Natural Language Processing
Lecture 01 Introduction; Grammars; Morphology; Word Frequency; Collocations

Qun Liu, Valentin Malykh
Huawei Noah's Ark Lab

Spring 2020
A course delivered at MIPT, Moscow

Content

Content

Logistics

- Instructors: Prof. Qun Liu, Valentin Malykh
- Time: 18.30 each Thursday
- Location: Klimentovsky lane, 1 bld. 1, auditorium 308
- Slides: will be uploaded to the course website before each class.
- Mailing List:
- Website:

Course description

Natural Language Processing (NLP) is a domain of research whose objective is to analyze and understand human languages and develop technologies to enable human machine interactions with natural languages. NLP is an interdisciplinary field involving linguistics, computer sciences and artificial intelligence. The goal of this course is to provide students with comprehensive knowledge of NLP. Students will be equipped with the principles and theories of NLP, as well as various NLP technologies, including rule-based, statistical and neural network ones. After this course, students will be able to conduct NLP research and develop state-of-the-art NLP systems.

Grading policy

Assignments

Projects

Examinations

Content

Natural language processing in Wikipedia

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

Synonyms of NLP

- **Computational Linguistics**
- **Natural Language Processing**
- **Natural Language Understanding**
- **Human Language Processing**

- Subtleties

  Computational Linguistics is more regarded as a branch of Linguistics, whose main purpose is to understand the mechanism of human languages by means of computing

Synonyms of NLP

- **Computational Linguistics**
- **Natural Language Processing**
- **Natural Language Understanding**
- **Human Language Processing**

- Subtleties

  Natural Language Processing is a branch of computer sciences and artificial intelligent, whose main purpose is to develop technologies to enable human-computer interactions using human languages

Synonyms of NLP

- **Computational Linguistics**
- **Natural Language Processing**
- **Natural Language Understanding**
- **Human Language Processing**

- Subtleties

  Natural Language Understanding is one of the two main challenges in Natural Language Processing, while the other is **Natural Language Generation.**

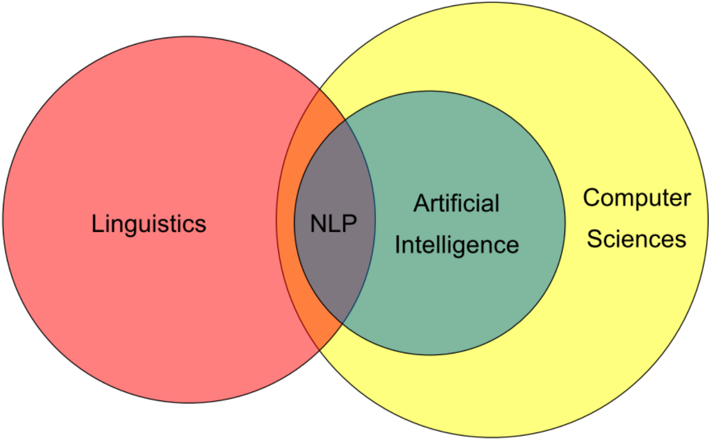
Synonyms of NLP

- **Computational Linguistics**
- **Natural Language Processing**
- **Natural Language Understanding**
- **Human Language Processing**

- Subtleties

  Human Language Technologies mainly refer to NLP technologies, but may also include other language related technologies, include speech technologies, optical character recognition (OCR), computer typesetting, etc.

NLP as an interdisciplinary study

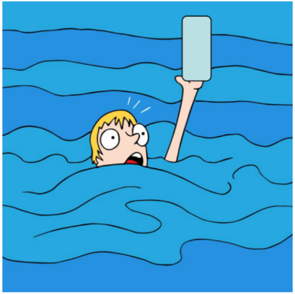


Understanding human languages is not easy

- We are getting used to the fact that human beings can understand each other using language communication.
- Although it is a natural result of evolution for human to obtain the language competence.
- It seems to be a miracle, due to its complexity.
- No other species in this planet can use languages at the degree as humans do.
- The mechanism behind human languages is not fully discovered.
- Understanding human languages by computer is difficult.

Understanding human languages is not easy

Tell my wife I love her!



From Husband:

I love her!

Research questions

- How humans understand each other by using language communication?
- Is it possible to simulate human language behaviors without understanding language mechanisms?

The way of NLP research

- Unlike linguists who develop numerous theories to explain the language mechanisms, NLP researchers try to simulate human language behaviors by computing, not necessary to understand the language mechanisms.

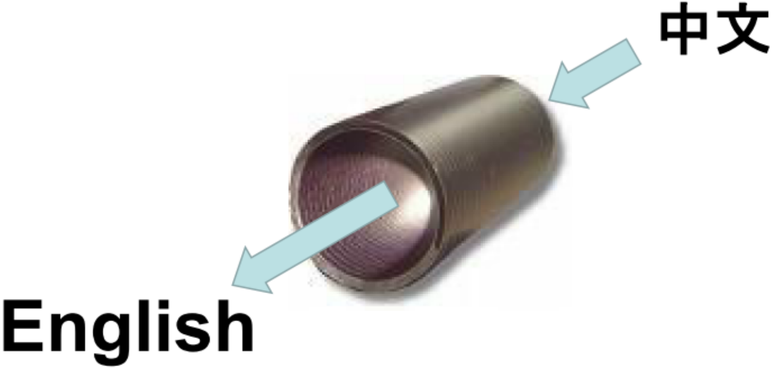
A brief history of NLP

- 1960s-1990s: Rule-based approaches
- 1990s-2010s: Statistical approaches
- 2010s-present: Neural network (deep learning) approaches

Holy grails of NLP

- Accurate machine translation between human languages
- Free conversation between humans and computers

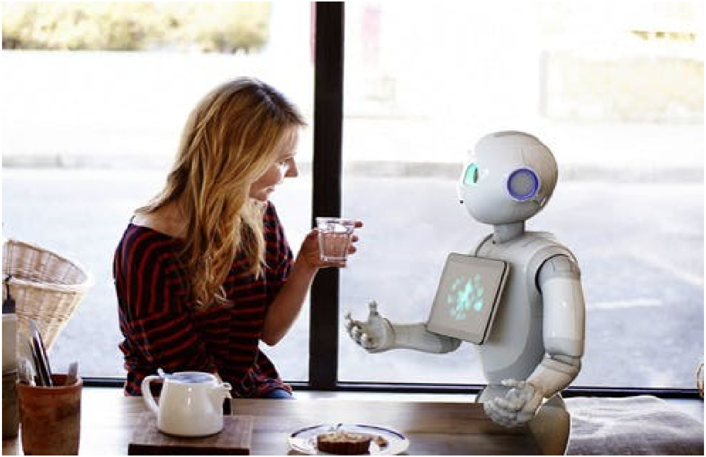
Accurate machine translation



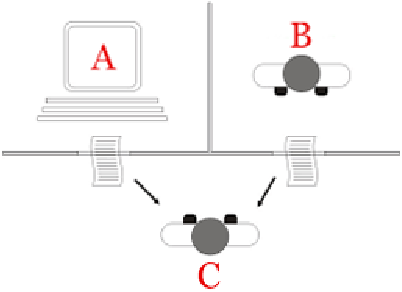
The Tower of Babel

Oil painting by Pieter Bruegel the Elder, 1563, from Wikipedia

Free human machine conversation



Turing test

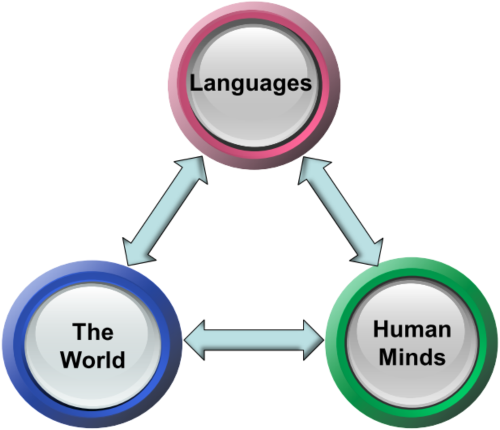


By Juan Alberto Sánchez Margallo, CC BY 2.5, from Wikipedia

NLP Tasks

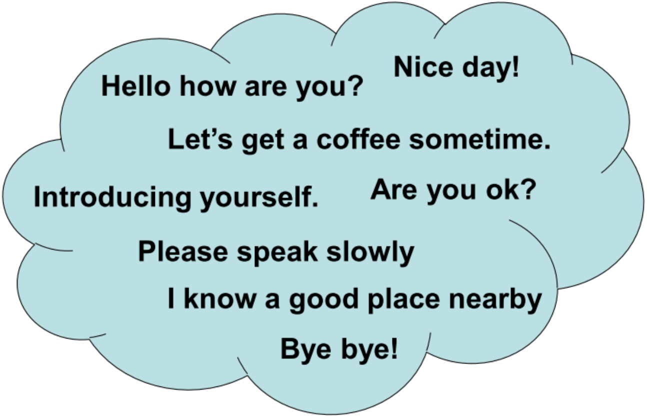
	Word / Phrase	Sentence	Document
Features / Expressions	word frequency, colocations, one-hot vectors, word embeddings	sentence embeddings, language models	bag of word / n-grams, word frequency vectors, tf-idfs, key words / phrases extraction, topic distributions, document embeddings
Classification	part-of-speech tagging, word sense disambiguation	sentiment analysis	text classification, sentiment analysis
Sequence labeling / Segmentation	stemming	word segmentation, part-of-speech tagging, named entity recognition	sentence segmentation, coreference resolution
Structural prediction / Parsing	morphological analysis	constituent parsing, dependency parsing, semantic role labeling, semantic parsing	discourse parsing
Sequence Generation /	machine translation, text summarization, dialog, style transfer, question answering, machine reading comprehension		

Languages, human minds and the world



Content

How can we define a language?



How can we define a language?

- A language can be defined as the set of sentences which can be accepted by the speakers of that language.
- It is not possible to define a natural language by enumerate all the sentences, because the number of sentences in a natural languages is infinite.
- Two feasible ways to define a language with infinite sentences:
  - By a Grammar
  - By an Automaton

Define a language by a grammar

- A grammar G is defined as:
  - A finite set of rules, and,
  - A mechanism to generate word sequences by applying the rules in G in a finite number of time steps.
- A sentence of a language is defined as:
  - A word sequence S is called a sentence of a language L if and only if S belongs to L.
- Given a grammar G, a language L could be defined by G as:
  - A word sequence S is a sentence of L if and only if S can be generated by G.

Define a language by an automaton (1)

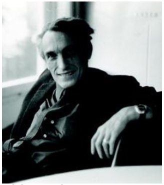
- An automaton A is a abstract machine which:
  - Takes a symbol sequence S as input, and determines if A will *accept* or *reject* S.
  - Has a finite number of states and a finite number of actions.
  - At each time step, S is in a state, and points to a position in S.
  - The current state and current symbol determines the action which A will execute, which determines the next state of A and the next position of S where A will point to.
  - Given a input S, A will run until it stops, and the final state of A determines if A will *accept* or *reject* S.

Define a language by an automaton (2)

- A language L can be defined by an automaton A as:
  - A word sequence S is a sentence of L, if and only if: when we input S to A, A will stop in a finite number of time steps at an *accept* state.

Chomsky hierarchy of formal grammars

- The Chomsky hierarchy (occasionally referred to as the Chomsky–Schützenberger hierarchy) is a nested hierarchy of classes of formal grammars.
- This hierarchy of grammars was described by Noam Chomsky in 1956.
- It is also named after Marcel-Paul Schützenberger, who played a crucial role in the development of the theory of formal languages.

Wikipedia - Chomsky hierarchy

Formal grammars

- A formal grammar G is a quadruple $\{N, T, S, R\}$:
  - R: a finite set of production rules ($\text{left-hand side} \rightarrow \text{right-hand side}$, or $\text{LHS} \rightarrow \text{RHS}$), where each side consists of a finite sequence of the symbols from N, T or $\{S\}$
  - N: a finite set of nonterminal symbols (indicating that some production rule can yet be applied)
  - T: a finite set of terminal symbols (indicating that no production rule can be applied)
  - S: a start symbol (a distinguished nonterminal symbol)

Formal grammars and formal languages

- A *formal grammar G* provides an axiom schema for a *formal language L*, which is a set of finite-length sequences of symbols that may be constructed by applying *production rules* to another sequence of symbols, which initially contains just the *start symbol*.
- A *production rule* may be applied by replacing an occurrence of the symbols on its left-hand side (LHS) with those that appear on its right-hand side (RHS).

Formal grammars and formal languages

- A sequence of rule applications is called a *derivation*.
- A formal grammar G defines a formal language L: all sequences of symbols consisting solely of terminal symbols which can be reached by a derivation from the start symbol.

An Example

The language $\{a^n b^n\}$ (i.e. n copies of a followed by n copies of b) can be defined by the following grammars:

Grammar 1	Grammar 2
Terminals: $\{a, b\}$	Terminals: $\{a, b\}$
Nonterminals: $\{S, A, B\}$	Nonterminals: $\{S\}$
Rules:	Rules:
$S \rightarrow AB$	$S \rightarrow aSb$
$S \rightarrow \epsilon$	$S \rightarrow \epsilon$
$S \rightarrow aS$	
$S \rightarrow b$	

ϵ is an empty string

Chomsky hierarchy of grammars

Grammar	Languages	Production Rules	Examples
Type 0	Recursively Enumerable	$\alpha A \beta \rightarrow \delta$	
Type 1	Context Sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$	$L = \{ a^n b^n c^n \| n > 0\}$
Type 2	Context Free	$A \rightarrow \alpha$	$L = \{ a^n b^n \| n > 0\}$
Type 3	Regular	$A \rightarrow a$ or $A \rightarrow aB$	$L = \{ a^n \| n > 0\}$

Type 0 grammars

- Also called Unrestricted Phrase Structure Grammars.
- A Type 0 Grammar generates a Recursively Enumerable Language.
- A Recursively Enumerable Language L is semi-decidable by a Turing Machine T:
  - For any sentence S in L, T will accept it in a finite number of time steps.
  - For a sentence S not in L, it is not guaranteed that T can reject it in a finite number of time steps.

Type 1 grammars

- Also called Context Sensitive Grammar.
- A Type 1 Grammar generates a Context Sensitive Language.
- A Context Sensitive Language is decidable by a Linear Bounded Automaton and the complexity of this decision problem is NP-complete.
- It is not practical to use Type 1 Grammars in NLP because of its time complexity.

Type 2 grammars

- Also called Context Free Grammars.
- A Type 2 Grammar generates a Context Free Language.
- A Context Free Language is decidable by a Pushdown Automaton and the complexity of this decision problem is polynomial.
- Type 2 Grammars are the theoretical basis of all programming languages.
- Type 2 Grammars are commonly used in NLP, however, natural languages are not context free languages actually.

Type 3 grammars

- Also called Regular Grammars.
- A Type 3 Grammar generates a Regular Language.
- A Context Free Language is decidable by a Finite State Automaton / Machine and the complexity of this decision problem is linear.
- Type 3 Grammars are the theoretical basis of the lexical analyzers of all programming languages.
- Type 3 Grammars are very broadly used in NLP for many different purposes.

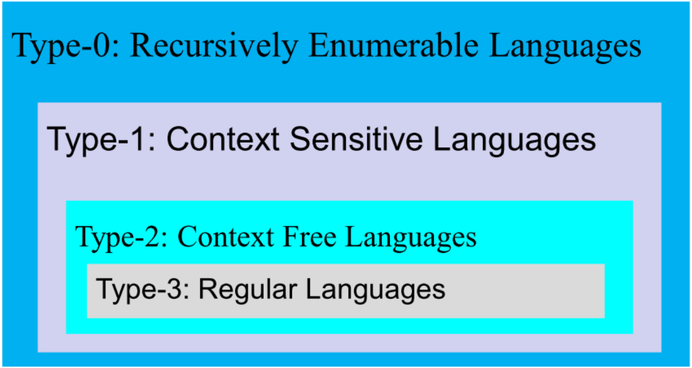
Regular expressions

- Regular Expressions are very useful tools which are supported by most of the modern programming languages and text editors.
- A Regular Expression is equivalent to a Regular Grammar, and vice versa.

Regular expressions in Python

Character	Description	Example
[]	A set of characters	"[a-m]"
\	Signals a special sequence (can also be used to escape special characters)	"\d"
.	Any character (except newline character)	"he..o"
^	Starts with	"^hello"
$	Ends with	"world$"
*	Zero or more occurrences	"aix*"
+	One or more occurrences	"aix+"
{}	Exactly the specified number of occurrences	"al{2}"
|	Either or	"falls|stays"
()	Capture and group	

Chomsky hierarchy



Set inclusions described by the Chomsky hierarchy

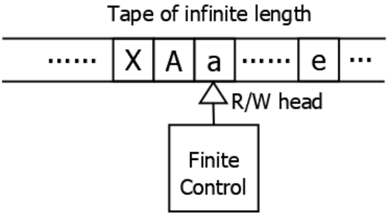
Other grammar classes

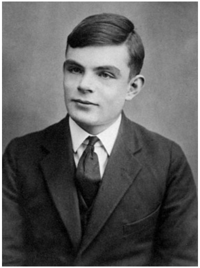
- Mild Context Sensitive Grammars:
  - Grmmars classes which define subsets of Context Sensitive Languages, but beyond Context Free Languages.
  - Examples:
    - Index Grammars (IGs)
    - Tree Adjoin Grammars (TAGs)

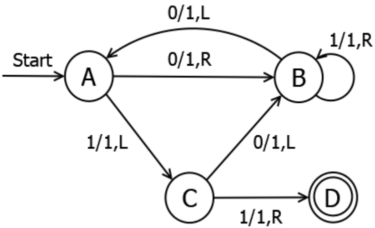
Grammars and automata

Grammar	Languages	Automaton	Decidability and Complexity
Type 0	Recursively Enumerable	Turing Machine	Semi-Decidable
	Recursive	Decider, or Total Turing Machine	Decidable
Type 1	Context Sensitive	Linear Bounded Automaton (LBA)	NP Complete
Type 2	Context Free	Pushdown Automaton (PDA)	Polynomial
	Deterministic Context Free	Deterministic Pushdown Automaton (PDA)	Linear
Type 3	Regular	Deterministic / Nondeterministic Finite State Machine (FSM)	Linear

Turing machine







Turing machine

A Turing machine consists of: (to be continued)

- A *tape* divided into cells, one next to the other. Each cell contains a symbol from some finite alphabet. The alphabet contains a special blank symbol and some other symbols. The *tape* is assumed to be arbitrarily extendable to the left and to the right.
- A *read/write head* that can read and write symbols on the *tape* and move the *tape* left and right one (and only one) cell at a time.

Turing machine

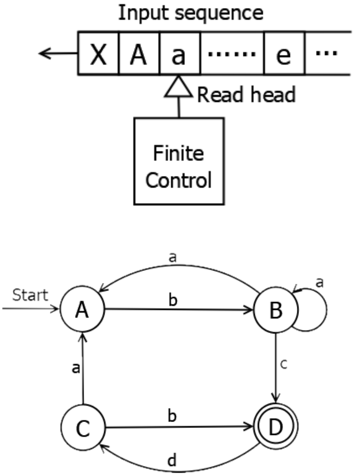
- A Turing machine consists of: (continued)
  - A *state register* that stores the state of the Turing machine, one of finitely many. Among these is the special *start state* with which the state register is initialized.
  - A finite *table* of instructions that, given the *state* the machine is currently in and the symbol it is reading on the *tape*, tells the machine to do the following in sequence:
    - Either erase or write a symbol.
    - Move the *head* to the left or right cell.
    - Assume the same or a *new state* as prescribed.

Linear bounded automaton

A linear bounded automaton is a Turing Machine that satisfies the following three conditions:

- Its input alphabet includes two special symbols, serving as *left and right endmarkers*.
- Its *transitions* may not print other symbols over the *endmarkers*.
- Its *transitions* may neither move to the left of the *left endmarker* nor to the right of the *right endmarker*.

Finite state automaton / machine (FSA/FSM)

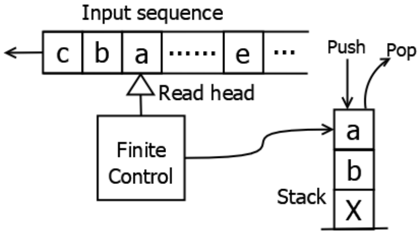


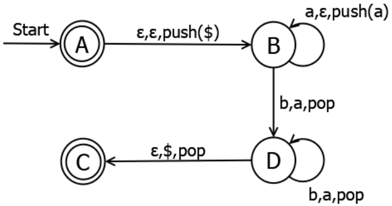
Finite state automaton / machine (FSA/FSM)

A Finite State Automaton (FSA), or Finite State Machine (FSM), consists of:

- A finite number of *states*, while the FSM can be in one *states* at each given time;
- A *head* which read a symbol from a sequence of symbols as the *input*. The *head* always goes to the next symbol at the next time step;
- A *transition* matrix which determines the next *states* of the FSM according to the current *states* and the current symbol.

Pushdown automaton (PDA)





Pushdown automaton (PDA)

- A pushdown Automaton (PDA) is similar to DFA except that it maintains a *stack*:
  - It can use the top of the *stack* to decide which *transition* to take. In each step, it chooses a *transition* by indexing a table by *input symbol*, *current state*, and the *symbol* at the top of the *stack*.
  - It can manipulate the *stack* as part of performing a *transition*. The manipulation can be to *push* a particular symbol to the top of the *stack*, or to *pop* off the top of the *stack*.

Content

Text segmentation

- In NLP, text is segmented into units of various granularities, which include:
  - Chapters and sections;
  - Paragraphs;
  - Sentences;
  - Clauses;
  - Phrases;
  - Words;
  - Morphemes (stems, suffixes, prefixes).

Text segmentation

- Text segmentation is not straightforward in many cases:
  - For languages like Chinese, Japanese, Tibetan, Thai, there are no spaces between words;
  - For languages like Thai and Tibetan, the delimiters between sentences, clauses or phrases are not ambiguous, which makes it hard to segment sentences;
  - Even for English, sentence segmentation is not a trivial task, because the full stop mark (.) is also used for abbreviations, decimals, etc., which may or may not terminate a sentence.

Thai

โลกเราเป็นอะไรหนอในช่วงนี้ ฝั่งหนึ่งของโลกมีอากาศ
อันแปรปรวนวิปริต หนาวเหน็บอย่างไม่เคยเกิดขึ้นมาก่อน
และยังเกิดแผ่นดินพิโรธโกรธาคร่าชีวิตคนไปเป็นเรือนแสน
ส่วนบ้านเรานั้นในปีที่ผ่านมาแทบไม่มีฤดูหนาวให้ชื่นใจกันเลย
อากาศกลับร้อน แถมมีทั้งฝนหลงฤดูในช่วงนี้อีกต่างหาก
ทุกคนพูดว่า เป็นเพราะภาวะโลกร้อนนั่นเองที่ทำให้ทุกอย่าง
ดูไม่เหมือนเดิม ประเทศที่มีอากาศหนาวก็หนาวสุดขั้ว ประเทศ

Spaces are not reliable boundaries between sentences.

Chinese

西游记 4　真假猴王

师徒四人继续西行。有一天，他们来到一个地方，前面是望不到边的水面，唐僧发愁（chóu）道："这么大的水，怎么过去呢？"

四个人正不知道怎么办，忽然看见远处好像有一个人在河边，于是就走过去，想问一问。

走近了一看，那不是一个人，而是一块石头，石头上写着三个大字"通天河"，旁边还有一行小字——"河宽（kuān）八百里，自古少人行"，意思是这条河有八百里宽，很少有人能通过。

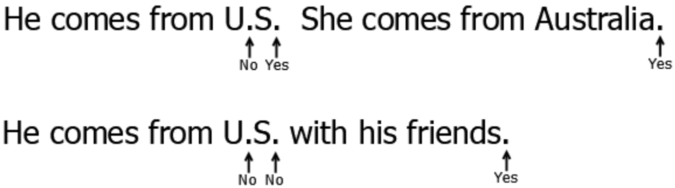
There are not spaces between words.

English sentence segmentation

- Dot marks (.) are ambiguous:
  - Full stop: *This is an apple.*
  - Decimal: *235.6*
  - Abbreviations: *U.S. Ph.D. etc.*
  - A dot mark can take multiple roles: *He comes from U.S.*
- To segment English text into sentences, we need to determine whether a dot mark is an end of sentence or not.
- It can be solved as a classification problem.

English sentence segmentation

— as a classification task



Chinese word segmentation

(a)

下雨天留客天留我不留	Unpunctuated Chinese sentence
下雨、天留客。天留、我不留！	*It is raining, the god would like the guest to stay. Although the god wants you to stay, I do not!*
下雨天、留客天。留我不？ 留！	*The rainy day, the staying day. Would you like me to stay? Sure!*

(b)

我喜欢新西兰花	Unsegmented Chinese sentence
我　喜欢　新西兰　花	*I like New Zealand flowers*
我　喜欢　新　西兰花	*I like fresh broccoli*

http://what-when-how.com/how-to-build-a-digital-library/word-segmentation-and-sorting-digital-library/

Chinese word segmentation may results in different meanings.

Chinese word segmentation

— as a character tagging task

S	S	B	E	B	M	E	S
我	有	一	台	计	算	机	。
(I)	(have)	(a)		(computer)			(.)

Wang & Xu, Convolutional Neural Network with Word Embeddings for Chinese Word Segmentation, IJCNLP 2017

Tags:

- **S**: single character word
- **B**: beginning character of a word
- **M**: middle character of a word
- **E**: end character of a word

English word segmentation - Tokenization

— A example of Stanford Tokenizer

Input

Another ex-Golden Stater, Paul Stankowski from Oxnard, is contending for a berth on the U.S. Ryder Cup team after winning his first PGA Tour event last year and staying within three strokes of the lead through three rounds of last month's U.S. Open. H.J. Heinz Company said it completed the sale of its Ore-Ida frozen-food business catering to the service industry to McCain Foods Ltd. for about $500 million. It's the first group action of its kind in Britain and one of only a handful of lawsuits against tobacco companies outside the U.S.

Note: Text in red: change, text in blue: Keep

English word segmentation - Tokenization

— A example of Stanford Tokenizer

Output

Another ex-Golden Stater , Paul Stankowski from Oxnard , is contending for a berth on the U.S. Ryder Cup team after winning his first PGA Tour event last year and staying within three strokes of the lead through three rounds of last month 's U.S. Open . H.J. Heinz Company said it completed the sale of its Ore-Ida frozen-food business catering to the service industry to McCain Foods Ltd. for about $ 500 million . It 's the first group action of its kind in Britain and one of only a handful of lawsuits against tobacco companies outside the U.S. .

Note: Text in red: change, text in blue: Keep

Morphological analysis

- To break word down into component morphemes and build a structured representation
- A morpheme is the minimal meaning-bearing unit in a language.
    - **Stem**: the morpheme that forms the central meaning unit in a word
    - **Affix**: prefix, suffix, infix, circumfix
        - **Prefix**: e.g., possible → impossible
        - **Suffix**: e.g., walk → walking
        - **Infix**: e.g., hingi → humingi (Tagalog)
        - **Circumfix**: e.g., sagen → gesagt (German)

a slide from UW LING 570 by Fei Xia

Two slightly different tasks

- Stemming:
  - Ex: writing $\rightarrow$ writ + ing (or write + ing)
- Lemmatization:
  - Ex1: writing $\rightarrow$ write +V +Prog
  - Ex2: books $\rightarrow$ book +N +Pl
  - Ex3: writes $\rightarrow$ write +V +3Per +Sg

a slide from UW LING 570 by Fei Xia

Ambiguity in morphology

- flies $\rightarrow$ fly +N +PL
- flies $\rightarrow$ fly +V +3rd +Sg

a slide from UW LING 570 by Fei Xia

Language variation

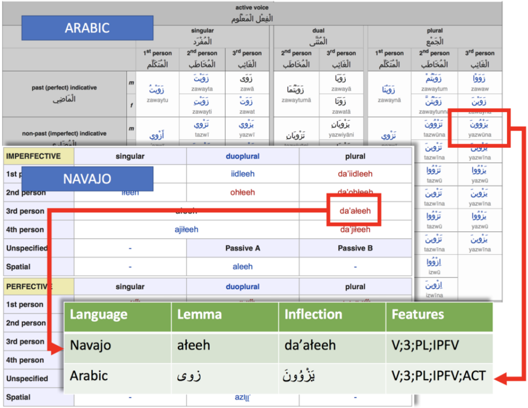
- Analytic languages: e.g., Chinese; English as a language with analytic tendency.
- Synthetic flexive languages: e.g., Russian
- Synthetic agglutinate languages: e.g., Turkish

Ways to combine morphemes to form words

- Inflection: stem + gram. morpheme $\rightarrow$ same class
  - Ex: help + ed $\rightarrow$ helped
- Derivation: Derivation: stem + gram. morpheme $\rightarrow$ different class
  - Ex: civil + -zation $\rightarrow$ civilization
- Compounding: multiple stems
  - Ex: cabdriver, doghouse
- Cliticization: stem + clitic
  - Ex: they'll, she's (*I don't know who she is)

a slide from UW LING 570 by Fei Xia

UniMorph 2.0: Universal Morphology

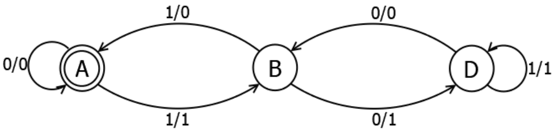


Kirov et al., UniMorph 2.0: Universal Morphology, LREC 2018

Finite state transducers (FSTs)

- Finite State Transducers are an extension to Finite State Machines, where an output symbol will be given for each input symbol.
- FSTs are commonly used tools for morphological analysis.
- A FST can be used in a inverse direction with the input and the output swapped.

Finite state transducers (FSTs)



input	output
0	0
11	01
110	010
1001	0011
1100	0100
1111	0101
10010	00110

English morphology

- Affixes: prefixes, suffixes; no infixes, no circumfixes.
- Inflectional:
  - Noun: -s
  - Verbs: -s, -ing, -ed, -ed
  - Adjectives: -er, -est
- Derivational:
  - Ex: V + suf $\rightarrow$ N
    computerize + -ation $\rightarrow$ computerization
    kill + er $\rightarrow$ killer
- Compound: pickup, database, heartbroken, etc.
- Cliticization: ’m, ’ve, ’re, etc.

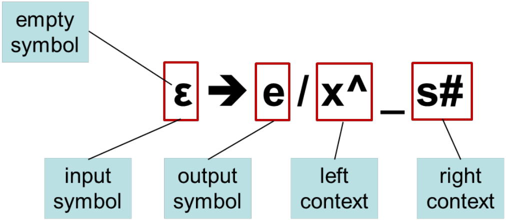
a slide from UW LING 570 by Fei Xia

Three components

- Lexicon: the list of stems and affixes, with associated features.
  - Ex1: book: N
  - Ex2: -s: +PL
- Morphotactics:
  - Ex: +PL follows a noun
- Orthographic rules (spelling rules): to handle exceptions that can be dealt with by rules.
  - Ex3: $\epsilon \rightarrow$ e / x ^ _ s#

a slide from UW LING 570 by Fei Xia

Rewrite rules



An example

Task: foxes ➔ fox +N +PL

Surface: foxes

↑ Orthographic rules

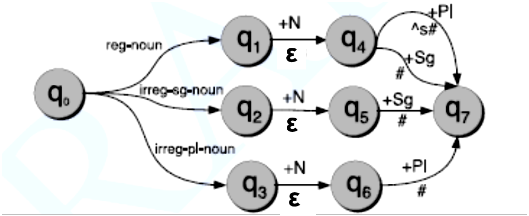
Intermediate: fox ^s

↑ Lexicon + morphotactics

Lexical: fox +N +pl

a slide from UW LING 570 by Fei Xia

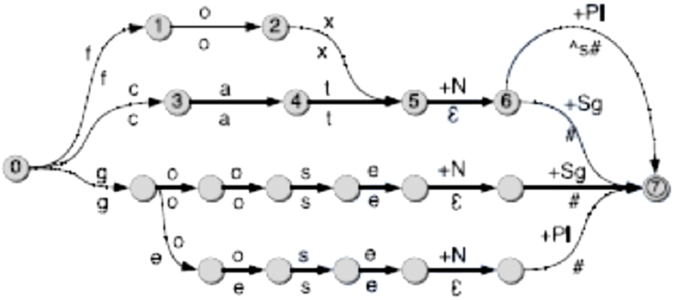
An FST



$$\text{cat +N +PL} \rightarrow \text{cat \^{}s \#}$$
$$\text{cat +N +Sg} \rightarrow \text{cat \#}$$

a slide from UW LING 570 by Fei Xia

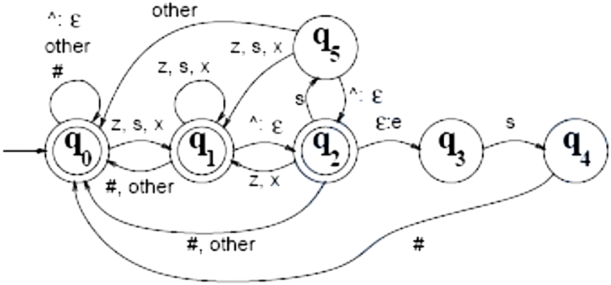
Expanding FST



fox +N + Pl → fox ˆ s #
cat +N + Pl → cat ˆ s #
goose +N +Sg → goose #
goose +N +Pl → geese #

a slide from UW LING 570 by Fei Xia

Representing orthographic rules as FSTs



$\epsilon \rightarrow$ e / (s|x|z) ˆ _ s #

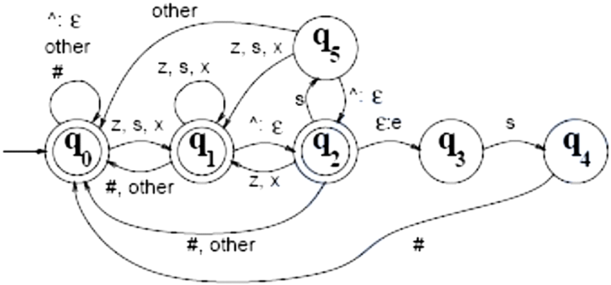
Input: ...(s|x|z) ˆs # immediate level

Output: ...(s|x|z)es # surface level

To reject (fox ˆs, foxs)

a slide from UW LING 570 by Fei Xia

Representing orthographic rules as FSTs



(fox, fox): q0, q0, q0, q1

(fox#, fox#): q0, q0, q0, q1, q0

(foxˆz#, foxz#), q0, q0, q0, q1, q2, q1, q0

(foxˆs#, foxes#): q0, q0, q0, q1, q2, q3, q4, q0

(foxˆs, foxs): q0, q0, q0, q1, q2, q5

a slide from UW LING 570 by Fei Xia

Further reading on morphological analysis

- Fei Xia, slides on morphological analysis

  https://www.powershow.com/viewfl/6a39a-ZDc1Z/Morphological_analysis_powerpoint_ppt_presentation

- Mans Hulden (2011), Morphological analysis with FSTs

  https://fomafst.github.io/morphtut.html

Content

Top 5000 words in American English

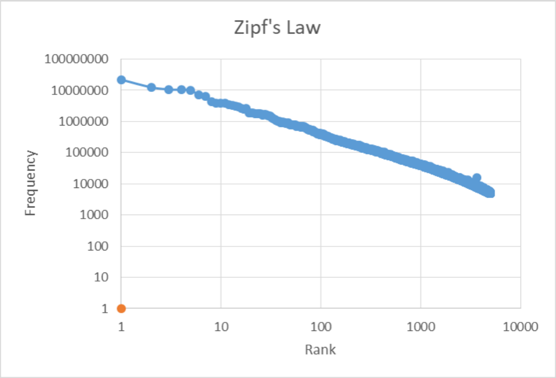
Rank	Word	Part of speech	Frequency	Dispersion
1	the	a	22038615	0.98
2	be	v	12545825	0.97
3	and	c	10741073	0.99
4	of	i	10343885	0.97
5	a	a	10144200	0.98
6	in	i	6996437	0.98
7	to	t	6332195	0.98
8	have	v	4303955	0.97
9	to	i	3856916	0.99
10	it	p	3872477	0.96

Rank	Word	Part of speech	Frequency	Dispersion
1	the	a	22038615	0.98
2	be	v	12545825	0.97
3	and	c	10741073	0.99
4	of	i	10343885	0.97
5	a	a	10144200	0.98
6	in	i	6996437	0.98
7	to	t	6332195	0.98
8	have	v	4303955	0.97
9	to	i	3856916	0.99
10	it	p	3872477	0.96

Statics from Corpus of the Contemporary American English

http://www.wordfrequency.info/

Top 5000 words in American English

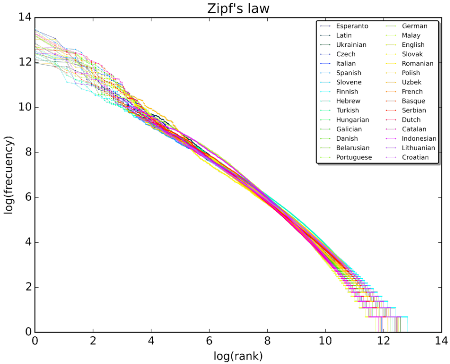


Zipf's Law

The frequency of any word is inversely proportional to its rank in the frequency table:

$$p(w_r) \propto \frac{1}{r}$$

Zipf's law



A plot of the rank versus frequency for the first 10 million words in 30 Wikipedias (dumps from October 2015) in a log-log scale.

(By SergioJimenez - Own work, CC BY-SA 4.0, from Wikipedia)

Collocation or multi-word expression (MWE)

- A COLLOCATION is an expression consisting of two or more words that correspond to some conventional way of saying things.
- The words together can mean more than their sum of parts
  - The Times of India, disk drive
  - hot dog, mother in law

Collocation or multi-word expression (MWE)

- Examples of collocations
  - noun phrases like *strong tea* and *weapons of mass destruction*
  - phrasal verbs like to *make up*, and other phrases like the *rich and powerful*.
- Valid or invalid?
  - *a stiff breeze* but not a *stiff wind* (while either a *strong breeze* or a *strong wind* is okay).
  - *broad daylight* (but not bright daylight or narrow darkness).

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Criteria for collocations (or MWE)

- Typical criteria for collocations:
  - non-compositionality
  - non-substitutability
  - non-modifiability.
- Collocations usually cannot be translated into other languages word by word.
- A phrase can be a collocation even if it is not consecutive (as in the example *knock ... door*).

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Non-Compositionality

- A phrase is compositional if the meaning can be predicted from the meaning of the parts.
  - E.g. new companies
- A phrase is non-compositional if the meaning cannot be predicted from the meaning of the parts
  - E.g. *hot dog*

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Non-Compositionality

- Collocations are not necessarily fully compositional in that there is usually an element of meaning added to the combination.
    - E.g. *strong tea*
- Idioms are the most extreme examples of non-compositionality
    - E.g. *to hear it through the grapevine*

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Non-Substitutability

- We cannot substitute near-synonyms for the components of a collocation.
- For example
  - We can't say *yellow wine* instead of *white wine* even though *yellow* is as good a description of the color of *white* wine as white is (it is kind of a yellowish white).

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Non-Substitutability

- Many collocations cannot be freely modified with additional lexical material or through grammatical transformations (Non-modifiability).
  - E.g. *white wine*, but not *whiter wine*
  - E.g. *mother in law*, but not *mother in laws*

Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999

Metrics for Collocation or MWE Extraction

- Frequency
- Mean and Variance of Distances between Words
- Hypothesis Testing
  - *t*-test
  - χ^2 test
  - likelihood ratio test
- Mutual Information
- Left and Right Context Entropy
- C-Value

Further reading on collocation and MWE

- Manning & Schütze, Fundamentals of Statistical Natural Language Processing, 1999, Chapter 3 (A general introduction to collocation)
- Katerina T. Frantzi, Sophia Ananiadou, Junichi Tsujii, The C-value / NC-value Method of Automatic Recognition for Multi-word Terms, ECDL 1998: Research and Advanced Technology for Digital Libraries pp 585-604 (proposed the C-value metric)
- Zhiyong Luo, Rou Song, An integrated method for Chinese unknown word extraction, SIGHAN 2004. Barcelona, Spain. (proposed the context entropy method)

Content