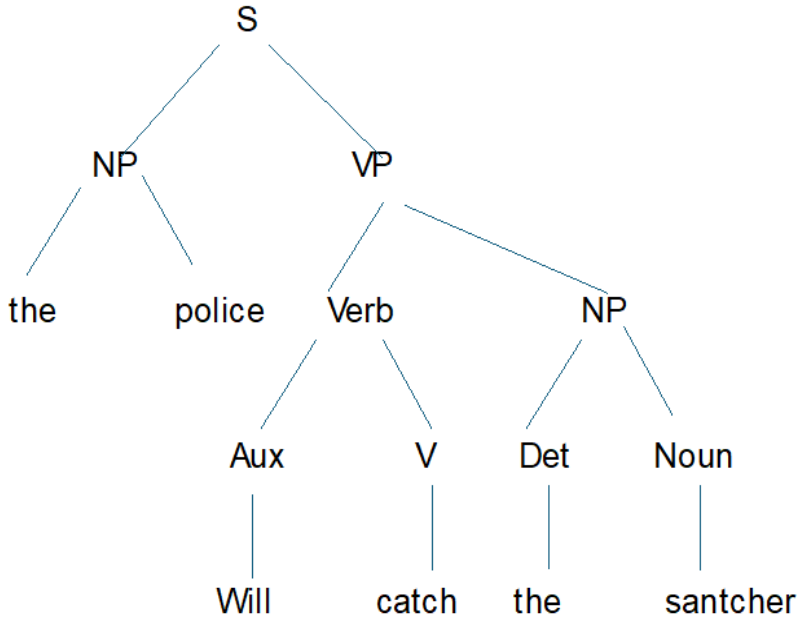
Example: An active sentence into Passive sentence

The rule relating active and passive sentences are given by Chomsky is

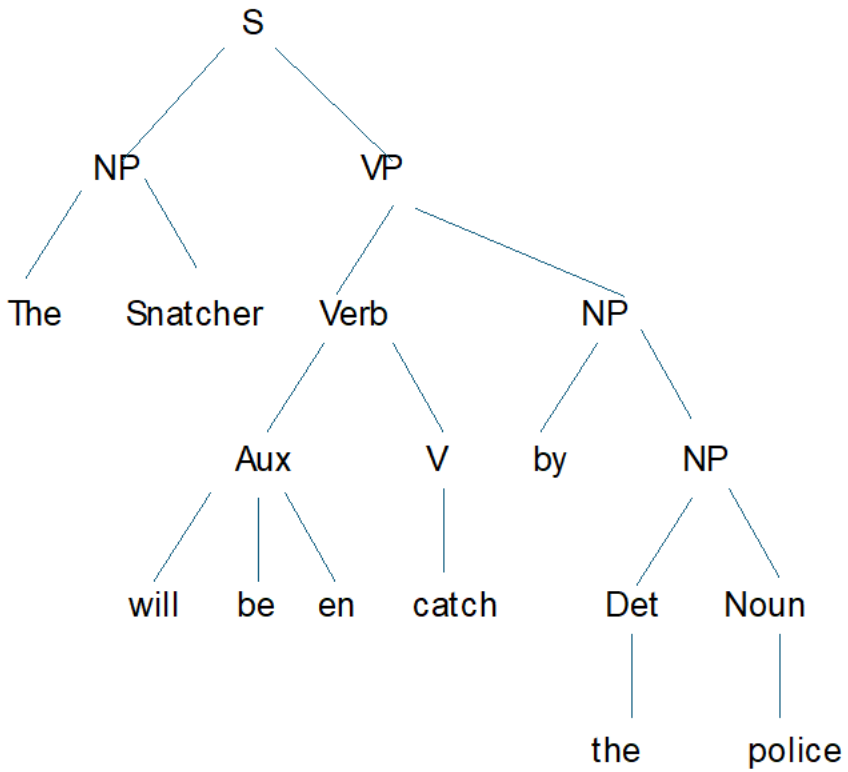
NP1-Aux-V-NP2 -> NP2-Aux+be+en-V-by-NP1

This rule says that NP1-Aux-V-NP2 is replaced by NP2-Aux+be+en-V-by-NP1

- Consider the active sentence “Police will catch the snatcher”
- The passive transformation rule will convert the sentence into:
- The + culprit+ will + be+ en + catch + by+ police



Parse Structure of Sentence



Structure of Sentence after applying passive Transformations

Basic Language Processing Tasks, Tools and Algorithms

- Basic Text Processing: Tokenization, Stemming, Spelling Correction
- Language Modeling: N-grams, smoothing
- Morphology, Parts of Speech Tagging
- Syntax: PCFGs, Dependency Parsing
- Lexical Semantics, Word Sense Disambiguation
- Distributional Semantics, Word Embeddings
- Topic Models

Processing Indian Languages

- There are number of differences between Indian languages and English.
- Unlike English, Indic scripts have non-linear structure
- Unlike English, Indian languages have SOV(subject-object-verb) as the default sentence structure.

Why study NLP?

- Text is the largest repository of human knowledge
- news articles, web pages, scientific articles, patents, emails, government documents ....
- Tweets, Facebook posts, comments, Quora ...
- Fundamental and Scientific Goal
- Deep understanding of broad language
- Engineering Goal
- Design, implement, and test systems that process natural languages for practical applications

Why is NLP hard?

Lexical Ambiguity

- *Will Will will Will's will?*
- *Rose rose to put rose roes on her rows of roses.*
- *Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo.*
  $\longrightarrow$ Buffaloes from Buffalo, NY, whom buffaloes from Buffalo bully, bully buffaloes from Buffalo.

Why is NLP hard?

Language ambiguity: Structural

- *The man saw the boy with the binoculars.*
- *Flying planes can be dangerous.*
- *Hole found in the room wall; police are looking into it.*

Language imprecision and vagueness

- *It is very warm here.*
- *Q: Did your mother call your aunt last night?*
  A: I'm sure she must have.

Why is the teacher wearing sun-glasses?

...

Because the class is so **bright**.

Ambiguities

News Headlines

- Hospitals Are Sued by 7 Foot Doctors
- Stolen Painting Found by Tree
- Teacher Strikes Idle Kids

Ambiguity is pervasive

- Find at least 5 meanings of this sentence:
  - I made her duck
- I cooked duck for her
- I cooked duck belonging to her
- I created the (artificial) duck, she owns
- I caused her to quickly lower her head or body
- I waved my magic wand and turned her into a duck

Ambiguity is pervasive

Syntactic Category

- 'Duck' can be a noun or verb
- 'her' can be a possessive ('of her') or dative ('for her') pronoun

Word Meaning

- 'make' can mean 'create' or 'cook'

Ambiguity is pervasive

Grammar

make can be

- **Transitive:** (verb with a noun direct object)
- **Ditransitive:** (verb has 2 noun objects)
- **Action-transitive:** (verb has a direct object + verb)

Phonetics

- I'm eight or duck
- I'm aid her duck

- I saw the man with the telescope. **2 parses**
- I saw the man on the hill with the telescope. **5 parses**
- I saw the man on the hill in Texas with the telescope. **14 parses**

Ambiguity is Explosive

- I saw the man with the telescope. **2 parses**
- I saw the man on the hill with the telescope. **5 parses**
- I saw the man on the hill in Texas with the telescope. **14 parses**
- I saw the man on the hill in Texas with the telescope at noon. **42 parses**

Ambiguity is Explosive

- I saw the man with the telescope. **2 parses**
- I saw the man on the hill with the telescope. **5 parses**
- I saw the man on the hill in Texas with the telescope. **14 parses**
- I saw the man on the hill in Texas with the telescope at noon. **42 parses**
- I saw the man on the hill in Texas with the telescope at noon on Monday. **132 parses**

- The goal in the production and comprehension of natural language is *efficient* communication.

Why Is Language Ambiguous?

The goal in the production and comprehension of natural language is *efficient* communication.

Allowing resolvable ambiguity

- permits shorter linguistic expressions
- avoids language being overly complex

Language relies on people's ability to use their knowledge and inference abilities to properly resolve ambiguities

Why else is NLP hard?

Why is NLP hard?

Why else is NLP hard?

Non-standard English

Great job @justinbieber! Were SOO PROUD of what youve accomplished! U taught us 2 #neversaynever & you yourself should never give up either

Segmentation Issues

the New York-New Haven Railroad

Why else is NLP hard?

Non-standard English

Great job @justinbieber! Were SOO PROUD of what youve accomplished! U taught us 2 #neversaynever & you yourself should never give up either

Segmentation Issues

the New York-New Haven Railroad

the [New] [York-New] [Haven] [Railroad]

Why else is NLP hard?

Idioms

- dark horse
- Ball in your court
- Burn the midnight oil

neologisms

- unfriend
- retweet
- Google/Skype/photoshop

Why is NLP hard?

New Senses of a word

- That's *sick* dude!
- Giants ... *multinationals, conglomerates, manufacturers*

Tricky Entity Names

- Where is *A Bug's Life* playing ...
- *Let It Be* was recorded ...

What we do in NLP?

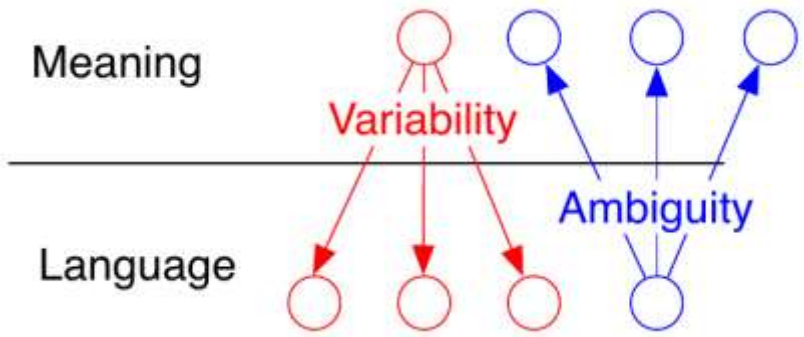
Tools Required

- Knowledge about language
- Knowledge about the world
- A way to combine knowledge resources

How is it generally done?

- Probabilistic models built from language data
  - P(“maison” $\longrightarrow$ “house”) is high
  - P(I saw a van) > P(eyes awe of an)
- Extracting rough text features does half the job.

Why is NLP hard?



Variability:

He drew the house
He made a sketch of the house
He showed me his drawing of the house
He portrayed the house in his paintings
He drafted the house in his sketchbook

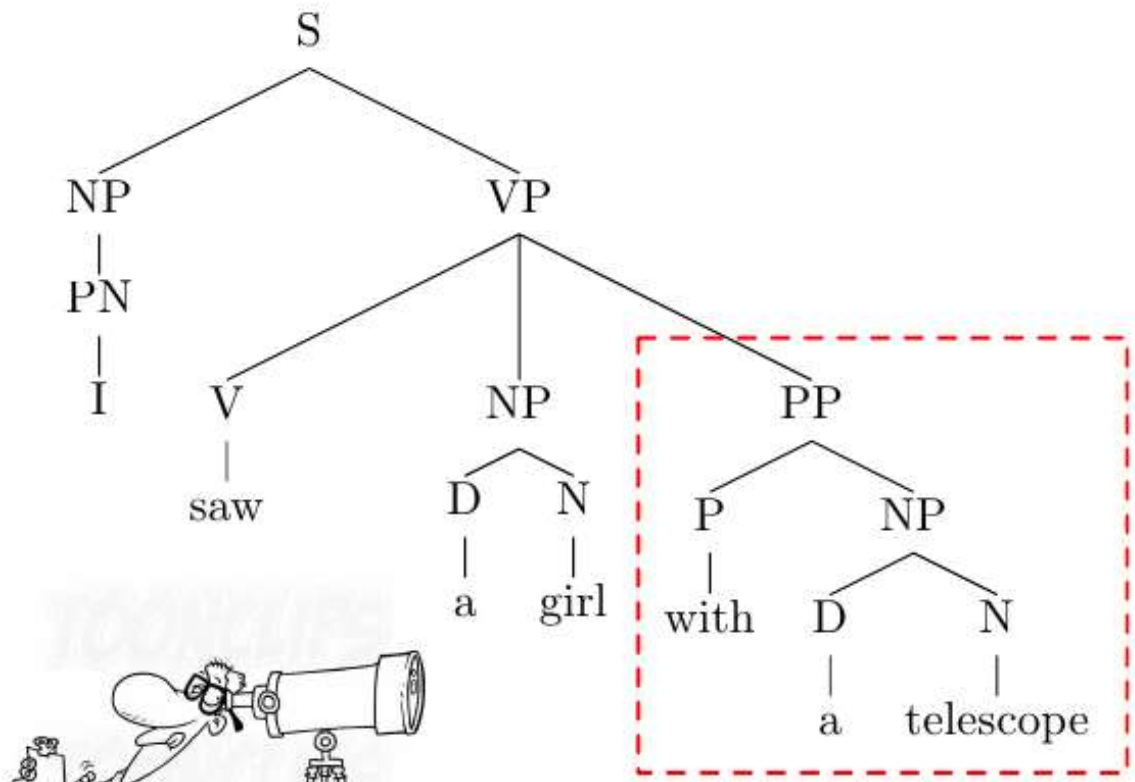
Ambiguity:

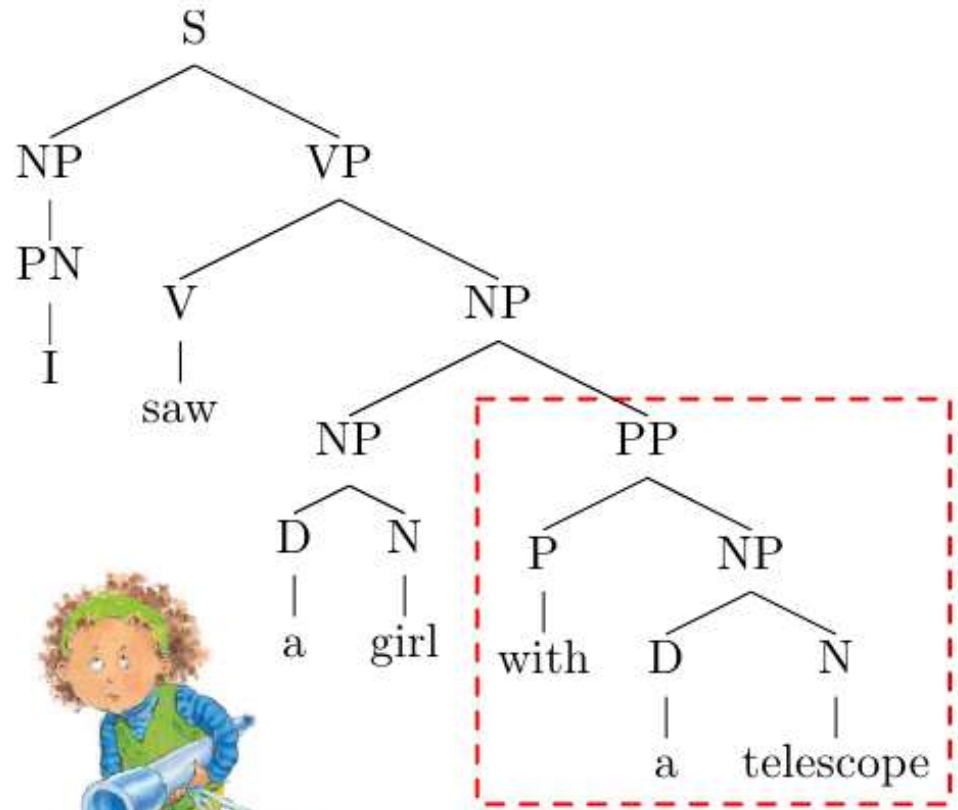
She drew a picture of herself ~ sketched, made a drawing of
cart drawn by two horses... ~ pulled
He drew crowds wherever he went ... ~ attracted
The driver slowed as he drew even with me ~ proceeded
The officer drew a gun and pointed it at ... ~ took out, produced

Ambiguity at many levels

- Homophones: blew and blue
- Word senses: bank (finance or river?)
- Part of speech: chair (noun or verb?)
- Syntactic structure: I saw a girl with a telescope
- We'll look into this in more detail!
- Quantifier scope: Every child loves some movie
- Multiple: I saw her duck
- Reference: John dropped the goblet onto the glass table and it broke.
- Discourse: The meeting is canceled. Emily isn't coming to the office today.
- How can we model ambiguity, and choose the correct analysis in context?

Syntactic Ambiguity: Prepositional Phrase Attachment





Text Processing: Basics

Tokenization

- Tokenization is the process of segmenting a string of characters into words.
- Word tokenization in NLP is the process of breaking down a text into individual words, called tokens
- Tokenization is the process of dividing a text into smaller units (tokens)
- Specifically, word tokenization involves splitting the text into individual words.
- **Example**
- In the sentence "This is an example.", word tokenization would result in the tokens: "This", "is", "an", "example", and ".“
- **Purpose**
- Tokenization helps computers understand and process human language by breaking down text into manageable units.
- Depending on the application in hand, you might have to perform sentence segmentation as well.

- *“Dr. Watson, Mr. Devi Priya”, said Stamford, introducing us.*
- can be tokenized as in the following example, where each token is enclosed in single quotation marks:
- `'" `Dr.' `Watson' `,' `Mr.’ `Devi’ `Priya' `'" `,’ `said' `Stamford' `,' `introducing' `us' `.’
- **Purpose**
- The primary goal is to segment text into meaningful units, which are often individual words, to facilitate further analysis such as text classification, sentiment analysis, machine translation, or information retrieval.
- **Process**
- It typically involves identifying word boundaries within a text, often based on whitespace, punctuation, or other defined delimiters.

- Tokenization is the task of chopping text into pieces, called tokens
- Token is an instance of a sequence of characters in some particular document (grouped as a useful semantic unit)
- Type is the class of all tokens containing the same character sequence
- Term is a (perhaps normalized) type that is included in the corpus dictionary
- Example: to sleep more to learn
- Token: to, sleep, more, to, learn
- Type: to, sleep, more, learn
- Term: sleep, more, learn (stop words removed)
- **In the sentence, "In Kolkata I took my hat off. But I can't put it back on.", total number of word tokens and word types are:**
- 1. 14, 13
- 2. 13, 14
- 3. 15, 14
- 4. 14, 15
- **Answer:** 1. 14, 13.
- **Solution:** Here, the word "I" is repeated two times so type count is one less than token count.

Text Normalization

- Every NLP task requires text normalization:
  1. Tokenizing (segmenting) words
  2. Normalizing word formats
  3. Segmenting sentences

Space-based tokenization

- A very simple way to tokenize
  - For languages that use space characters between words
    - Arabic, Cyrillic, Greek, Latin, etc., based writing systems
  - Segment off a token between instances of spaces
- Unix tools for space-based tokenization
  - The "tr" command
  - Inspired by Ken Church's UNIX for Poets
  - Given a text file, output the word tokens and their frequencies

Word-based Tokenization

- Approach
- Splitting the text by spaces
- Other delimiters such as punctuation can be used
- Advantages
- Easy to implement
- Disadvantages
- High risk of missing words; e.g., Let and Let's will have two different types
- Languages like Chinese do not have space
- Huge vocabulary size (token type)
- o Limit the number of words that can be added to the vocabulary

Simple Tokenization in UNIX

- (Inspired by Ken Church's UNIX for Poets.)
- Given a text file, output the word tokens and their frequencies

```
tr -sc 'A-Za-z' '\n' < shakes.txt
      | sort
      | uniq -c
```

Change all non-alpha to newlines

Sort in alphabetical order

Merge and count each type

```
1945 A
  72 AARON
  19 ABBESS
   5 ABBOT
 ... ...
```

```
25 Aaron
 6 Abate
 1 Abates
 5 Abbess
 6 Abbey
 3 Abbot
....   …
```

The first step: tokenizing

```
tr -sc 'A-Za-z' '\n' < shakes.txt | head
THE
SONNETS
by
William
Shakespeare
From
fairest
creatures
We
...
```

The second step: sorting

```
tr -sc 'A-Za-z' '\n' < shakes.txt | sort | head
```

```
A
A
A
A
A
A
A
A
A
...
```

More counting

- Merging upper and lower case

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c
```

- Sorting the counts

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c | sort -n -r
```

```
23243 the
22225 i
18618 and
16339 to
15687 of
12780 a
12163 you
10839 my
10005 in
8954  d
```

What happened here?

Issues in Tokenization

- Can't just blindly remove punctuation:
  - m.p.h., Ph.D., AT&T, cap'n
  - prices ($45.55)
  - dates (01/02/06)
  - URLs (http://www.stanford.edu)
  - hashtags (#nlproc)
  - email addresses (someone@cs.colorado.edu)
- Clitic: a word that doesn't stand on its own
  - "are" in we're, French "je" in j'ai, "le" in l'honneur
- When should multiword expressions (MWE) be words?
  - New York, rock 'n' roll

Tokenization in NLTK

Bird, Loper and Klein (2009), *Natural Language Processing with Python*. O'Reilly

```
>>> text = 'That U.S.A. poster-print costs $12.40...'
>>> pattern = r'''(?x)    # set flag to allow verbose regexps
...     ([A-Z]\.)+        # abbreviations, e.g. U.S.A.
...   | \w+(-\w+)*        # words with optional internal hyphens
...   | \$?\d+(\.\d+)?%?  # currency and percentages, e.g. $12.40, 82%
...   | \.\.\.            # ellipsis
...   | [][.,;"'?():-_`]  # these are separate tokens; includes ], [
... '''
>>> nltk.regexp_tokenize(text, pattern)
['That', 'U.S.A.', 'poster-print', 'costs', '$12.40', '...']
```

Tokenization in languages without spaces

Many languages (like Chinese, Japanese, Thai) don't use spaces to separate words!

How do we decide where the token boundaries should be?

Word tokenization in Chinese

Chinese words are composed of characters called **"hanzi"** (or sometimes just **"zi"**)

Each one represents a meaning unit called a morpheme.

Each word has on average 2.4 of them.

But deciding what counts as a word is complex and not agreed upon.

How to do word tokenization in Chinese?

- 姚明进入总决赛 “Yao Ming reaches the finals”

- 3 words?
- 姚明　进入　总决赛
- YaoMing　reaches　finals

- 5 words?

ls

How to do word tokenization in Chinese?

- 姚明进入总决赛 “Yao Ming reaches the finals”

- 3 words?
- 姚明　进入　总决赛
- YaoMing　reaches　finals

- 5 words?

ls

How to do word tokenization in Chinese?

- 姚明进入总决赛 “Yao Ming reaches the finals”

- 3 words?
- 姚明　进入　总决赛
- YaoMing　reaches　finals

- 5 words?
- 姚　明　进入　总　决赛
- Yao　Ming　reaches　overall　finals

- 7 characters? (don't use words at all):
- 姚　明　进　入　总　决　赛

How to do word tokenization in Chinese?

- 姚明进入总决赛 “Yao Ming reaches the finals”

- 3 words?
- 姚明　进入　总决赛
- YaoMing　reaches　finals

- 5 words?
- 姚　明　进入　总　决赛
- Yao　Ming　reaches　overall　finals

- 7 characters? (don't use words at all):
- 姚 明　进　入　总　决　赛
- Yao Ming enter enter overall decision game

Word tokenization / segmentation

So in Chinese it's common to just treat each character (zi) as a token.

- So the **segmentation** step is very simple

In other languages (like Thai and Japanese), more complex word segmentation is required.

- The standard algorithms are neural sequence models trained by supervised machine learning.

Another option for text tokenization

Instead of

- white-space segmentation
- single-character segmentation

Use the data to tell us how to tokenize.

Subword tokenization (because tokens can be parts of words as well as whole words)

Subword tokenization

- Three common algorithms:
  - **Byte-Pair Encoding (BPE)** (Sennrich et al., 2016)
  - **Unigram language modeling tokenization** (Kudo, 2018)
  - **WordPiece** (Schuster and Nakajima, 2012)
- All have 2 parts:
  - A token **learner** that takes a raw training corpus and induces a vocabulary (a set of tokens).
  - A token **segmenter** that takes a raw test sentence and tokenizes it according to that vocabulary

Sub-word Tokenization

Approach

- Frequently used words should not be split into smaller subwords
- Rare words should be decomposed into meaningful subwords
- Uses a special symbol to indicate which word is the start of the token and which word is the completion of the start of the token
- Tokenization → “Token”, “##ization”
- State-of-the-art approaches for NLP and IR rely on this type

Advantages

- Out-of-vocabulary word problem solved

Character-based Tokenization

Approach

- Splitting the text into individual characters

Advantages

- There will be no or very few unknown words (Out Of Vocabulary)
- Useful for languages that characters carry information
- Fewer number of tokens
- Easy to implement

Disadvantages

- A character usually does not have a meaning
- Cannot learn semantic for words
- Larger sequence to be processed by models
- More input to process

Sentence Segmentation

- The problem of deciding where the sentences begin and end.
- *Challenges Involved*
  - While '!', '?' are quite unambiguous
  - Period "." is quite ambiguous and can be used additionally for
    - Abbreviations (Dr., Mr., m.p.h.)
    - Numbers (2.4%, 4.3)
- *Approach: build a binary classifier*
- For each "."
  - Decides End Of Sentence/Not End Of Sentence
  - Classifiers can be: hand-written rules, regular expressions, or machine learning

Sentence Segmentation

!, ? mostly unambiguous but **period** “.” is very ambiguous

- Sentence boundary
- Abbreviations like Inc. or Dr.
- Numbers like .02% or 4.3

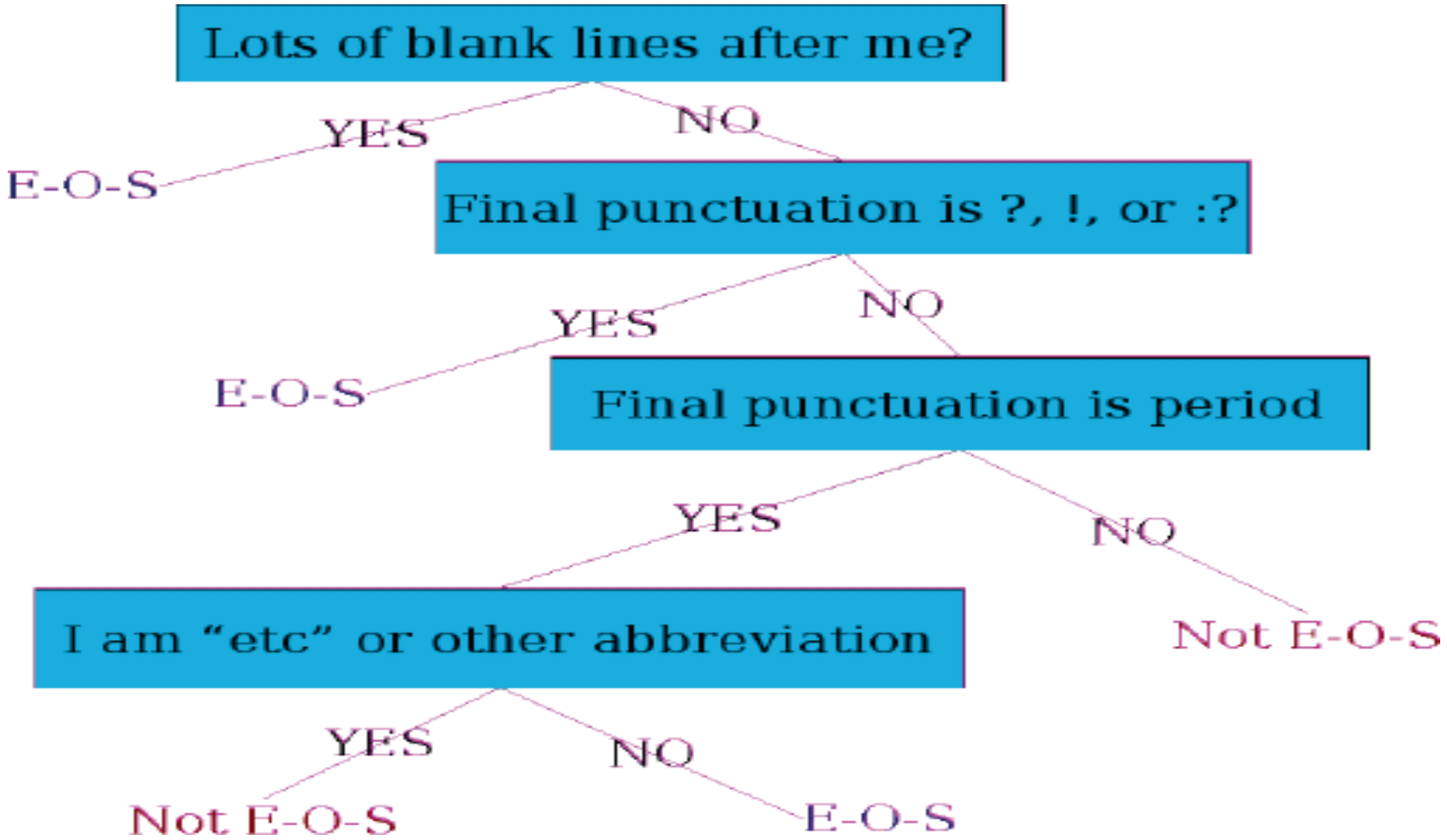
Common algorithm: Tokenize first: use rules or ML to classify a period as either (a) part of the word or (b) a sentence-boundary.

- An abbreviation dictionary can help

Sentence segmentation can then often be done by rules based on this tokenization.

Sentence Segmentation: Decision Tree Example

Decision Tree: Is this word the end-of-sentence (E-O-S)?



Implementing Decision Trees

- Just an if-then-else statement
- Choosing the features is more important
- For numeric features, thresholds are to be picked
- With increasing features including numerical ones, difficult to set up the structure by hand
- Decision Tree structure can be learned using machine learning over a training corpus

Basic Idea

Usually works top-down, by choosing a variable at each step that best splits the set of items.

Popular algorithms: ID3, C4.5, CART

Other Classifiers

The questions in the decision tree can be thought of as features, that could be exploited by any other classifier:

- Support Vector Machines
- Logistic regression
- Neural Networks

Byte-Pair Encoding (BPE) Tokenization

- Uses Huffman encoding for tokenization (greedy algorithm)
- Training Steps:

1. Starts with splitting the input words into single characters

    (each of them corresponds to a symbol in the final vocabulary)

    * In practice we commonly add special end of word symbol "__" before space

2. Find the most frequent occurring pair of symbols from the current vocabulary

3. Add this to the vocabulary and size of vocabulary increases by one

4. Repeat steps (2) and (3) till the defined number of tokens are built **or** no new combination of symbols exist with required frequency

Byte Pair Encoding (BPE) token learner

Let vocabulary be the set of all individual characters

= {A, B, C, D,…, a, b, c, d….}

- Repeat:
  - Choose the two symbols that are most frequently adjacent in the training corpus (say 'A', 'B')
  - Add a new merged symbol 'AB' to the vocabulary
  - Replace every adjacent 'A' 'B' in the corpus with 'AB'.
- Until *k* merges have been done.

BPE token learner algorithm

function BYTE-PAIR ENCODING(strings C, number of merges k) **returns** vocab V

$V \leftarrow$ all unique characters in C # initial set of tokens is characters
for $i = 1$ **to** k **do** # merge tokens til k times
  $t_L, t_R \leftarrow$ Most frequent pair of adjacent tokens in C
  $t_{NEW} \leftarrow t_L + t_R$ # make new token by concatenating
  $V \leftarrow V + t_{NEW}$ # update the vocabulary
  Replace each occurrence of t_L, t_R in C with t_{NEW} # and update the corpus
return V

Byte Pair Encoding (BPE) Addendum

Most subword algorithms are run inside space-separated tokens.

So we commonly first add a special end-of-word symbol '__' before space in training corpus

Next, separate into letters.

BPE token learner

Original (very fascinating☺) corpus:

low low low low low lowest lowest newer newer newer
newer newer newer wider wider wider new new

Add end-of-word tokens, resulting in this vocabulary:

vocabulary

_, d, e, i, l, n, o, r, s, t, w

BPE token learner

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w
2 l o w e s t _	
6 n e w e r _	
3 w i d e r _	
2 n e w _	

Merge e r to er

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er
2 l o w e s t _	
6 n e w er _	
3 w i d er _	
2 n e w _	

BPE

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er
2 l o w e s t _	
6 n e w er _	
3 w i d er _	
2 n e w _	

Merge er _ to er_

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_
2 l o w e s t _	
6 n e w er_	
3 w i d er_	
2 n e w _	

BPE

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_
2 l o w e s t _	
6 n e w er_	
3 w i d er_	
2 n e w _	

Merge n e to ne

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne
2 l o w e s t _	
6 ne w er_	
3 w i d er_	
2 ne w _	

BPE

The next merges are:

Merge	Current Vocabulary
(ne, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new
(l, o)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo
(lo, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low
(new, er_)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_
(low, _)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_, low_

BPE token **segmenter** algorithm

On the test data, run each merge learned from the training data:

- Greedily
- In the order we learned them
- (test frequencies don't play a role)

So: merge every e r to er, then merge er _ to er_, etc.

- Result:
  - Test set "n e w e r _" would be tokenized as a full word
  - Test set "l o w e r _" would be two tokens: "low er_"

Properties of BPE tokens

Usually include frequent words

And frequent subwords

- Which are often morphemes like *-est* or *–er*

A **morpheme** is the smallest meaning-bearing unit of a language

- *unlikeliest* has 3 morphemes *un-*, *likely*, and *-est*

Word Normalization

- Putting words/tokens in a standard format
  - U.S.A. or USA
  - uhhuh or uh-huh
  - Fed or fed
  - am, is, be, are

Case folding

- Applications like IR: reduce all letters to lower case
  - Since users tend to use lower case
  - Possible exception: upper case in mid-sentence?
    - e.g., ***General Motors***
    - ***Fed*** vs. ***fed***
    - ***SAIL*** vs. ***sail***
- For sentiment analysis, MT, Information extraction
  - Case is helpful (***US*** versus ***us*** is

Dealing with complex morphology is necessary for many languages

- e.g., the Turkish word:
- Uygarlastiramadiklarimizdanmissinizcasina
- `(behaving) as if you are among those whom we could not civilize'
- Uygar `civilized' + las `become'
  + tir `cause' + ama `not able'
  + dik `past' + lar 'plural'
  + imiz 'p1pl' + dan 'abl'
  + mis 'past' + siniz '2pl' + casina 'as if'

- **Training corpus:** low low low low low lowest lowest newer newer newer newer newer newer wider wider wider new new
- **Corpus**

5 l o w __

2 l o w e s t __

6 n e w e r __

3 w i d e r __

2 n e w __

- **Vocabulary**
- __, d, e, i, l, n, o, r, s, t, w

Challenges

- Word tokenization can be complex due to factors like:
- **Contractions:** Handling words like "don't" or "you're" (should they be one token or two?).
- **Punctuation:** Deciding whether punctuation marks should be treated as separate tokens or attached to words.
- **Compound words:** Dealing with words like "firefighter" or "high-tech" in languages where they might be written as single units or hyphenated.
- **Languages without clear word boundaries:** Some languages, like Chinese or Japanese, do not use spaces to separate words, requiring more sophisticated tokenization methods.

Methods

- Various techniques exist, ranging from simple whitespace-based splitting to more advanced methods like:
- **Regular expressions:** Using patterns to define token boundaries.
- **Rule-based systems:** Applying linguistic rules to identify tokens.
- **Statistical models:** Learning tokenization patterns from large text corpora.
- **Sub word tokenization:** Breaking down words into smaller units (subwords) to handle out-of-vocabulary words and manage vocabulary size, often used in modern NLP models like BERT and GPT.

Word Tokenization

What is Tokenization?

Tokenization is the process of segmenting a string of characters into words.

I have a can opener; but I can't open these cans.

Word Token

- An occurrence of a word
- For the above sentence, 11 word tokens.

Word Type

- A different realization of a word
- For the above sentence, 10 word types.

- NLTK Toolkit (Python)
- Stanford CoreNLP (Java)
- Unix Commands

Word Tokenization

Issues in Tokenization

- Finland's → Finland Finlands Finland's ?
- What're, I'm, shouldn't → What are, I am, should not ?
- San Francisco → one token or two?
- m.p.h. → ??

For information retrieval, use the same convention for documents and queries

Handling Hyphenation

Hyphens can be `Stopwords Removal`

End-of-Line Hyphen

Used for splitting whole words into part for text justification.
This paper describes MIMIC, an adaptive mixed initia-tive spoken dialogue system that provides movie show-time information.

Lexical Hyphen

Certain prefixes are offen written hyphenated, e.g. co-, pre-, meta-, multi-, etc.

Sententially Determined Hyphenation

Mainly to prevent incorrect parsing of the phrase. Some possible usages:

- Noun modified by an 'ed'-verb: *case-based, hand-delivered*
- Entire expression as a modifier in a noun group: *three-to-five-year direct marketing plan*

Lemmatization

- Reduce inflectional/variant forms to base form or
- Represent all words as their lemma, their shared root = dictionary headword form
- am, are, is → be
- car, cars, car's, cars' → car
- the boy's cars are different colors → the boy car be different color
  - He is reading detective stories → He be read detective story
- Lemmatization implies doing “proper” reduction to dictionary headword form
- e.g., WordNet is a lexical database of semantic relations between words in more than 200 languages
- Stemming is a process that extract stems by removing last few characters from a word, often leading to incorrect meanings and spelling
- Lemmatization considers the context and converts the word to its meaningful base form, which is called Lemma

Lemmatization is done by Morphological Parsing

- Morphology studies the internal structure of words, how words are built up from smaller meaningful units called **morphemes**
- *Morphemes are divided into two categories*
  - Stems: The core meaning bearing units
  - Affixes: Bits and pieces adhering to stems to change their meanings and grammatical functions or Parts that adhere to stems, often with grammatical functions
    - Prefix: un-, anti-, etc (a-, ati-, pra- etc.)
    - Suffix: -ity, -ation, etc (-taa, -ke, -ka etc.)
    - Infix: '*n*' in '*vindati*' (he knows), as contrasted with *vid* (to know).
- Morphological Parsers:
  - Parse *cats* into two morphemes *cat* and *s*
  -

Token Normalization

Why to "normalize"?

Indexed text and query terms must have the same form.

- U.S.A. and USA should be matched
- We implicitly define equivalence classes of terms

- Token/Term Normalization: Reducing tokens to canonical token (creating equivalence classes)
- Deleting periods to form a term
- U.S.A, USA
- How about C.A.T (Caterpillar Inc) and CAT? (Test Google on this!)
- Deleting hyphens to form a term
- anti-discriminatory, anti discriminatory
- Keeping relationships between unnormalized tokens
- car, automobile
- British vs. American spelling:
- color vs. colour

Stop words Removal

- Stopping: Removing common words from the stream of tokens that become index terms
- Words that are function words helping form sentence structure: the, of, and, to, ….
- For an application, an additional domain specific stop words list may be constructed
- Why do we need to remove stop words?
- Reduce indexing (or data) file size
- Usually has no impact on the NLP task's effectiveness, and may even improve it
- Can sometimes cause issues for NLP tasks:
- e.g., phrases: "to be or not to be", "let it be", "flights to Portland Maine"
- Some tasks consider very small stopwords list
- Sometimes perhaps only "the"
- List of stopwords: https://www.ranks.nl/stopwords

Stemming

- Stemming: To group words that are derived from a common stem
- Reduce terms to stems, chopping off affixes crudely
- e.g, “fish”, “fishes”, “fishing” could be mapped to “fish”
- Generally produces small improvements in tasks effectiveness
- Similar to stopping, stemming can be done aggressively, conservatively, or not at all
- Aggressively: consider “fish” and “fishing” the same
- Conservatively: just identifying plural forms using the letter “s”
- issues: ‘Centuries’ → ‘Centurie”
- Not at all: Consider all the word variants
- In different languages, stemming can have different importance for effectiveness:
- In Arabic, morphology is more complicated than English
- In Chinese, stemming is not effective

Evaluation of Stemmers

- There are three criteria for evaluating stemmers:

1. Correctness
2. Efficiency of the task
3. Compression performance

- There are two ways in which stemming can be incorrect:
- Over-stemming (too much of the term is removed)
- Two or more words being reduced to the same wrong root
- e.g., ‘centennial’, ‘century’, ‘center’: ‘cent’
- Under-stemming (too little of the term is removed)
- Two or more words could be wrongly reduced to more than one root word
- e.g., ‘acquire’, ‘acquiring’, ‘acquired’: acquir ‘acquisition’: ‘acquis’

Porter Stemmer (1980)

- The most common stemmer for English, introduced by Martin Porter A rule-based stemmer with rules for mostly suffix-stripping such as:
- “ing” → “-” connecting → connect
- “sses” → “ss” caresses → caress
- (m>0) “EED” → “EE” feed → feed agreed → agree
- m=measure of word or word part, when represented in form (VC+)
- V= vowel and C=consonants
- Advantage: It produces the best output as compared to other stemmers, and it has less error rate Disadvantage: Morphological variants produced are not always real words (produces stems)

- Based on a series of rewrite rules run in series
  - A cascade, in which output of each pass fed to next pass
- Some sample rules:

ATIONAL $\rightarrow$ ATE (e.g., relational $\rightarrow$ relate)

ING $\rightarrow$ ϵ if stem contains vowel (e.g., motoring $\rightarrow$ motor)

SSES $\rightarrow$ SS (e.g., grasses $\rightarrow$ grass)

Porter's algorithm

Step 1a

- sses → ss (caresses → caress)
- ies → i (ponies → poni)
- ss → ss (caress → caress)
- s → ϕ(cats → cat)

Step 1b

- (*v*)ing → ϕ(walking → walk, king → king)
- (*v*)ed → ϕ(played → play)

Porter's algorithm

Step 2

- ational → ate (relational → relate)
- izer → ize (digitizer → digitize)
- ator → ate (operator → operate)

Step 3

- al → ϕ(revival → reviv)
- able → ϕ(adjustable → adjust)
- ate → ϕ(activate → activ)

Statistical Language Model

M E Palanivel
SITAMS

Probabilistic Language Modeling

- **Goal:** Compute the probability of a sentence or sequence of words:

- $P(W) = P(w_1 w_2 w_3 \ldots w_n)$

- **Related Task:** probability of an upcoming word:

- $P(w_4 / w_1 w_2 w_3)$

- A model that computes either of these is called a **language model**

Computing P(W)

- S= The office is about fifteen minutes from my house
- How to compute the joint probability
- P(about, fifteen, minutes, from)
- Basic Idea Rely on the Chain Rule of Probability
- The Chain Rule
- Conditional Probabilities
- $P(B/A) = \frac{P(A,B)}{P(A)}$
- P(A,B) = P(B/A) P(A)
- P(A,B,C,D) = P(A)P(B/A)P(C/A,B)P(D/A,B,C)
- The Chain Rule in General
- $P(x_1,x_2,\ldots,x_n) = P(x_1)P(x_2/x_1)P(x_3/x_1,x_2)\ldots P(x_n/x_1,\ldots,x_{n-1})$

Probability of words in sentences

- $P(w_1 w_2 \ldots w_n) = \prod P(w_i / w_1 w_2 \ldots w_{i-1})$
- P("about fifteen minutes from") =

P(about) x P(fifteen | about) x P(minutes | about fifteen) x P(from | about fifteen minutes)

- P(office | about fifteen minutes from)

$$= \frac{\text{Count (about fifteen minutes from office)}}{\text{Count (about fifteen minutes from)}}$$

Definition

- A Statistical language model is the probability distribution P(s) over all possible word sequences or any other linguistic unit like words, sentences, paragraphs, documents .
- The dominant approach in Statistical language model is N-Gram model.
- The goal of Statistical language model is to estimate the probability (likelihood) of a sentence.
- This can be calculated by decomposing sentence probability into a product of conditional probabilities using the chain rule.

$$P(s) = P(w_1\ w_2\ w_3, \ldots\ w_n)$$

$$= P(w1) * P(w2/w1) * P(w3/w1w2) * P(w4/w1w2w3)$$

$$\ldots\ldots P(w_n/w_1w_2w_3\ldots..w_{(n-1)})$$

$$= \prod P(w\,i \;/\; w1w2\ldots wi\text{-}1)$$

$$= \prod P(w\,i \,/\, h\,i)$$

Where hi is history of word wi defined as w1w2w3…..wn-1

- In order to calculate sentence probability, we need to calculate the probability of a word, given the sequence of words preceding it.
- An N-gram model simplifies the task by approximating the probability of a word given all the previous words by the conditional probability given previous (n-1) words only.
- $P(w_i / h_i) = P(w_i / w_{i-n-1}\ldots w_i)$
- An N-gram model calculates $P(w_i / h_i)$ by modelling language as Markov Model of order (n-1) words. i.e by looking at previous (n-1) words only.

Markov Assumption Hidden Markov Model

Andrei Markov

- Markov Assumption use only the previous word.
- One previous word
- The office is about fifteen minutes from my house

Example : P(office/ about fifteen minutes from) ≈ P(office / from)

- Couple previous words
- P(office / about fifteen minutes from) ≈ P (office/ minutes from)
- More formally: k th order Markov model
- Using Markov assumption only k previous words
- $P(w_1 w_2 \ldots w_n) \approx \prod P(w_i / w_{i-k} \ldots w_{i-1})$
- We approximate each component in the product
- $P(w_i / w_1 w_2 \ldots w_{i-1}) \approx P(w_i / w_{i-k} \ldots . w_{i-1})$

N-Gram Models

- An N-gram model is an (N-1) order Markov Model.
- An N-gram Model uses only (N-1) words of prior context.
- $P(w_1 w_2 \ldots w_n) \approx \prod P(w_i / w_{i-k} \ldots w_{i-1})$ -------1
- $P(w_i / w_1 w_2 \ldots w_{i-1}) \approx P(w_i / w_{i-k} \ldots . w_{i-1})$ ------ 2
- From 1 & 2 it is clear that
- $P(w_1 w_2 \ldots w_n) \approx \prod P(w_i / w_1 w_2 \ldots w_{i-1})) \approx \prod P(w_i / w_{i-k} \ldots w_{i-1})$

Unigram model

- A model that has no conditions the probability of a word to the previous words is called Uni-gram model
- The office is about fifteen minutes from my house
- Example : P(office/ about fifteen minutes from) = P(office)
- $P(w_1 w_2 \ldots w_n) \approx P(w_1) * P(w_2) * \ldots P(w_n)$

$$\approx \prod P(w_i)$$

Bi-gram Model

- A model that limits the history previous one word only is called as Bi-gram (n=1) model.
- This model Calculates the probability of a sentence as follows

$$P(s) \approx \prod P(w_i / w_{i-1})$$

$$P(w_i / w_1 w_2 \ldots w_{i-1}) \approx P(w_i / w_{i-1})$$

The office is about fifteen minutes from my house

Example : P(office/ about fifteen minutes from) = P(office / from)

The Bi-gram approximation of

P(east / the Arabian Knights are fairly tales of the) is P(east / the)

Example : I am here
who am I
I would like to know

- <s>I am here </s>
- <s>who am I </s>
- <s>I would like to know </s>
- Bi-grams are
- <s>I, Iam, amhere, here </s>
- <s>who, whoam, amI, I </s>
- <s>I, Iwould, wouldlike, liketo, toknow, know </s>

Tri-gram Model

- A Model that conditions the probability of a word to the previous two words is called Tri-gram (n=2) model

$$P(s) \approx \prod P(w_i / w_{i-2} w_{i-1})$$

$$P(w_i / w_1 w_2 \ldots w_{i-1}) \approx P(w_i / w_{i-2}\ w_{i-1})$$

The office is about fifteen minutes from my house

Example : P(office/ about fifteen minutes from) = P(office / minutes from)

The Bi-gram approximation of

P(east / the Arabian Knights are fairly tales of the) is P(east / of the)

Processing Indian Languages

- There are number of differences between Indian languages and English.
- Unlike English, Indic scripts have non-linear structure
- Unlike English, Indian languages have SOV(subject-object-verb) as the default sentence structure.
- Indian languages have a free word order i.e words can be moved freely within a sentence without changing the meaning of the sentence.
- Spelling standardization is more subtle in Hindi than in English.
- Indian languages have a relatively rich set of morphological variants.
- Indian languages make extensive and productive use of complex predicates.

- Indian languages use post position case markers instead of prepositions.
- Indian languages use verb complexes consisting of sequences of verbs. The auxiliary verbs in this sequence provide information about tense, aspect and modality.
- Except for the direction in which its script is written, Urdu is closely related to Hindi.
- Both share similar phonology, morphology and syntax . Both are free-word –order languages and use post positions.
- Paninian grammars provides a framework Indian language models. These can be used for computation of Indian languages.
- The grammar focuses on extraction of karaka relations from a sentence.