CS 481

Artificial Intelligence Language Understanding

January 17, 2023

Announcements / Reminders

- **Please follow the Week 01 To Do List instructions**

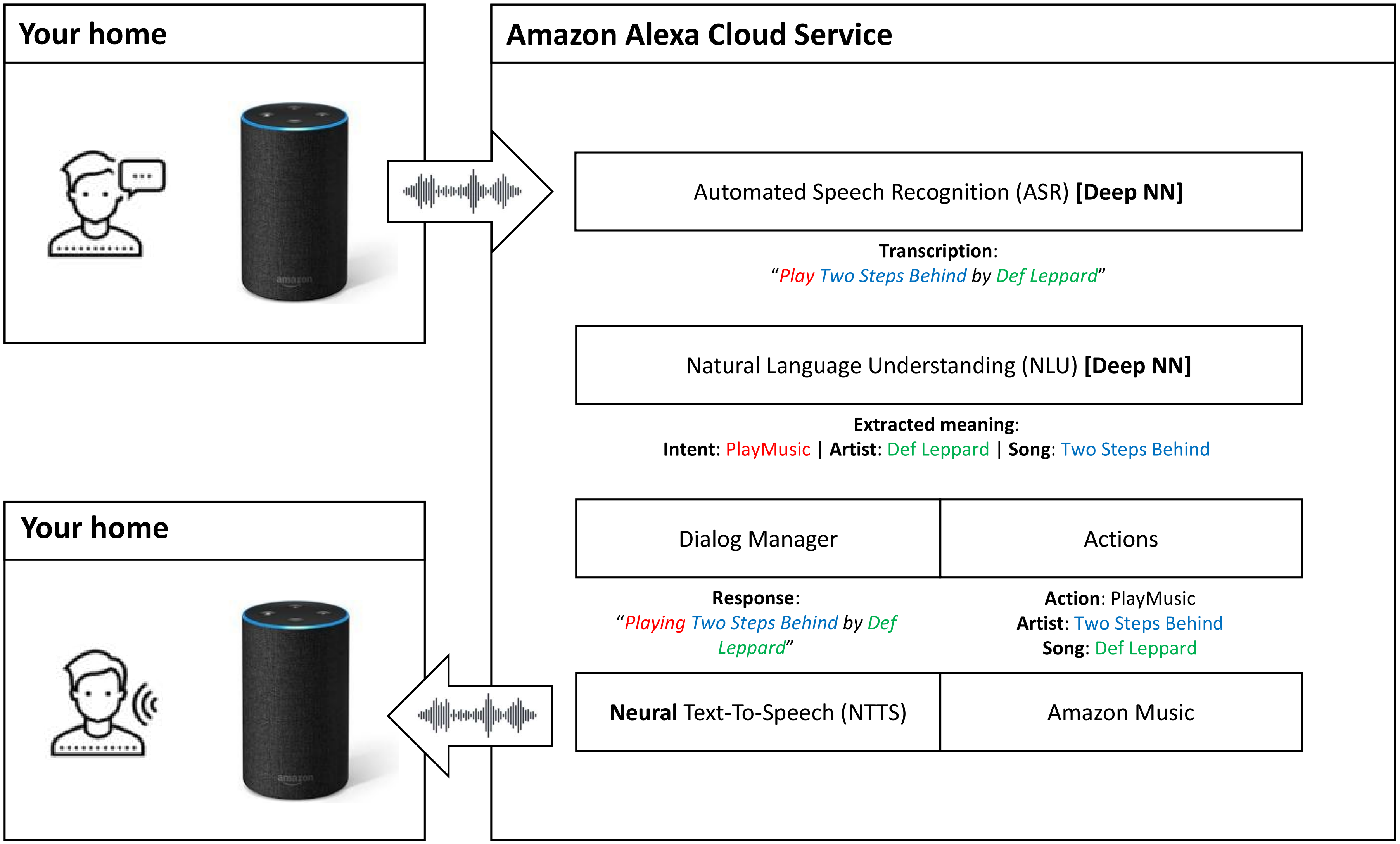
- **Quiz #01 due on Sunday 01/22/23 at 11:59 PM CST**

- **Exam dates:**
  - **Midterm: 03/02/2023 during Thursday lecture time**
  - **Final: 04/27/2023 during Thursday lecture time**

Plan for Today

- Introduction to NLP - continued
- Language basics - continued
- Text pre-processing
- Regular Expressions (RegEx)
  - Introduction
  - RegEx for basic text pre-processing
- Python libraries / packages for NLP
- Text corpora

Voice Assistant: Alexa



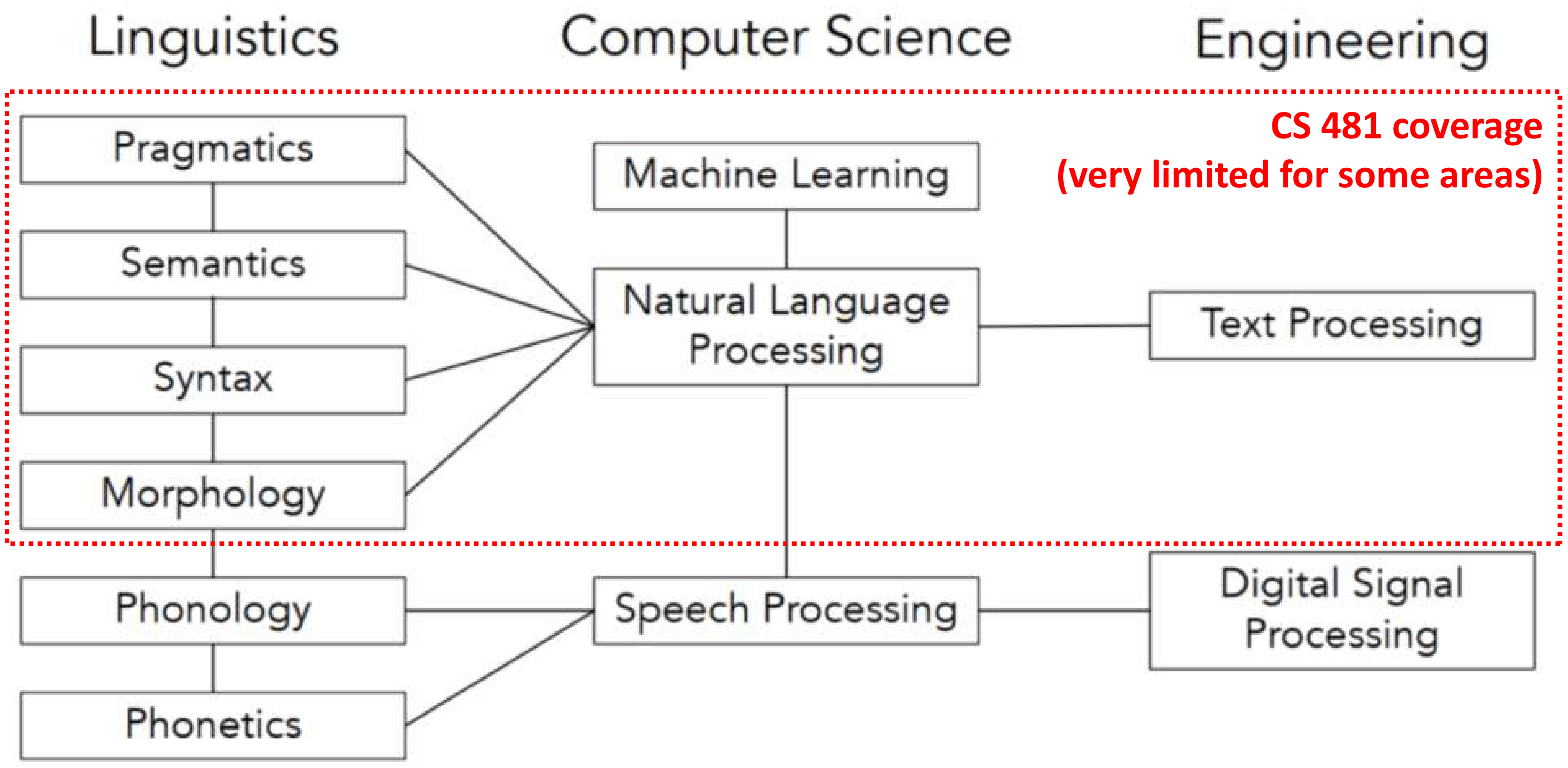
Your home
Amazon Alexa Cloud Service
Automated Speech Recognition (ASR) **[Deep NN]**
Transcription:
"*Play Two Steps Behind* by *Def Leppard*"
Natural Language Understanding (NLU) **[Deep NN]**
Extracted meaning:
Intent: PlayMusic | **Artist**: Def Leppard | **Song**: Two Steps Behind
Your home
Dialog Manager
Actions
Response:
"*Playing Two Steps Behind* by *Def Leppard*"
Action: PlayMusic
Artist: Two Steps Behind
Song: Def Leppard
Neural Text-To-Speech (NTTS)
Amazon Music

Generative Pre-trained Transformer 3

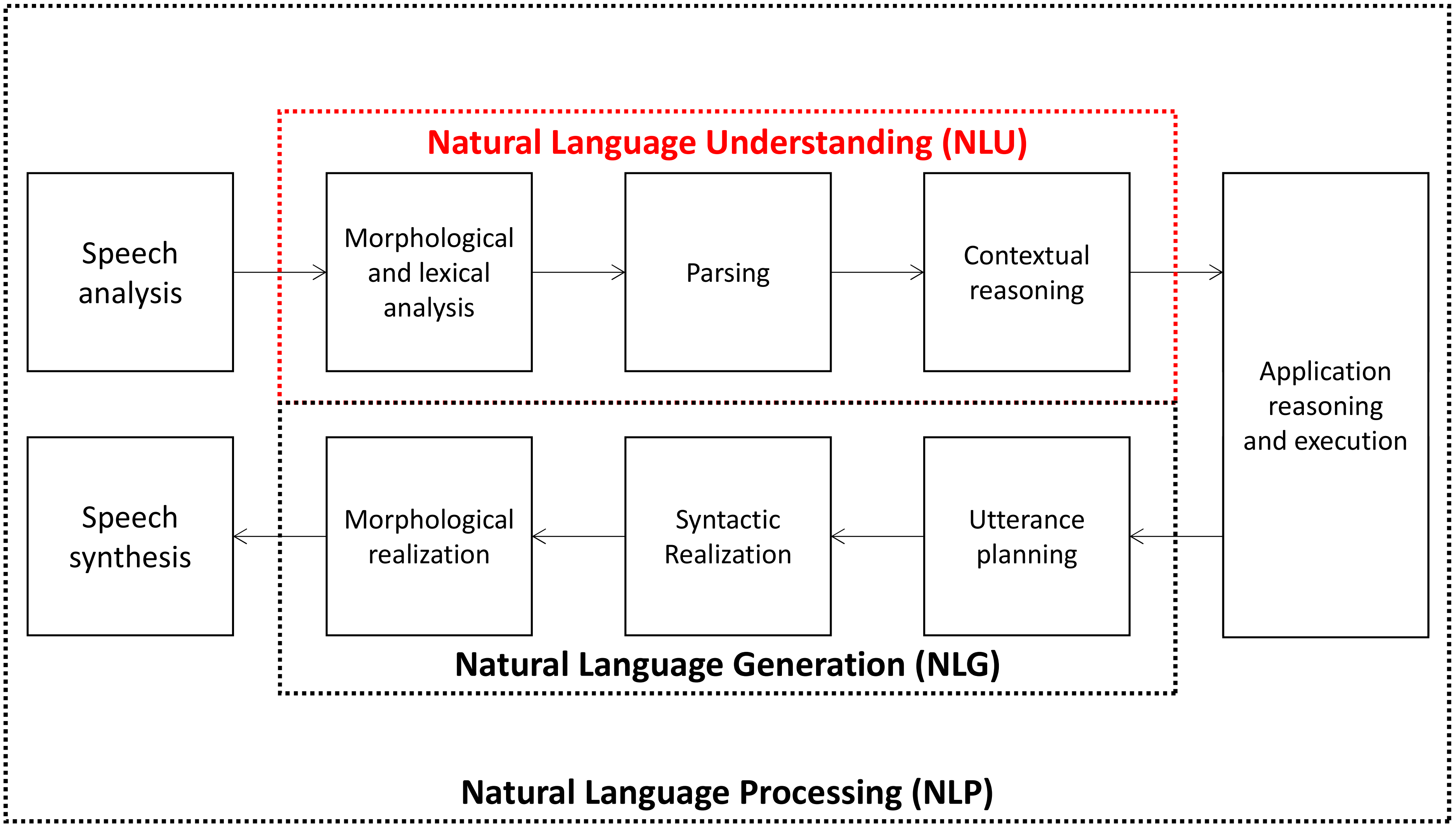
What is it?

Generative Pre-trained Transformer 3 (GPT-3) is an autoregressive **language model that uses deep learning to produce human-like text**. It is the third-generation language prediction model in the GPT-n series (and the successor to GPT-2) created by OpenAI, a San Francisco-based artificial intelligence research laboratory. GPT-3's full version has a capacity of **175 billion machine learning parameters**.

NLP vs. Adjacent Fields

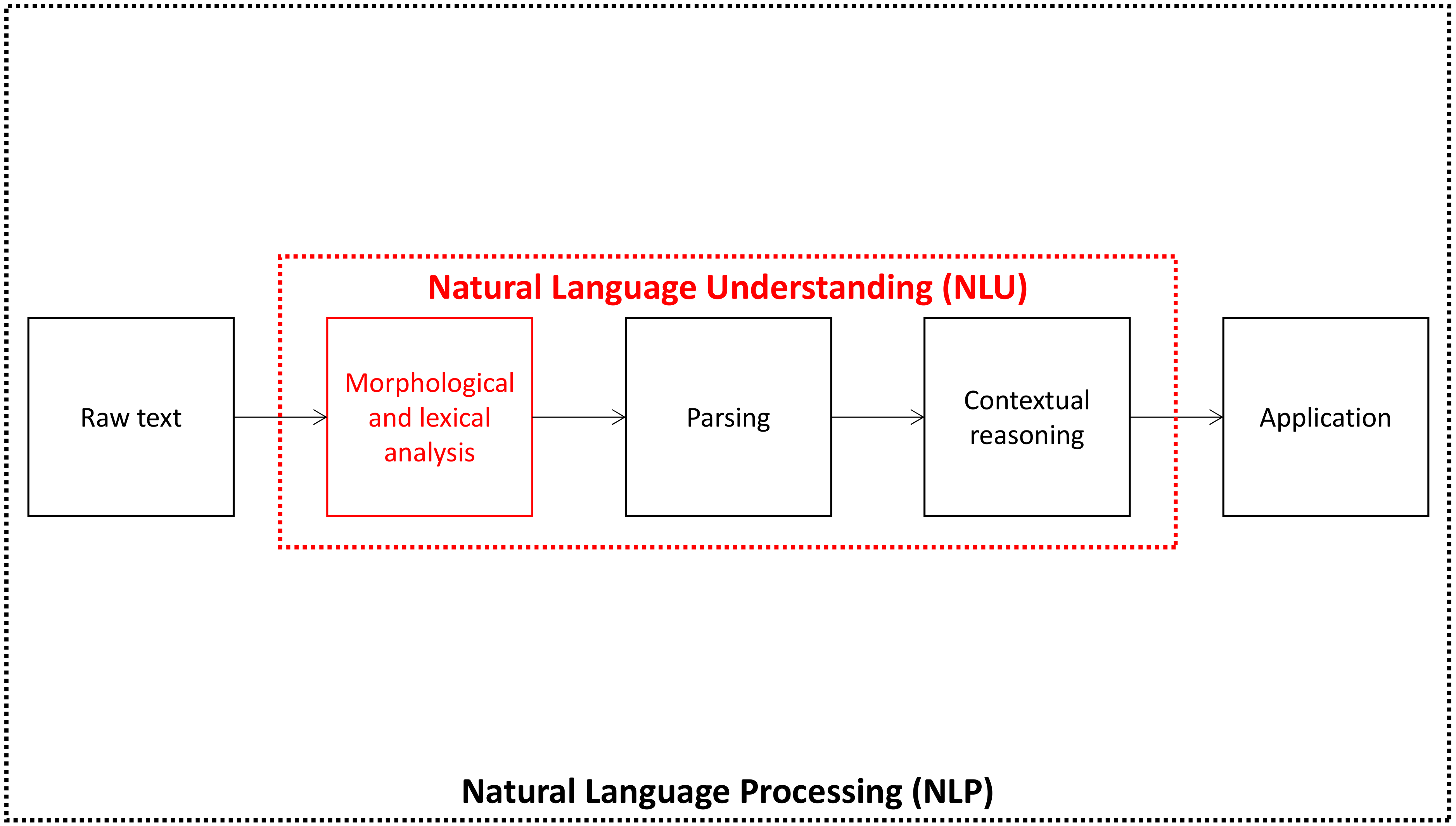


Basic NLP Spoken Language Pipeline



Natural Language Understanding (NLU)
Speech analysis
Morphological and lexical analysis
Parsing
Contextual reasoning
Application reasoning and execution
Speech synthesis
Morphological realization
Syntactic Realization
Utterance planning
Natural Language Generation (NLG)
Natural Language Processing (NLP)

Basic NLP Text Processing Pipeline


Natural Language Understanding (NLU)
Raw text
Morphological and lexical analysis
Parsing
Contextual reasoning
Application
Natural Language Processing (NLP)

Common Lexical Categories

Lexical category	Definition*	Example
Adjective	A word or phrase naming an attribute, added to or grammatically related to a noun to modify or describe it	The ***quick red*** fox jumped over the ***lazy brown*** dogs.
Adverb	A word or phrase that modifies or qualifies an adjective, verb, or other adverb, or a word group, expressing a relation of place, time, circumstance, manner, cause, degree, etc.	The dogs ***lazily*** ran down the field after the fox.
Conjunction	A word that joins two words, phrases, or clauses	The quick red fox ***and*** the silver coyote jumped over the lazy brown dogs.
Determiner	A modifying word that determines the kind of reference a noun or noun group has, for example ***a***, ***the***, ***very***	***The*** quick red fox jumped over ***the*** lazy brown dogs.
Noun	A word used to identify any of the class of people, places, or things, or to name a particular one of these.	The quick red ***fox*** jumped over the lazy brown ***dogs***.
Preposition	A word governing, and usually preceding, a noun or pronoun and expressing a relation to another word or element in the clause	The quick red fox jumped ***over*** the lazy brown dogs.
Verb	A word used to describe an action, state, or occurence, and forming the main part of the predicate of a sentence, such as ***hear***, ***become***, and ***happen***	The quick red fox ***jumped*** over the lazy brown dogs.

* all definitions are taken from the New Oxford American Dictionary, 2nd Edition

Lexical Categories: Subcategories

Nouns:

- **common nouns represent classes of entities:**
  - *town*, *ocean*, *person*
- **proper nouns represent unique entities:**
  - *London*, *John*, *Eiffel Tower*
- **pronouns are nouns representing other entities (usually mentioned previously):**
  - *he*, *she*, *it*

Morphology

- **Morphology is a study of the internal structure of words**
- **Words consist of:**
  - **lexeme (root form)**
  - **affixes (suffix, prefix)**
- **Morphology has two categories:**
  - **inflectional - does not create new lexemes (happier)**
  - **derivational - creates new lexemes (unhappy)**
- **Inlectional morphemes carry grammatical meaning (plural -s), but they do not change the meaning of the word**

Suffix	Example	Verb
-ation	nomination	nominate
-ee	appointee	appoint
-ure	closure	close
-al	refusal	refuse
-er	runner	run

Suffix	Example	Adjective
-dom	freedom	free
-hood	likelihood	likely
-ist	realist	real
-th	warmth	warm
-ness	happiness	happy

Suffix	Example	Marked form
N/A	look	base form
-ing	looking	gerund form
-s	looks	third person singular
-ed	loooked	past tense form
-en	taken	past participle

Phrases

- Phrases consist of multiple words
- Phrases are rooted by at least one word of a particular type, but can also consist of words and phrases of other types
- Phrases can be combined to form clauses that are the minimal units to construct a sentence

Phrases and Clauses

Every sentence is constructed from phrases and/or clauses.

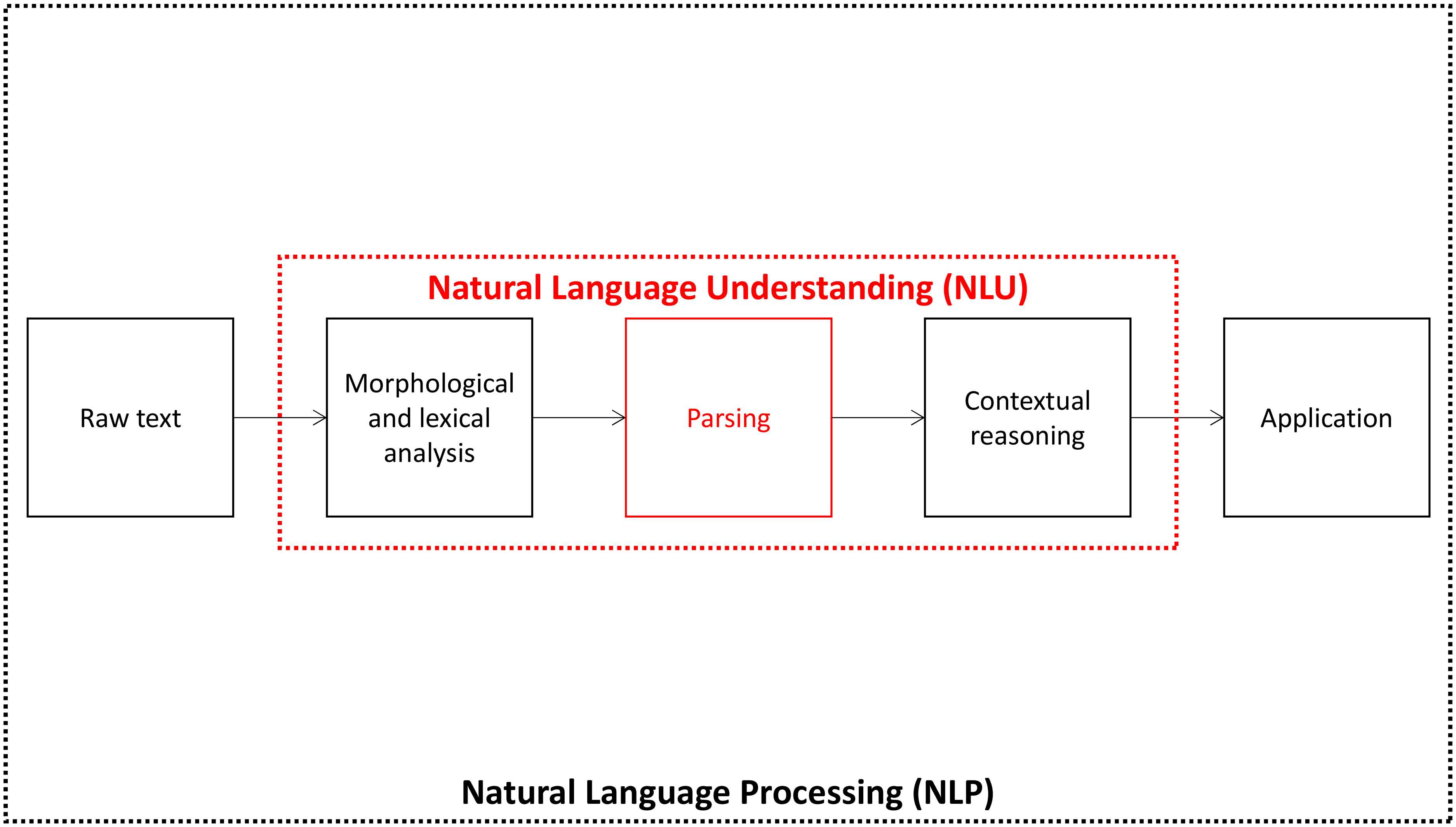
- **A phrase is a group of words, but it doesn't contain a subject and a verb.**
  - **Example:** *The big clock*
- **A clause is a group of words that contains a subject and a verb.**
  - **Example:** *The big clock chimed*

Phrase $\rightarrow$ Clause $\rightarrow$ Sentence

Common Phrasal Categories

Type	Example	Comments
Adjective	The *unusually red* fox jumped over the *exceptionally lazy* dogs.	The adverbs *unusually* and *exceptionally* modify the adjectives *red* and *lazy*, respectively, to create adjectival phrases.
Adverb	The dogs *almost always* ran down the field after the fox.	The adverb *almost* modifies the adverb *always* to create adverbial phrase.
Conjunction	The quick red fox *as well as* the silver coyote jumped over the lazy brown dogs.	Though this is somewhat of an exceptional case, you can see that the phrase *as well as* performs the same function as a conjunction such as *and*.
Noun	*The quick red fox jumped* over *the lazy brown dogs*.	The noun *fox* and its modifiers *the*, *quick*, and red create a noun phrase, as does the noun *dogs* and its modifiers *the*, *lazy*, and *brown*.
Preposition	The quick red fox jumped *over the lazy brown dogs*.	The preposition *over* and the noun phrase *the lazy brown dogs* form a prepositional phrase that modifies the verb *jumped*.
Verb	The quick red fox *jumped over the lazy brown dogs*.	The verb *jumped* and its modifier the prepositional phrase *over the lazy brown dogs* form a verb phrase.

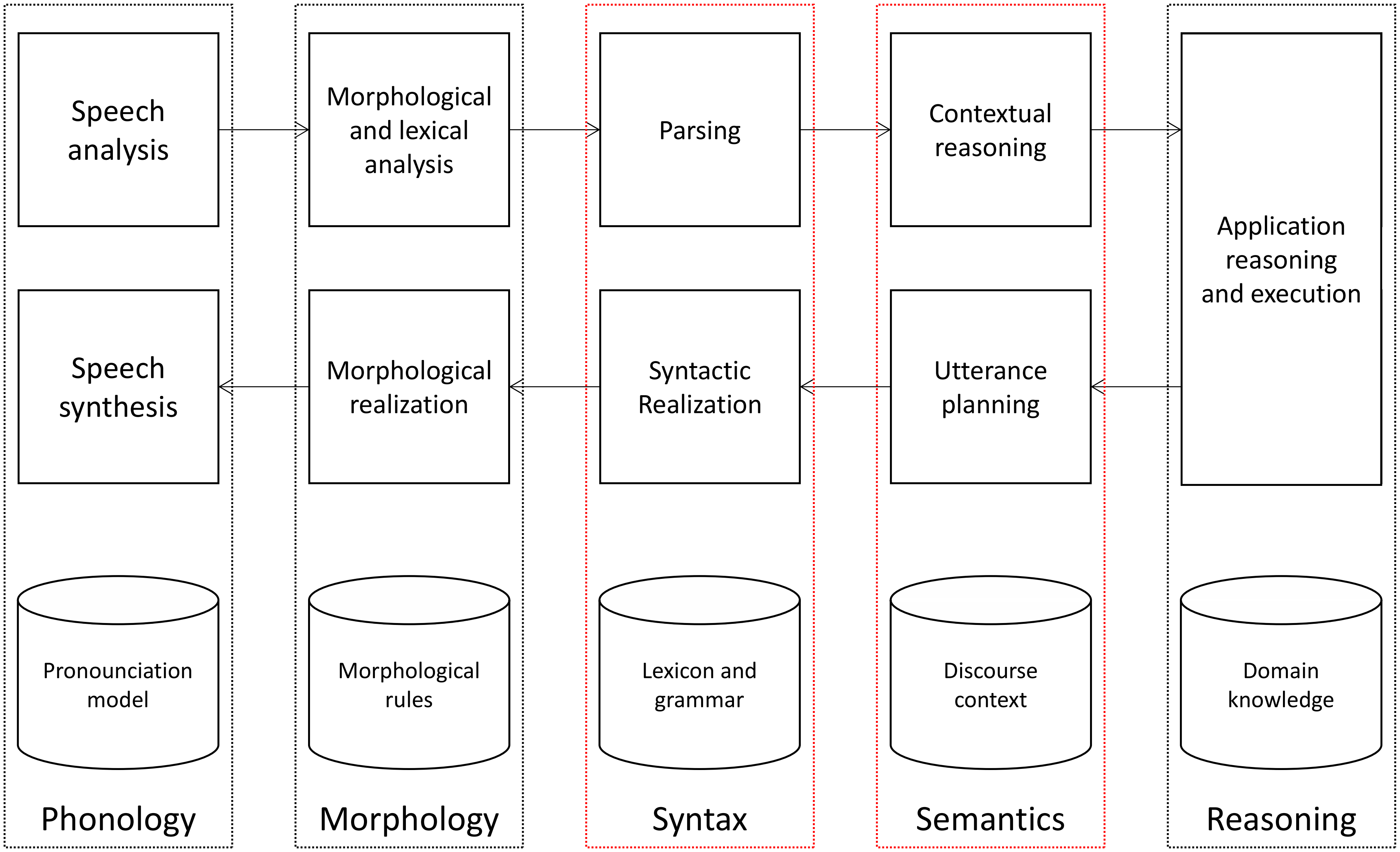
Basic NLP Text Processing Pipeline


Natural Language Understanding (NLU)
Raw text
Morphological and lexical analysis
Parsing
Contextual reasoning
Application
Natural Language Processing (NLP)

Parsing

The task of determining the parts of speech, phrases, clauses, and their relationship to one another is called parsing.

Basic NLP Spoken Language Pipeline



Knowledge Levels / Forms for NLP

Level	Description
Phonetic and phonological knowledge	Concerned with how the words are related to sounds that realize them. Such knowledge is crucial for speech-based systems.
Morphological knowledge	Concerned with how words are constructed from the basic meaning units called **morphemes**.
Syntactic knowledge	Concerned with how words can be put together to form correct sentences and determines what structural role each word plays in the sentence and what phrases are subparts of what other phrases.
Semantic knowledge	Concerned with what the words mean and how these meanings combine in sentences to form sentence meanings. This is the study of context-independent meaning - the meaning a sentence has regardless of the context in which it is used.
Pragmatic knowledge	Concerned with how sentences are used in different situations and how use affects the interpretation of the sentence.
Discourse knowledge	Concerned with how the immediately preceding sentences affect the interpretation of the next sentence. This information is especially important for interpreting pronouns and for interpreting the temporal aspects of the information.
World knowledge	Includes the general knowledge about the structure of the world that language users must have in order to, for example, maintain a conversation. It includes what each language user must know about the other user's beliefs and goals.

(English) Syntax

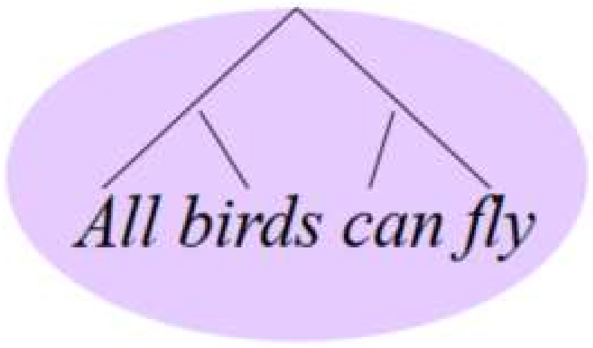
The structure of words and phrases within a sentence:

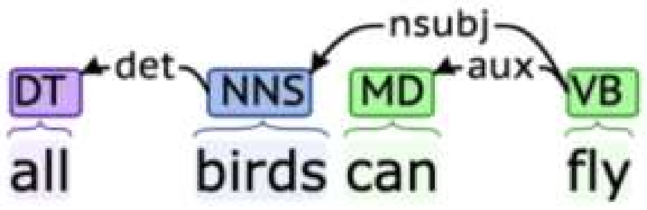
- Different formalisms, coming from the American (phrase structure) and European (dependency grammar) structuralist traditions

Applications:

- Part-of-speech tagging
- Entity extraction
- Syntactic parsing (Context-Free Grammar)
- Syntactic parsing (dependencies)

Examples:





Semantics

The representation of meaning in language:

- **at different levels: lexical, sentential, textual**
- **logical formalisms: reference and truth conditions**

Applications:

- **Word embedding / encoding**
- **Lexical resources**
- **Semantic role labeling**

Example:

$$\forall x \, [\mathrm{bird}(x) \Rightarrow \mathrm{fly}(x)]$$

Pragmatics

How language is used to achieve specific intentions:

- conversational implicatures: how I interpret what you say because of what I assume you are trying to do
- speech acts

Applications:

- Speech act labeling
- Discourse structure parsing
- Dialogue systems

Examples:

"I ate most of your cookies"

□

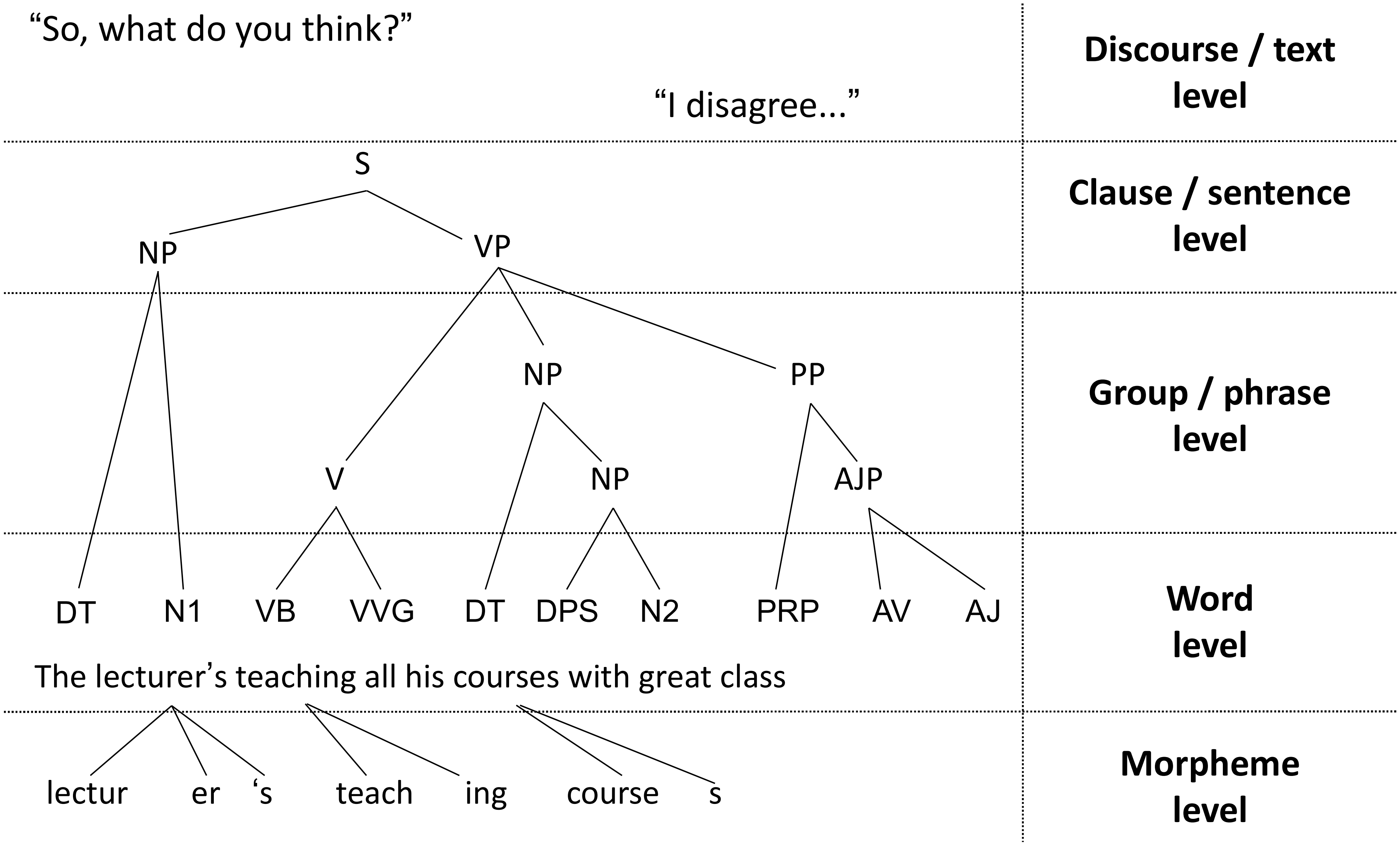
I did not eat all of your cookies

"Where does your brother live?"

□

I do not know where your brother lives

Structure / Rank Levels for NLP



Syntax - Semantics - Pragmatics

- Syntax: what is its "formal" relation structure?
- Semantics: what does it "mean"?
- Pragmatics: how is it "used"?

Consider a hypothetical "first" CS 481 sentence:

Sentence	Syntax	Semantics	Pragmatics
Language is one of the fundamental aspects of human behavior and is a crucial component of our lives.	👍	👍	👍
Green frogs have large noses.	👍	👍	👎
Green ideas have large noses.	👍	👎	👎
Large have green ideas nose.	👎	👎	👎

Knowledge - Rank Mapping: NLP Tasks

Rank / Domain	Syntax	Semantics	Pragmatics
Word	Parts-of-speech Morphology	Word senses Word similarity	Sentiment analysis
Group / Phrase	Shallow parsing	Named entity recognition Semantic role labeling	Deixis Coreference
Clause / Sentence	Parsing	Information extraction Entailment	Speech act interpretation Sentiment analysis
Discourse / Text	Rhetorical discourse structure	Text categorization Story understanding	Coherence Sentiment analysis

Representations and Understanding

Text understanding involves computing a representation of the meaning of sentences and texts.

- **The sentence itself <u>is not enough</u> to represent the meaning:**
  - **word *cook* has a verb and a noun sense,**
  - **word *catch* can mean a baseball move, a fish, etc.**

Representation Language Properties

Useful representation language have two properties:

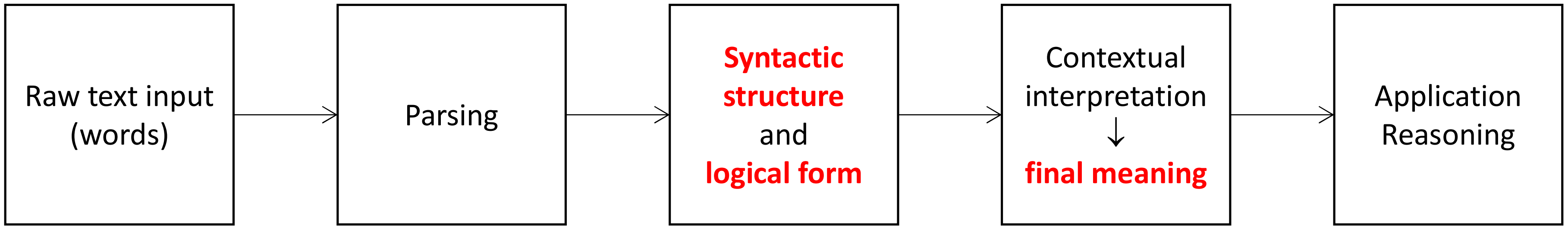
- The representation must be precise and unambiguous. You should be able to express every distinct reading of a sentence as a distinct formula in the representation.
- The representation should capture the intuitive structure of the natural language sentences that it represents. For example:
  - sentences that are structurally similar should have similar structural representations, and
  - the meanings of two sentences that are paraphreses of each other should be closely related to each other.

Syntax - Logical Form - Final Meaning

Representation can be realized using:

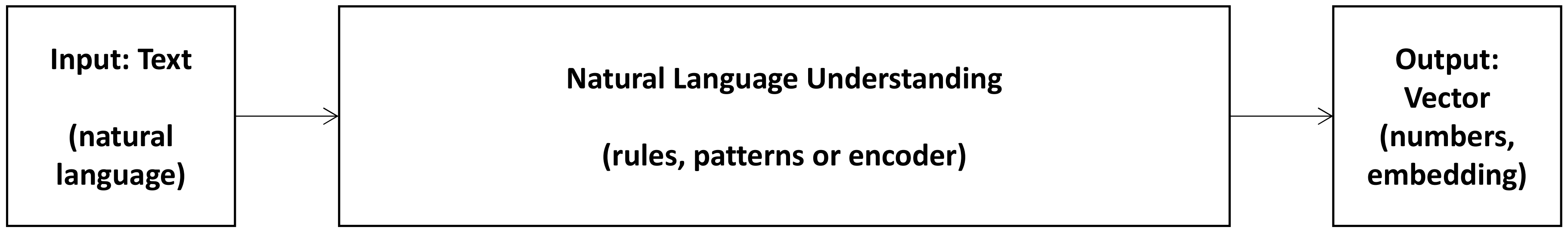
- **sentence syntactic structure: it indicates the way that individual words in a sentence**
  - **are related to each other**
  - **grouped together into phrases**
  - **modify other words**
  - **are of central importance**
- **the logical form: context-independent meaning of a sentence**
- **the final meaning: general knowledge representation meaning | context-dependent meaning**

NLU: Flow of Information


Raw text input (words)
Parsing
Syntactic structure and logical form
Contextual interpretation ↓ final meaning
Application Reasoning

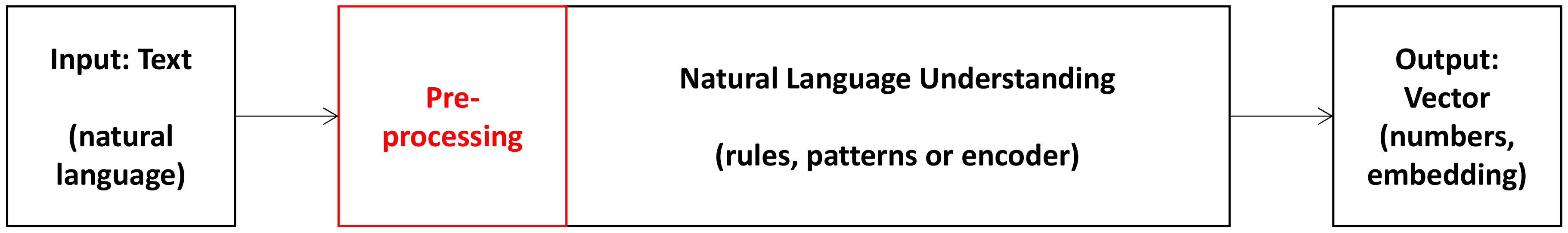
Automated Text Processing

The task of automatic processing of text is to extract a numerical representation of the meaning of that text. This is the natural language understanding (NLU) part of NLP. The numerical representation of the meaning of natural language usually takes the form of a vector called an embedding.

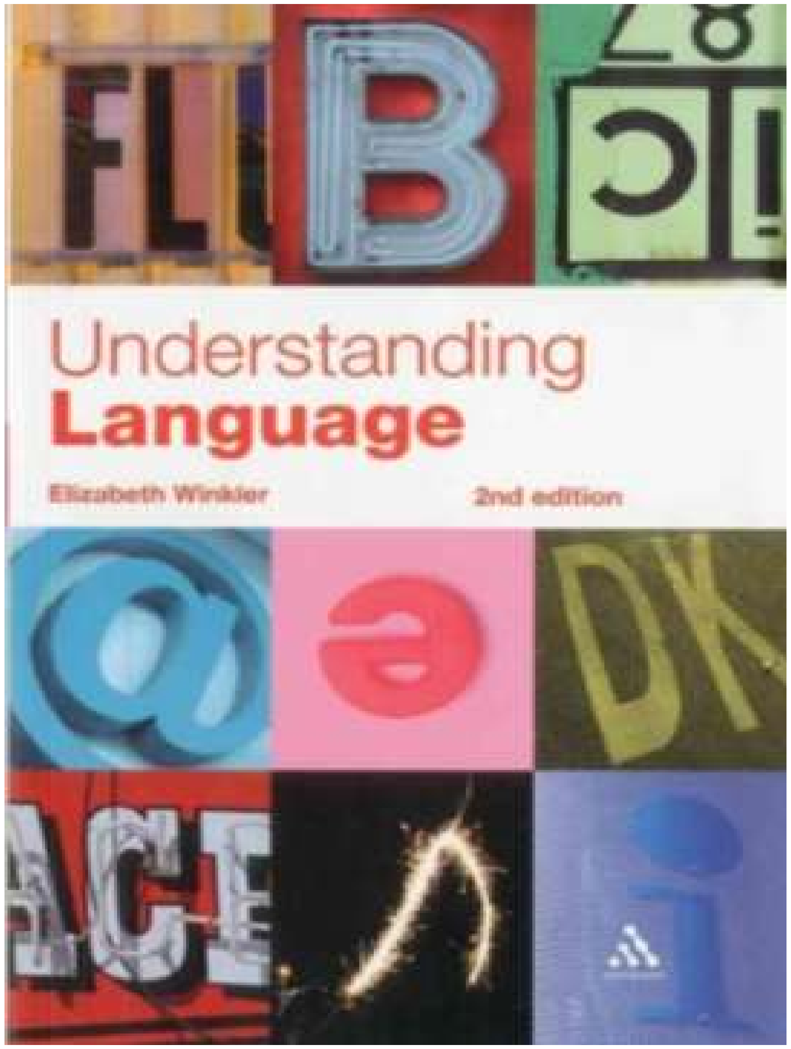

Input: Text
(natural language)
Natural Language Understanding
(rules, patterns or encoder)
Output: Vector (numbers, embedding)

Automated Text Processing

The task of automatic processing of text is to extract a numerical representation of the meaning of that text. This is the natural language understanding (NLU) part of NLP. The numerical representation of the meaning of natural language usually takes the form of a vector called an embedding.


Input: Text
(natural language)
Pre-processing
Natural Language Understanding
(rules, patterns or encoder)
Output: Vector (numbers, embedding)

Optional Reading



Title:

Understanding Language (2nd edition)

Authors:

Elizabeth Winkler

ISBN:

9781441138965

Publisher:

Continuum Books

Published:

March 29, 2012

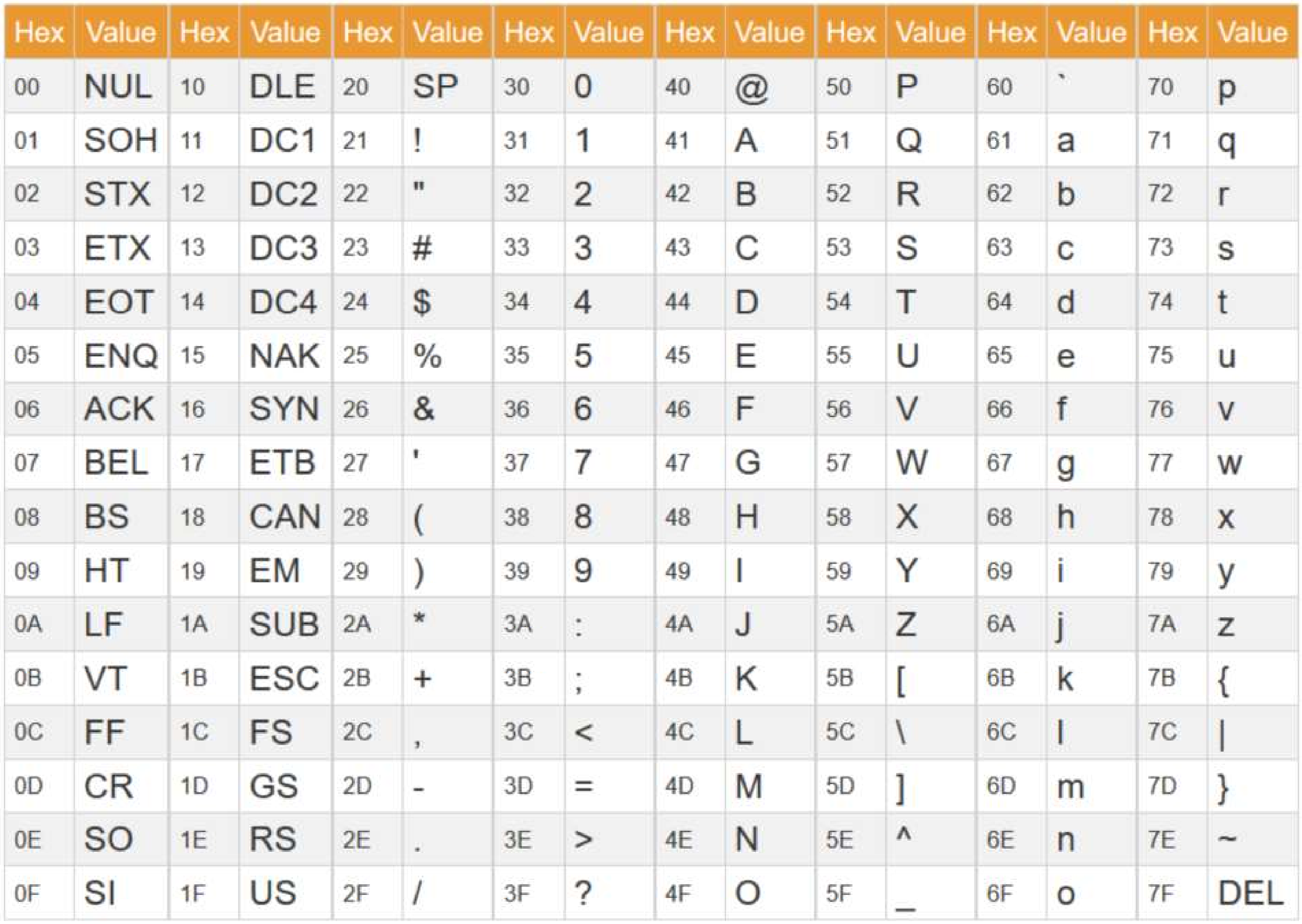
Character Encoding Standards

Encoding standards specify how to interpret individual text characters in documents.

Well-known standards for English:

- **ASCII (American Standard Code for Information Interchange)**
- **ISO 8859-1 Latin 1**
- **Unicode UTF-8 (also UTF-16 and others):**
  - **first 128 characters: ASCII**
  - **first 256 characters: ISO 8859-1 Latin 1**

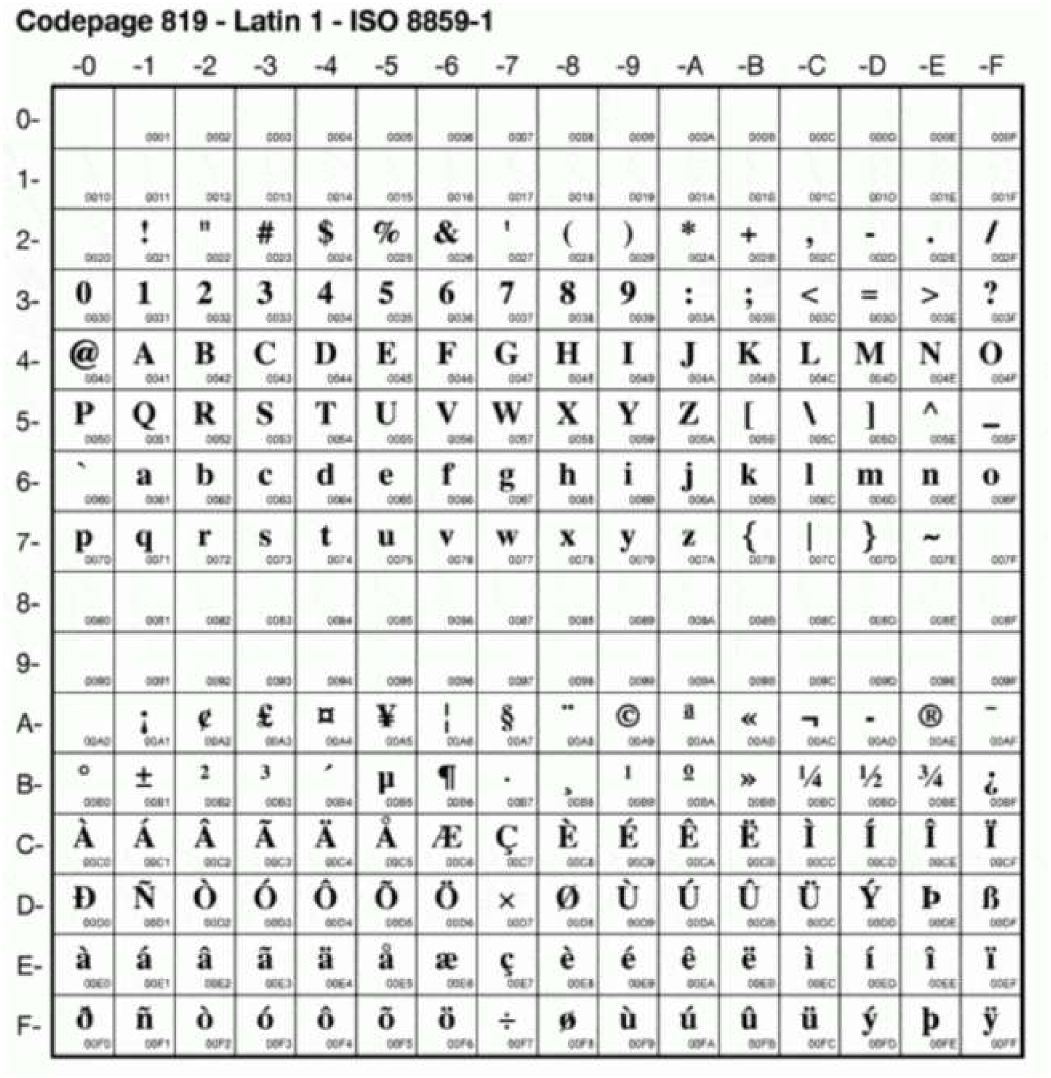
Character Encoding: ASCII



Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value
00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	`	70	p
01	SOH	11	DC1	21	!	31	1	41	A	51	Q	61	a	71	q
02	STX	12	DC2	22	"	32	2	42	B	52	R	62	b	72	r
03	ETX	13	DC3	23	#	33	3	43	C	53	S	63	c	73	s
04	EOT	14	DC4	24	$	34	4	44	D	54	T	64	d	74	t
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e	75	u
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f	76	v
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g	77	w
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h	78	x
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i	79	y
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
0B	VT	1B	ESC	2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
0C	FF	1C	FS	2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

Source: https://www.sciencebuddies.org/science-fair-projects/references/ascii-table

Character Encoding: ISO 8859-1 Latin 1

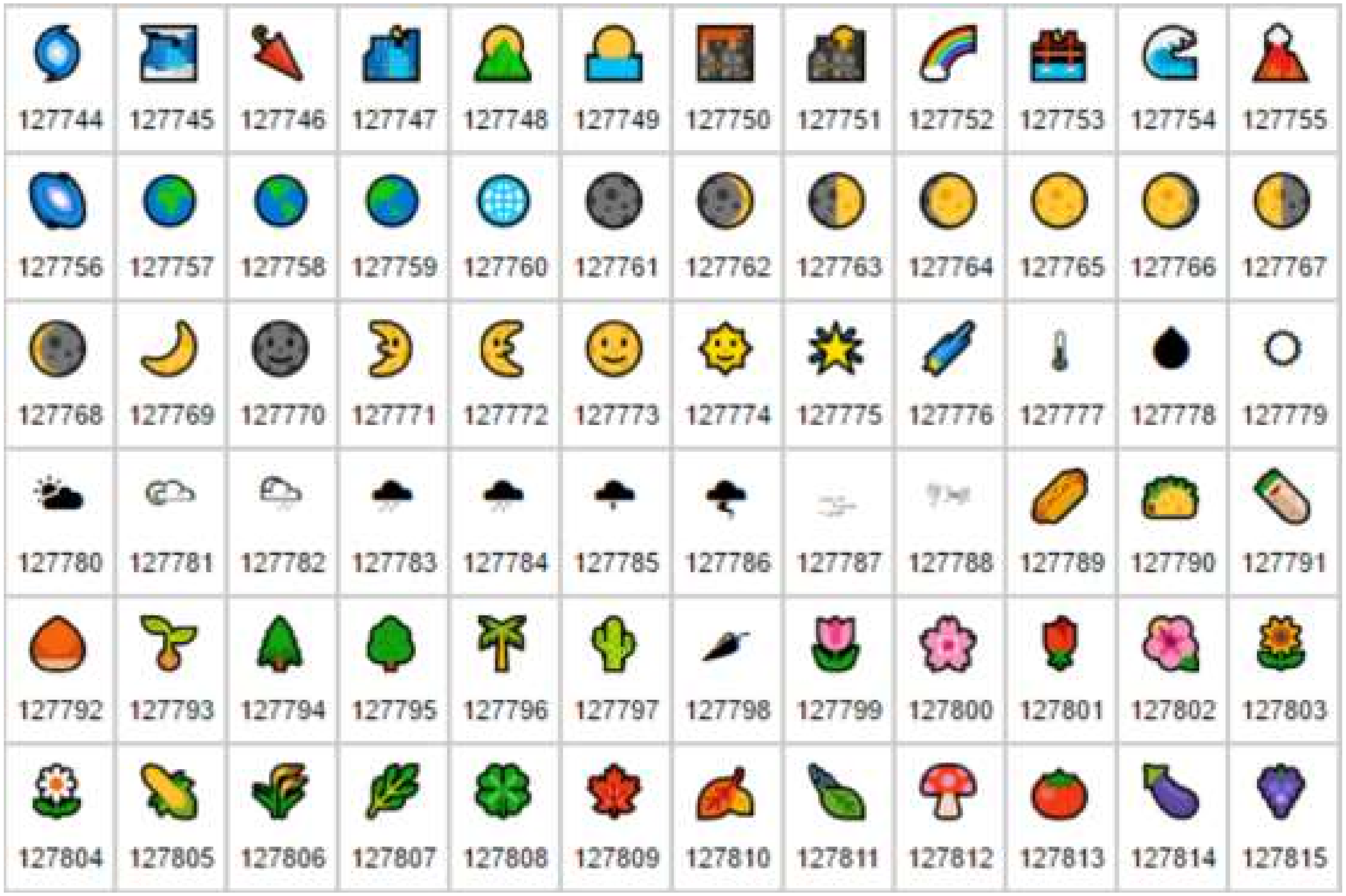


Codepage 819 - Latin 1 - ISO 8859-1

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-		0001	0002	0003	0004	0005	0006	0007	0008	0009	000A	000B	000C	000D	000E	000F
1-	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	001A	001B	001C	001D	001E	001F
2-	0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3-	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4-	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5-	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
6-	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7-	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	007F
8-	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	008A	008B	008C	008D	008E	008F
9-	0090	0091	0092	0093	0094	0095	0096	0097	0098	0099	009A	009B	009C	009D	009E	009F
A-	00A0	¡ 00A1	¢ 00A2	£ 00A3	¤ 00A4	¥ 00A5	¦ 00A6	§ 00A7	¨ 00A8	© 00A9	ª 00AA	« 00AB	¬ 00AC	- 00AD	® 00AE	¯ 00AF
B-	° 00B0	± 00B1	² 00B2	³ 00B3	´ 00B4	µ 00B5	¶ 00B6	· 00B7	¸ 00B8	¹ 00B9	º 00BA	» 00BB	¼ 00BC	½ 00BD	¾ 00BE	¿ 00BF
C-	À 00C0	Á 00C1	Â 00C2	Ã 00C3	Ä 00C4	Å 00C5	Æ 00C6	Ç 00C7	È 00C8	É 00C9	Ê 00CA	Ë 00CB	Ì 00CC	Í 00CD	Î 00CE	Ï 00CF
D-	Ð 00D0	Ñ 00D1	Ò 00D2	Ó 00D3	Ô 00D4	Õ 00D5	Ö 00D6	× 00D7	Ø 00D8	Ù 00D9	Ú 00DA	Û 00DB	Ü 00DC	Ý 00DD	Þ 00DE	ß 00DF
E-	à 00E0	á 00E1	â 00E2	ã 00E3	ä 00E4	å 00E5	æ 00E6	ç 00E7	è 00E8	é 00E9	ê 00EA	ë 00EB	ì 00EC	í 00ED	î 00EE	ï 00EF
F-	ð 00F0	ñ 00F1	ò 00F2	ó 00F3	ô 00F4	õ 00F5	ö 00F6	÷ 00F7	ø 00F8	ù 00F9	ú 00FA	û 00FB	ü 00FC	ý 00FD	þ 00FE	ÿ 00FF

Source: https://visual-integrity.com/iso-8859/

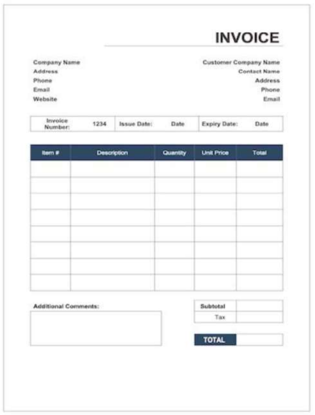
Character Encoding: Unicode (Sample)



🌀 127744	🌁 127745	🌂 127746	🌃 127747	🌄 127748	🌅 127749	🌆 127750	🌇 127751	🌈 127752	🌉 127753	🌊 127754	🌋 127755
🌌 127756	🌍 127757	🌎 127758	🌏 127759	🌐 127760	🌑 127761	🌒 127762	🌓 127763	🌔 127764	🌕 127765	🌖 127766	🌗 127767
🌘 127768	🌙 127769	🌚 127770	🌛 127771	🌜 127772	🌝 127773	🌞 127774	🌟 127775	🌠 127776	🌡 127777	🌢 127778	🌣 127779
🌤 127780	🌥 127781	🌦 127782	🌧 127783	🌨 127784	🌩 127785	🌪 127786	🌫 127787	🌬 127788	🌭 127789	🌮 127790	🌯 127791
🌰 127792	🌱 127793	🌲 127794	🌳 127795	🌴 127796	🌵 127797	🌶 127798	🌷 127799	🌸 127800	🌹 127801	🌺 127802	🌻 127803
🌼 127804	🌽 127805	🌾 127806	🌿 127807	🍀 127808	🍁 127809	🍂 127810	🍃 127811	🍄 127812	🍅 127813	🍆 127814	🍇 127815

Source: https://www.vertex42.com/ExcelTips/unicode-symbols.html

Textual Data Acquisition



INVOICE

Company Name
Address
Phone
Email
Website

Customer Company Name
Contact Name
Address
Phone
Email

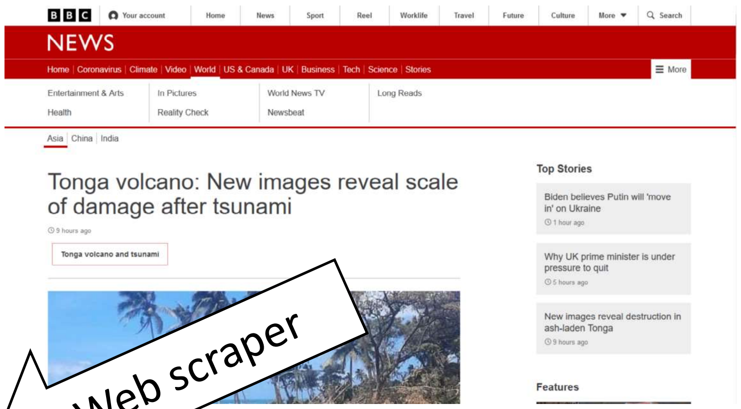
Invoice Number:	1234	Issue Date:	Date	Expiry Date:	Date

Item #	Description	Quantity	Unit Price	Total

Additional Comments:

Subtotal	
Tax	
TOTAL	

PDF document



BBC | Your account | Home | News | Sport | Reel | Worklife | Travel | Future | Culture | More | Search

NEWS

Home | Coronavirus | Climate | Video | World | US & Canada | UK | Business | Tech | Science | Stories | More

Entertainment & Arts | In Pictures | World News TV | Long Reads
Health | Reality Check | Newsbeat

Asia | China | India

Tonga volcano: New images reveal scale of damage after tsunami

9 hours ago

Tonga volcano and tsunami

Top Stories

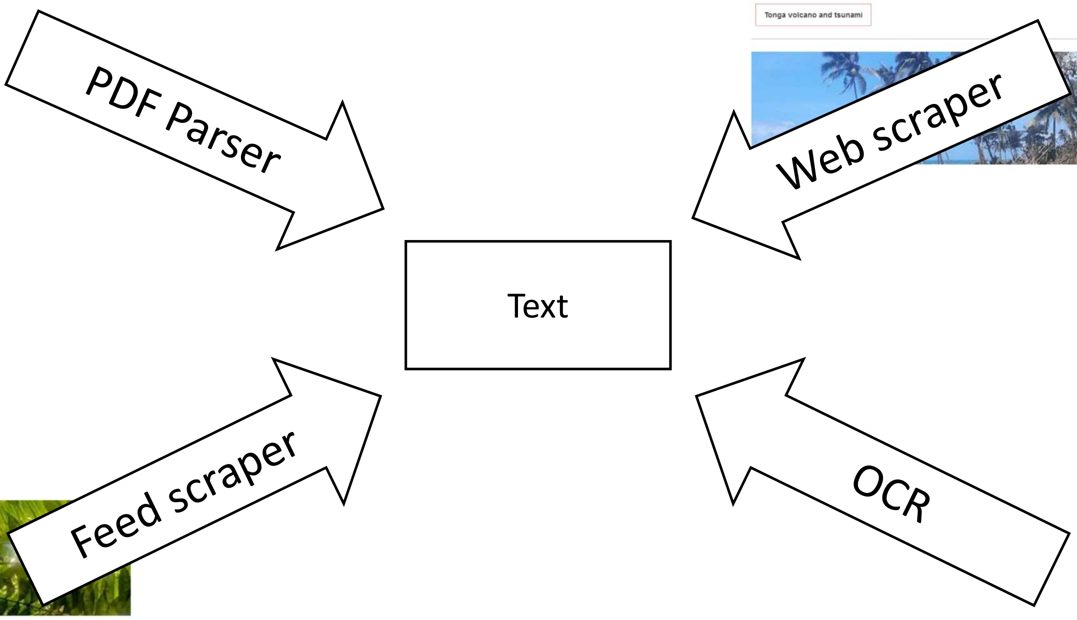
Biden believes Putin will 'move in' on Ukraine
1 hour ago

Why UK prime minister is under pressure to quit
5 hours ago

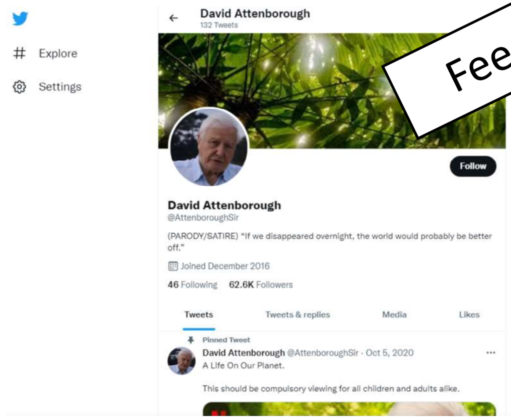
New images reveal destruction in ash-laden Tonga
9 hours ago

Features

Web page



Social media feed



Explore
Settings

David Attenborough
132 Tweets

Follow

David Attenborough
@AttenboroughSir

(PARODY/SATIRE) "If we disappeared overnight, the world would probably be better off."

Joined December 2016

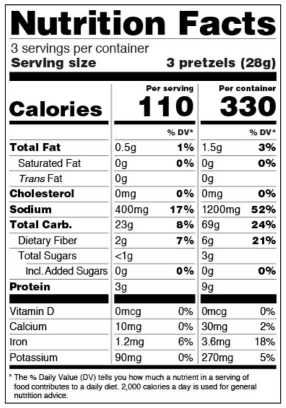
46 Following 62.6K Followers

Tweets | Tweets & replies | Media | Likes

Pinned Tweet
David Attenborough @AttenboroughSir · Oct 5, 2020
A Life On Our Planet.

This should be compulsory viewing for all children and adults alike.

Image with text



Nutrition Facts

3 servings per container
Serving size **3 pretzels (28g)**

	Per serving		Per container	
Calories	**110**		**330**	
		% DV*		% DV*
Total Fat	0.5g	**1%**	1.5g	**3%**
Saturated Fat	0g	**0%**	0g	**0%**
Trans Fat	0g		0g	
Cholesterol	0mg	**0%**	0mg	**0%**
Sodium	400mg	**17%**	1200mg	**52%**
Total Carb.	23g	**8%**	69g	**24%**
Dietary Fiber	2g	**7%**	6g	**21%**
Total Sugars	<1g		3g	
Incl. Added Sugars	0g	**0%**	0g	**0%**
Protein	3g		9g	
Vitamin D	0mcg	0%	0mcg	0%
Calcium	10mg	0%	30mg	2%
Iron	1.2mg	6%	3.6mg	18%
Potassium	90mg	0%	270mg	5%

* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

BTW:

PDF Parser:	https://py-pdf-parser.readthedocs.io/
Web scraping:	https://oxylabs.io/blog/python-web-scraping
Twitter scraper:	https://pypi.org/project/twitter-scraper/
OCR:	https://pypi.org/project/pytesseract/

some related features / tools present in NLP libraries

Pre-processing: Cleaning Text

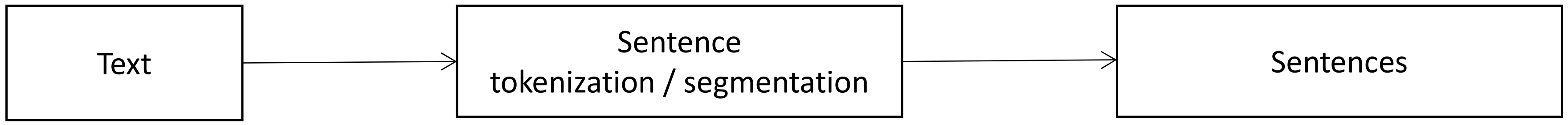
Some documents (email threads, comment sections, etc.) may require additional clean-up:

- special formatting and code:
  - special characters, HTML tags, hashtags
- salutations, signatures, addresses, etc.
  - example: polite phrases that are irrelevant to analysis
- replies
  - example: duplicate questions, etc.
- other

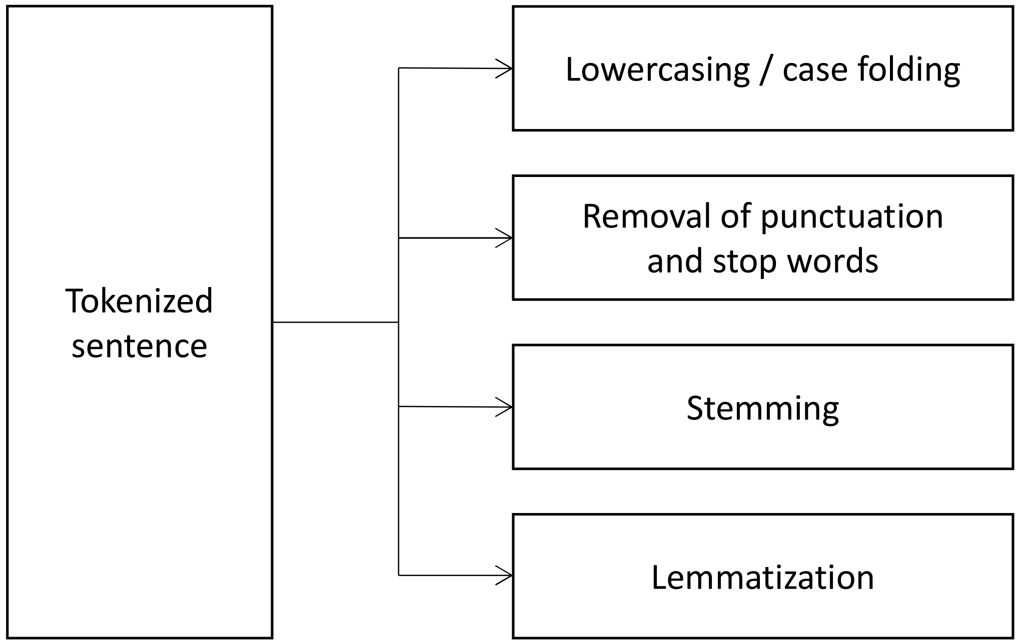
We will work with "clean" text only.

Basic Pre-Processing: Normalization

Document(s) / text level:



Tokenized sentence level:


Tokenized sentence
Lowercasing / case folding
Removal of punctuation and stop words
Stemming
Lemmatization

Note: depending on the nature of data, additional pre-processing steps may be required / important.

Pre-processing: Normalization

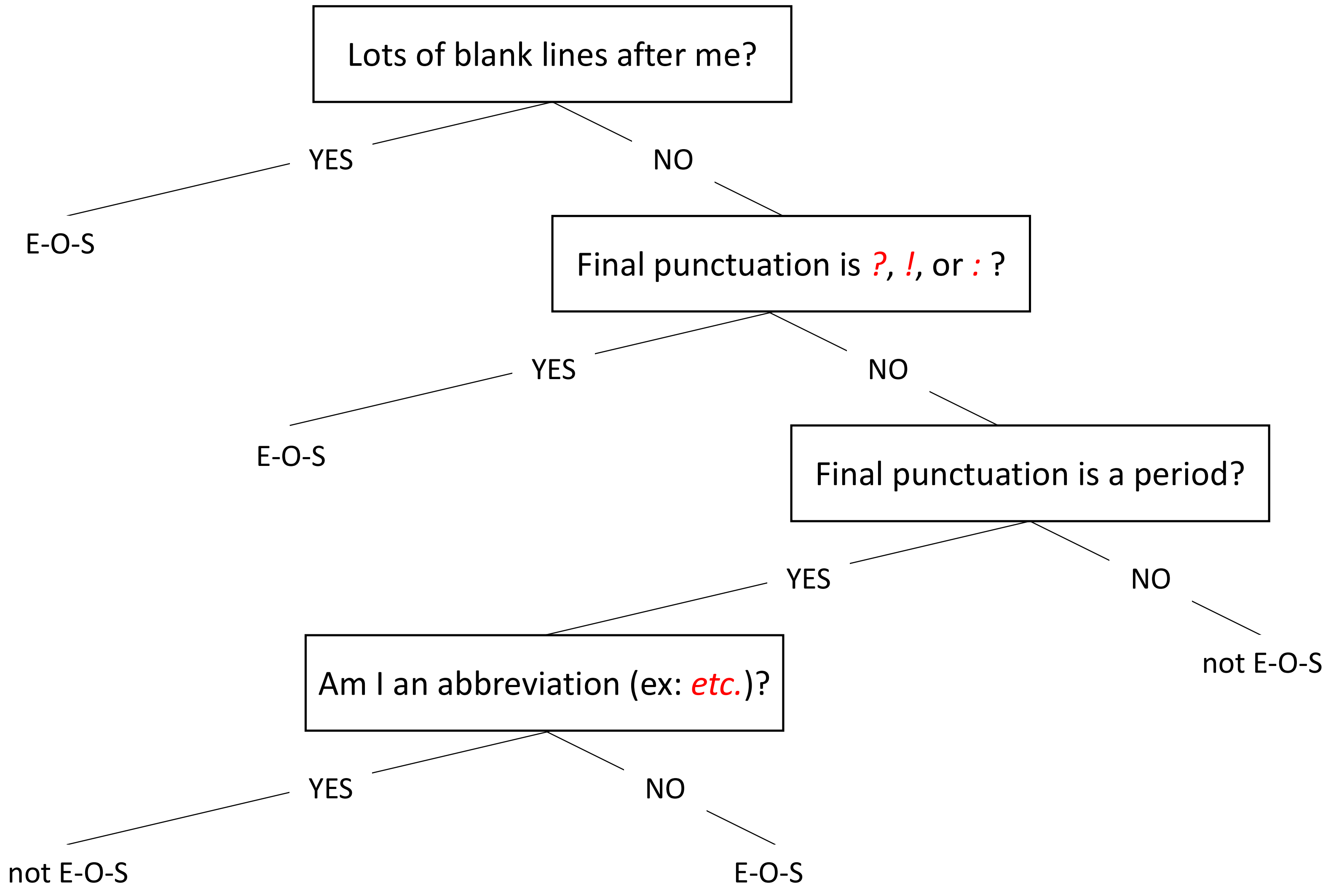
Every NLP task needs to do text normalization:

- Segmenting / tokenizing sentences in a document
- Segmenting / tokenizing words in sentences
- Normalizing word formats
  - lowercasing / case folding
  - stemming
  - lemmatization
  - etc.

Segmentation and Tokenization

- **Text segmentation (text into sentence): breaking up text into sentences at the appearance of full stops, exclamation and question marks.**
  - **abbreviations, forms of addresses (*Mrs.*), ellipses (*...*), numbers (*.02%*) are problematic**
    - **potential solution: build a "end of sentence" classifier**
  - **tools:** `.split()` **method, RegEx, NLP library specific**
- **Sentence tokenization (sentence to words): breaking up sentences into tokens based on the presence of whitespaces and punctuation marks (others possible).**
  - **tools:** `.split()` **method, RegEx, NLP library specific**

Segmentation with Decision Trees


Lots of blank lines after me?
YES
NO
E-O-S
Final punctuation is ?, !, or : ?
YES
NO
E-O-S
Final punctuation is a period?
YES
NO
not E-O-S
Am I an abbreviation (ex: *etc.*)?
YES
NO
not E-O-S
E-O-S

E-O-S: End-Of-Sentence

Tokenizaton: Type vs. Token

- **Type: an element of the vocabulary.**
- **Token: an instance of that type in runninga type in text.**

Vocabulary: a set of types

Example: *"A good course is a course that you like"*

- **9 tokens**
- **7 types (*a* and *course* are repeated)**

- **Type/token ratio (TTR): 7/9**

Tokenization: Problematic Cases

NLP applications should handle problematic cases during tokenization:

- **tokens containing periods:** *Dr.*, *xyz.com*
- **hyphens:** *rule-based*
- **clitics (connected word abbreviations):** *couldn't*, *we've*
- **numerical expressions and dates:** *(123) 555-5555*, *August 7th, 2019*
- **emojis, hashtags, email and web addresses (URLs)**

NLP libraries differ in this regard.

Tokenization: Problematic Cases

Finland's capital	$\rightarrow$	*Finland* **or** *Finlands* **or** *Finland's* **??**
what're, I'm, isn't	$\rightarrow$	*What are*, *I am*, *is not*
Hewlett-Packard	$\rightarrow$	*Hewlett Packard* **??**
state-of-the-art	$\rightarrow$	*state of the art* **??**
Lowercase	$\rightarrow$	*lower-case* **or** *lowercase* **or** *lower case* **??**
San Francisco	$\rightarrow$	**one token or two ??**
m.p.h., *PhD.*	$\rightarrow$	**??**

Other languages (German example):

- *Lebensversicherungsgesellschaftsangestellter* **is a noun compound for "***life insurance company employee***"**

Pre-processing: Lowercasing

Some applications (eg. Information Retrieval, search) reduce all letters to lower case:

- users tend to use lower case
- possible exception: upper case in mid-sentence?
  - General Motors
  - Fed vs. fed

For sentiment analysis, topic modeling:

- preserving case is important (US vs. us)

Pre-processing: Stemming

Stemming refers to the process of removing suffixes and reducing the word to some base form such that all different variants of that word can be represented by the same base form (*car* and *cars* are reduced to *car*).

- **use a set of rules to accomplish stemming**
  - **if the word ends in "*-es*", remove "*-es*"**
- **final base form may NOT be linguistically correct**
  - *airliner* $\rightarrow$ *airlin*
- **commonly used by search engines and in text classification**

Stemming: Before and After

Before:

For example compressed and compression are both accepted as equivalent to compress.

After:

For exampl compress and compress ar both accept as equival to compress.

Pre-processing: Lemmatization

Lemmatization is a process of mapping all the different forms of a word to its base word, or lemma. In other words: reduce inflections or variant forms to base form

- **has to find correct dictionary headword form**

Stemming vs. Lemmatization

Stemming	Lemmatization
adjustable → adjust	was → (to) be
	better → good
meeting → meet	meeting → meeting
studies → studi	studies → study
studying → study	studying → study

Tokenization / Lemmatization Example

Sentence input:

Chaplin wrote, directed, and composed music for most of his films.

Tokenization:

Chaplin	wrote	,	directed	,	and	composed	music	for	most	of	his	films	.

Chaplin wrote, directed, and composed music for most of his films.

Lemmatization:

Chaplin	write	,	direct	,	and	compose	music	for	most	of	he	film	.

Chaplin wrote, directed, and composed music for most of his films.

Pre-processing: Stop Words

Very common words (articles, propositions, pronouns, conjunctions,etc.) that do not add much information (but take up space) are called stop words and are frequently filtered out.

- **Examples in English:** *an*, *the*, *a*, *for*, *is*
- **Filtering based on the stop (word) list**
  - **generated based on collection frequency**
- **Tools: RegEx + stop list, NLP libraries have their own stop lists**
- **Careful: sometimes it may lead to removing important information**

Additional Pre-processing Steps

- **Additional normalization**
  - **in addition to stemming, lemmatization:**
    - **standardizing abbreviations (eg. expanding), hyphenations, digits to text (9 to nine) conversions, etc.**
- **Language detection**
- **Code mixing**
  - **embedding of linguistic units such as phrases, words, and morphemes of one language into an utterance of another language**
- **Transliteration**
  - **converting between different writing systems**

Regular Expressions (RegEx)

A regular expression (RegEx, regex or regexp) is a sequence of characters that specifies a search pattern in text.

- patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation.
- a technique developed in theoretical computer science and formal language theory. Related to the concept of a regular language (a formal language that can be defined by a regular expression).
- very efficient for pre-processing tasks
- there are two key RegEx libraries:
  - re (built-in)
  - and regex (external: https://pypi.org/project/regex/)

Regular Expressions: Disjunction

- **Disjunction**: characters within square brackets []

Pattern	Matches
`[wW]oodchuck`	**W**oodchuck **OR** **w**oodchuck
`[1234567890]`	Any digit (1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or 9 or 0)

- **Disjunction**: a **range** of characters, square brackets [] and dash -

Pattern	Matches	Example
`[A-Z]`	An upper case letter	**D**renched Blossoms
`[a-z]`	A lower case letter	**m**y beans were impatient
`[0-9]`	A single digit	Chapter **1**: Down the Rabbit Hole

Regular Expressions: Disjunction

- **Disjunction**: a pipe | also means OR

Pattern	Matches
`woodchuck|groundhog`	woodchuck **OR** groundhog
`yours|mine`	yours **OR** mine
`a|b|c`	same as `[abc]` or `[a-c]`

Regular Expressions: Negation

- **Negation**: a caret ^ means negation (has to be first within [])

Pattern	Matches	Example
`[^A-Z]`	NOT an upper case letter	Oyfn pripetchik
`[^Ss]`	Neither 'S' nor 's'	I have no reason for it

Regular Expressions: Optionality

- **Optionality**: the question mark ? marks optionality of previous

Pattern	Matches	Example
`woodchucks?`	woodchuck OR woodchucks	nice woodchuck!
`colou?r`	color OR colour	beautiful colour

Regular Expressions: . wildcard

- **. wildcard**: period . represents ANY character

Pattern	**Matches**	**Example**
`beg.n`	any character between *beg* and *n*	begin beg'n begun

Regular Expressions: Other Operators

- **Some additional and useful operators:**

Operator	Expansion	Match	Examples
`\d`	`[0-9]`	any digit	Party of 5
`\D`	`[^0-9]`	any non-digit	Blue moon
`\w`	`[a-zA-Z0-9_]`	any alphanumeric / underscore	Daiyu
`\W`	`[^\w]`	a non-alphanumeric	!!!!!!
`\s`	`[ \r\t\n\f]`	whitespace (space, tab)	
`\S`	`[^\s]`	non-whitespace	in Chicago

Regular Expressions: Backslash

- **Some characters need to be backlashed (operators in RegEx)**

Pattern	Matches	Comment
`\*`	an asterisk *	K*A*P*L*A*N
`\.`	a period .	Dr. Livingston, I presume
`\?`	a question mark	What is the time?
`\n`	a newline character	
`\t`	a tab	

Regular Expressions: Anchors

- **Anchors**: anchor regular explations to specific places in a string

Pattern	Matches	Comment
`^[A-Z]`	Palo Alto	Start of string anchor (^) **First** character has to be uppercase letter
`^[^A-Za-z]`	1 “Hello”	Start of string anchor (^) **First** character cannot be a letter
`\.$`	The end.	End of string anchor ($) Note the \ before . **Last** character has to be .
`.$`	The end? The end!	End of string anchor ($) **Last** character can be anything
`\b`		word boundary
`\B`		word non-boundary

Regular Expressions: Example

Find an instance of the word 'the' within input string.

RegEx patterns:

- `the` **: will miss capitalized 'The'**
- `[tT]he` **: will match substrings 'the' and 'The' within other words (o*the*r, *the*m)**
- `[^a-zA-Z][tT]he[^a-zA-Z]` **: this will do it**

Fixed two type of errors (to increase precision and recall):

- **Type I: matching strings we shouldn't have (false positive)**
- **Type II: not matching strings we should have matched (false negative)**

Python re Module / Library

Python's re module / library is built-in. Documentation:

`https://docs.python.org/3/library/re.html`

Key functions / methods:

- `match()`**: checks for matching string at the beginning**
- `search()`**: find first location of a matching string**
- `findall()`**: returns all non-overlapping matches**
- `split()`**: splits string by occurences of a pattern**
- `sub()`**: "replace"**

Python NLP Libraries / Packages

- Natural Language Toolkit (NLTK) [more academic]
- TextBlob
- CoreNLP
- Gensim
- spaCy [industry / production]
- Polyglot
- scikit-learn (machine learning)
- pyTorch (machine learning)
- Pattern
- PyNLPl

Natural Language Toolkit (NLTK)

“NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.”

Link:	**https://www.nltk.org/**
Anaconda:	**https://anaconda.org/anaconda/nltk**
Install:	**https://www.nltk.org/install.html**

TextBlob

“TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.”

Link:	**https://textblob.readthedocs.io/en/dev/**
Anaconda:	**https://anaconda.org/conda-forge/textblob**
Install:	**https://textblob.readthedocs.io/en/dev/install.html**

CoreNLP

"CoreNLP is your one stop shop for natural language processing in Java! CoreNLP enables users to derive linguistic annotations for text, including token and sentence boundaries, parts of speech, named entities, numeric and time values, dependency and constituency parses, coreference, sentiment, quote attributions, and relations. CoreNLP currently supports 8 languages: Arabic, Chinese, English, French, German, Hungarian, Italian, and Spanish."

Link:	https://stanfordnlp.github.io/CoreNLP/
Anaconda:	https://anaconda.org/auto/corenlp

Gensim

“Gensim is a Python library for topic modelling, document indexing and similarity retrieval with large corpora. Target audience is the natural language processing (NLP) and information retrieval (IR) community.”

Link:	**https://github.com/RaRe-Technologies/gensim**
Anaconda:	**https://anaconda.org/anaconda/gensim**
Install:	**https://github.com/RaRe-Technologies/gensim**

spaCy

“spaCy is a free, open-source library for advanced Natural Language Processing (NLP) in Python.”

Link:	**https://spacy.io/**
Anaconda:	**https://anaconda.org/conda-forge/spacy**
Install:	**https://spacy.io/usage**

Polyglot

"Polyglot is a natural language pipeline that supports massive multilingual applications."

Link: https://polyglot.readthedocs.io/en/latest/index.html

Anaconda: https://anaconda.org/syllabs_admin/polyglot

Install: https://polyglot.readthedocs.io/en/latest/Installation.html

scikit-learn

“Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.”

Link:	https://scikit-learn.org/stable/index.html
Anaconda:	https://anaconda.org/anaconda/scikit-learn
Install:	https://scikit-learn.org/stable/install.html

Pattern

“Web mining module for Python, with tools for scraping, natural language processing, machine learning, network analysis and visualization.”

Link:	**https://github.com/clips/pattern**
Anaconda:	**https://anaconda.org/conda-forge/pattern**
Install:	**https://github.com/clips/pattern**

PyNLPl

“PyNLPl (Python Natural Language Processing library), pronounced as 'pineapple', is a Python library for Natural Language Processing. It contains various modules useful for common, and less common, NLP tasks.”

Link:	https://github.com/proycon/pynlpl
Anaconda:	N/A?
Install:	https://github.com/proycon/pynlpl

Text Corpora

In linguistics, a **corpus** (plural **corpora**) or **text corpus** is a language resource consisting of a large and structured set of texts (nowadays usually electronically stored and processed).