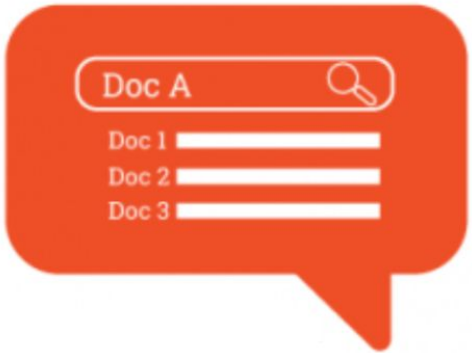
Introduction to Natural Language Processing

Georgeta Bordea
georgeta.bordea@u-bordeaux.fr

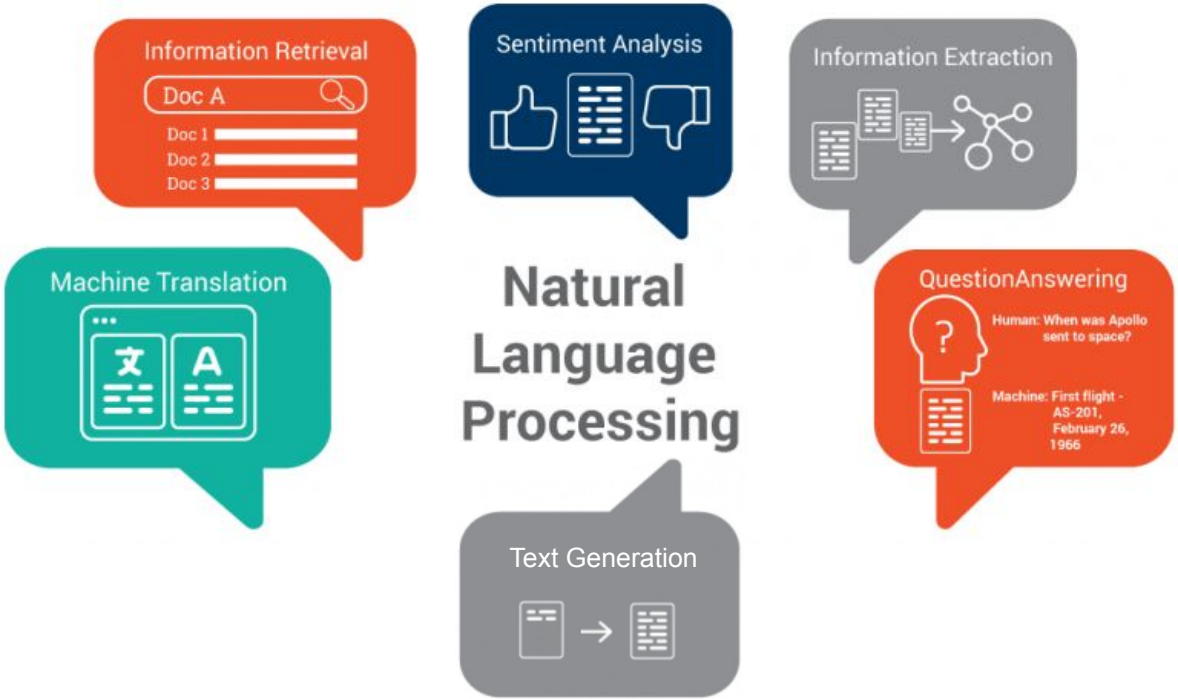
November 2023

Motivation

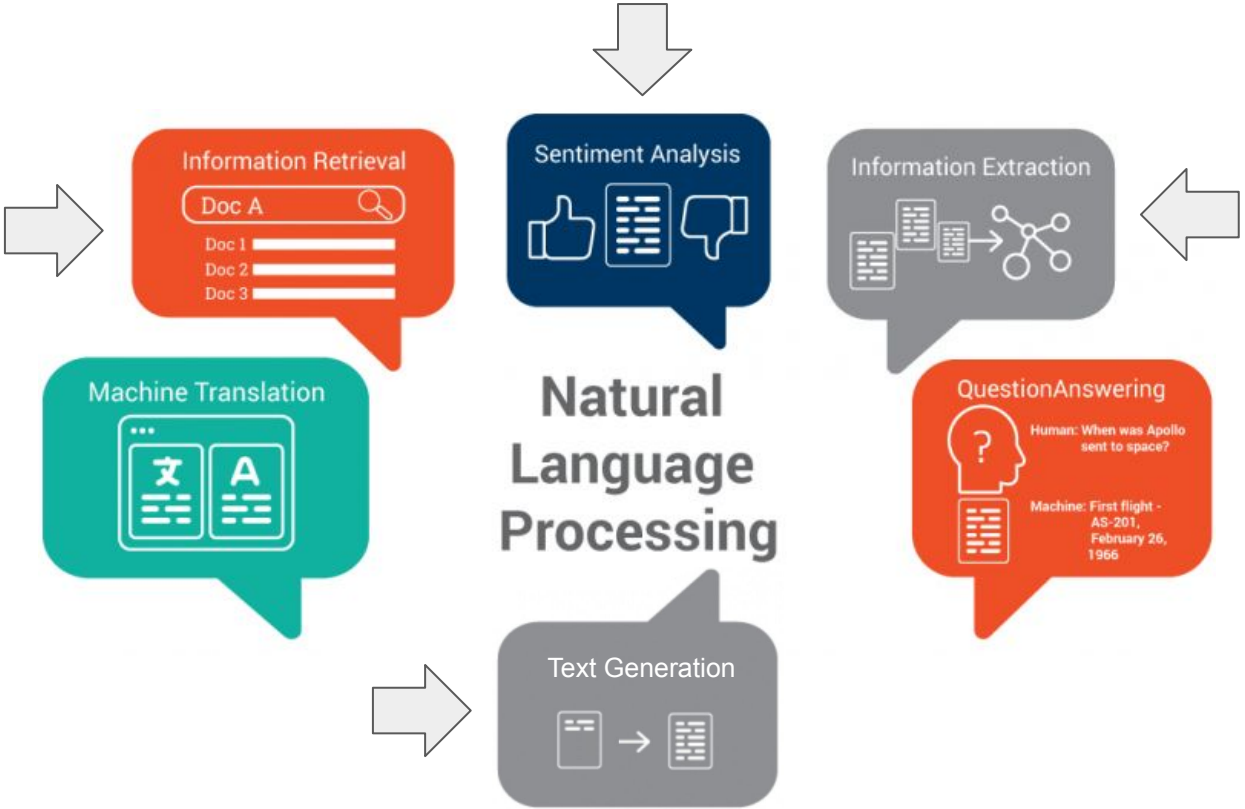
- Natural Language Processing (NLP) is the discipline concerned with developing systems which **process**, **understand** and **generate** language
- NLP uses in Information Retrieval
  - Search engines need to process and understand keyword queries
  - Search engines have to efficiently represent and retrieve documents
- Limitations
  - Traditional IR relies on word matching
  - There are two fundamental query matching problems:
    - Synonymy (image: likeness, portrait, facsimile, icon)
    - Polysemy (port: harbor, fortified wine, computer jack)



Natural Language Processing Applications

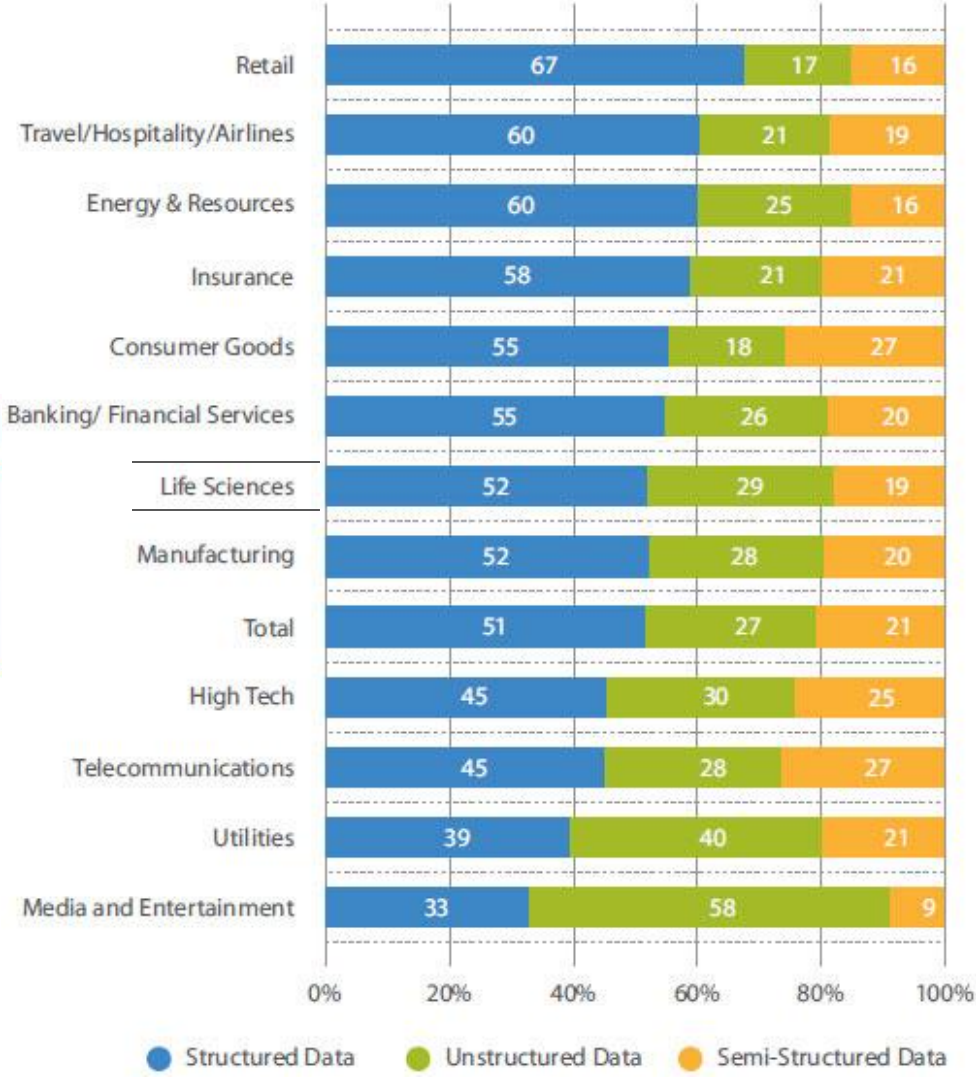


Natural Language Processing Applications



Information Extraction

Structured data	Semi-structured data	Unstructured data
Databases	XML / JSON data	Audio
	Email	Video
	Web pages	Image data
		Natural language
		Documents



Statistical vs. rule-based NLP

Statistical	Rule-based
Learn rules from data	Write rules by hand
High error rate	Low error rate
Good coverage	Poor coverage
Intensive algorithms	Intensive manual work
New understanding of language	Requires full understanding

Data

- Data is a prerequisite for statistical processing
- Types of data:
  - Corpus
    - Plain text
    - Tokenised
    - Tags, trees and graphs
  - Lexicon
    - Word lists
    - Inflections
    - Sense, semantic relations
  - Language data
    - Knowledge about language
    - Grammatical rules

Corpora

- A corpus is a collection of texts and the main item of study for statistical NLP
- Normally has some structure:
  - Documents, Sentence alignment, Character-level annotation, Token-level annotation, Inter-token annotation, Sentence annotation
- Types of corpora
  - Text corpora (British National Corpus), simple annotated corpora (Brown Corpus), complex annotated corpora (Penn TreeBank), parallel corpora (EuroParl)
- Other corpora like resources
  - Google n-gram corpus
- Domain specific corpora
  - Biomedical text (GENIA corpus)
  - Clinical text

Initial stages of text processing

- Tokenisation
  - Cut character sequence into word tokens
- Normalisation
  - Map text and query term to same form
  - Match *U.S.A.* and *USA*
  - User generated content: *Cuz tweets R haaaard!!!!1!! :)*
- Stemming
  - We may wish different forms of a root to match
  - *Authorise*, *authorisation*
- Stop words
  - We may omit very common words (or not)
  - the, a, to, of

Tokenisation

- The tokenisation tasks consists in extracting the tokens from a text
- For example, given the following sentence:

In logic, computational linguistics, and information retrieval, a token is an instance of a type.

- We would like to extract the following list of tokens:

In | *logic* | , | *computational* | *linguistics* | , | *and* | *information* | *retrieval* | , | *a* | *token* | *is* | *an* | *instance* | *of* | *a* | *type* | .

- More difficult for other languages

Difficult cases for tokenisation

- Trivial attempt is to split using whitespace characters

The “quote here” doesn’t tokenize well.

The | “quote | here” | doesn’t | tokenize | well.

- For languages without definite procedures for splitting words
  - Compile n-grams from a corpus with token annotations
  - Insert word breaks if the probability > 0.5

Part-of-Speech (POS) tagging

- The task of part-of-speech tagging consists in assigning the right part-of-speech to every token.
- Typically the following POS are used for Indo-European languages: Noun, Verb, Adjective, Adverb, Pronoun, Preposition, Article, Conjunction
- For example:

In	*logic*	*,*	*computational*	*linguistics*	*,*	*and*	*information*	*retrieval*	*,*
Prep	**Noun**	**Punc**	**Adj**	**Noun**	**Punc**	**Conj**	**Noun**	**Noun**	**Punc**
a	*token*	*is*	*an*	*instance*	*of*	*a*	*type*		
Det	**Noun**	**Verb**	**Det**	**Noun**	**Prep**	**Det**	**Noun**		

Parsing

- Parsing is the task of computing a phrase tree for a given sentence.
- For example, for the following sentence, a parser would produce as output:

The cat chases a mouse

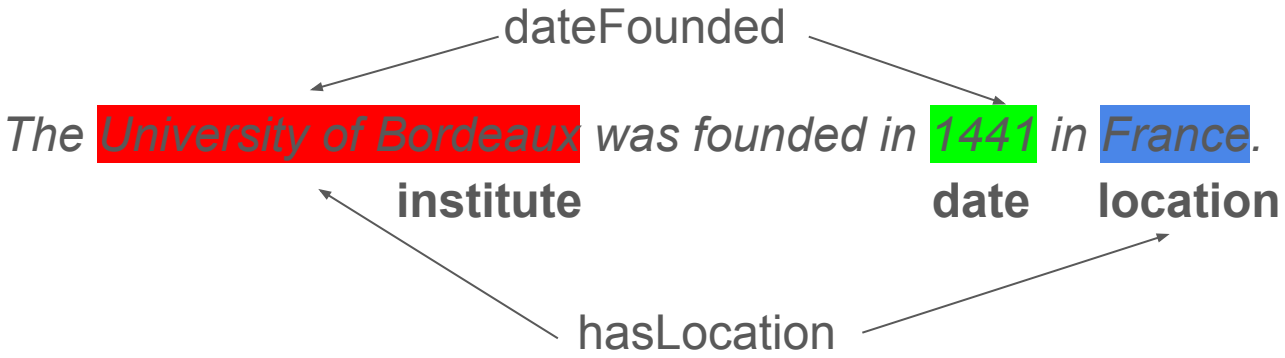
```
(S
      (NP
            (DET the) cat)
      (VP chases
            (NP (DET the) mouse)))
```

Information Extraction

- Information Extraction (IE) systems:
  - Find and understand limited relevant parts of texts (Named Entities)
  - Gather information from many pieces of text
  - Produce a structured representation of relevant information (Relations, Events)
- Goals:
  - Organize information so that it is useful to people
  - Put information in a semantically precise form that allows further inferences to be made by computer algorithms

Information Extraction example

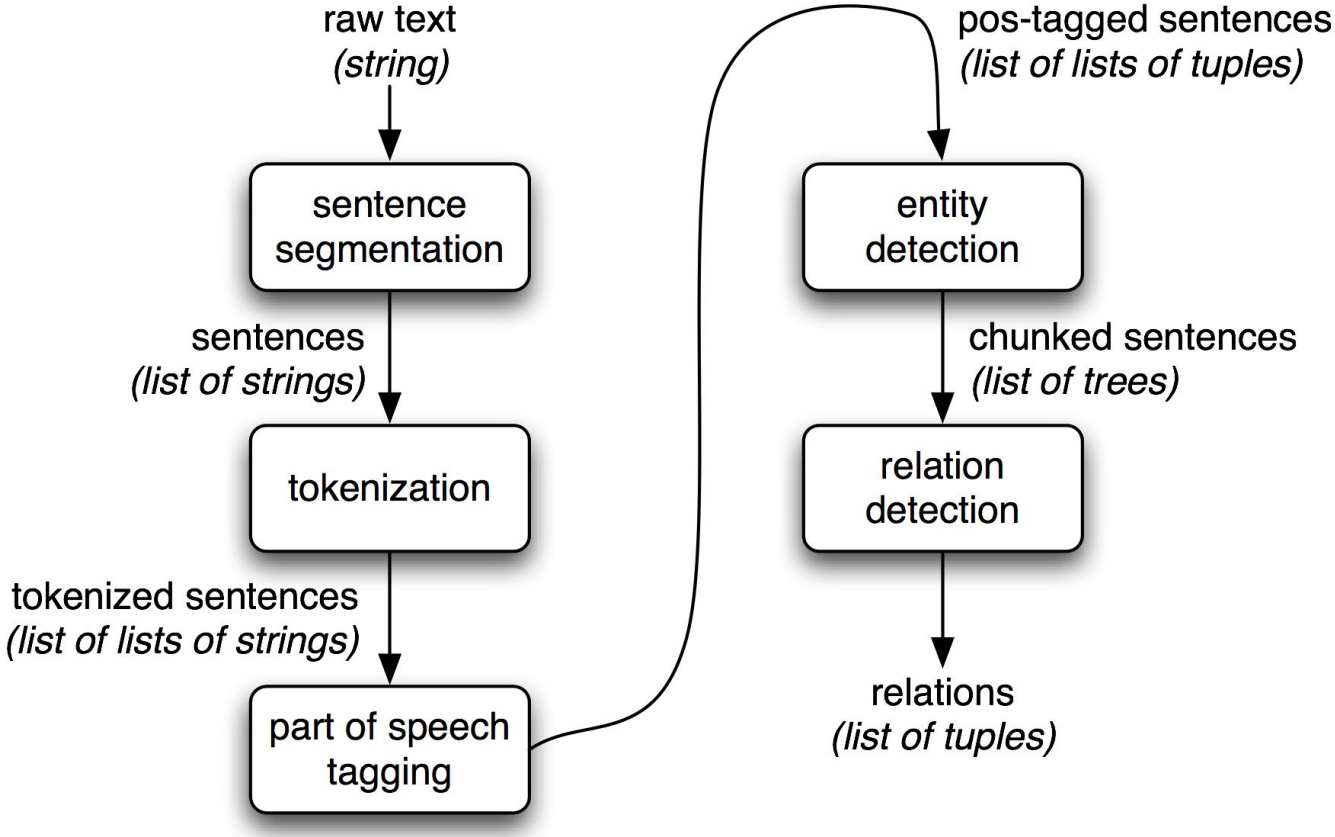
- The task of extracting structured information (facts) from unstructured information (text)



Information Extraction tasks

- Named Entity Recognition
  - Diseases, drugs, people names, symptoms, human anatomy
- Co-reference Resolution
- Relation Extraction
  - Protein-Protein interactions, Gene-Gene interactions, Drug-Drug interactions
  - Extraction of adverse effects
- Event Extraction
  - Protein biology
- Biomed IE tasks
  - Automatic assignment of ICD10 codes to clinical text
  - De-identification of discharge summaries
  - Patient smoking status discovery from discharge summaries

Architecture of an Information Extraction system



Semantic relations

- **Synonymy** Denotes the same as this entry (skin - tegument)
- **Antonymy** Denotes the opposite of this entry (up - down)
- **Hypernymy** Each listed hypernym is superordinate to this entry (liver - organ)
- **Hyponymy** Each listed hyponym is subordinate to this entry (organ - liver)
- **Meronymy** Denotes part of this entry's referent (body - arm)
- **Holonymy** Each listed holonym has this entry's referent as a part of itself (arm - body)
- **Coordinate term** Term that shares a hypernym with this entry (liver - lung)
- **Otherwise related** Each listed term semantically relates to this entry (anatomy - body)

WordNet

WordNet Search - 3.1

- WordNet home page - Glossary - Help

Word to search for: skin Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **skin**, tegument, cutis (a natural protective body covering and site of the sense of touch) *"your skin is the largest organ of your body"*
- S: (n) **skin** (an outer surface (usually thin)) *"the skin of an airplane"*
- S: (n) hide, pelt, **skin** (body covering of a living animal)
- S: (n) **skin** (a person's skin regarded as their life) *"he tried to save his skin"*
- S: (n) skinhead, **skin** (a member of any of several British or American groups consisting predominantly of young people who shave their heads; some engage in white supremacist and anti-immigrant activities and this leads to the perception that all skinheads are racist and violent)
- S: (n) baldhead, baldpate, baldy, skinhead, **skin** (a person whose head is bald or shaved)
- S: (n) peel, **skin** (the rind of a fruit or vegetable)
- S: (n) **skin** (a bag serving as a container for liquids; it is made from the hide of an animal)

Similarity measures

- Dice coefficient $S_{Dice}(x,y)$

$$\frac{2\,|\,x \cap y\,|}{|\,x\,| + |\,y\,|}$$

- Jaccard index $S_{Jaccard}(x,y)$

$$\frac{|\,x \cap y\,|}{|\,x\,| + |\,y\,| - |\,x \cap y\,|}$$

- Cosine similarity $S_{Cosine}(x,y)$

$$\frac{|\,x \cap y\,|}{\sqrt{|\,x\,||\,y\,|}}$$

Relation extraction approaches

- Using patterns to extract relations
  - **Examples** *animals such as lions*, *animals including lions*
  - **Disadvantage** Low recall, difficult to gather for each type of relation
- Supervised relation extraction
  - **Examples** MaxEnt, Naive Bayes, SVM, etc. with
  - Entity-based / Word-based / Syntactic Features
  - **Disadvantage** Require large amount of annotated text as training data
- Semi-supervised relation extraction
- Distant supervision
- Unsupervised relation extraction

Bootstrapping approaches to relation extraction

- Bootstrapping algorithm
  - Gather a set of seed pairs that have relation R
  - Iterate:
    - 1. Find sentences with these pairs
    - 2. Look at the context between or around the pair and generalise the context to create patterns
    - 3. Use the patterns to search for more pairs
- Example
  - Seed pair: <Mark Twain, Elmira>
  - Found sentences: *Mark Twain is buried in Elmira*; *The grave of Mark Twain is in Elmira*
  - Generalised patterns: ***X is buried in Y***; ***the grave of X is in Y***

Relation extraction with distant supervision

- Use a large database with many seeds to create features for a classifier
  - 1. For each relation
  - 2. For each tuple in the database
  - 3. Find sentences in a large corpus that contain both entities
  - 4. Extract frequent features
  - 5. Train supervised classifier
- Example
  - Relation: born_in
  - Database tuples: ***(Edwin Hubble, Marshfield), (Albert Einstein, Ulm)***
  - Features: **PER was born in LOC**, **PER's birthplace in LOC**

Unsupervised Relation Extraction

- Open Information Extraction from the web, without training data or predefined relations
  - 1. Use a small amount of parsed data to train a trustworthy tuple classifier
  - 2. Single-pass the web corpus to extract all relations between NPs, keep if trustworthy
  - 3. An assessor ranks relations based on text redundancy
- Example

  (FCI, specialises in, software development)

  (Tesla, invented, coil transformer)

Information Extraction evaluation

- Evaluation is performed per task
- Expected output can be precisely defined
- Evaluation in terms of Precision, Recall, Fscore (weighted harmonic mean)

$$P = \frac{\#\ correctly\ extracted\ items}{Total\ \#\ of\ extracted\ items} \quad (1)$$

$$R = \frac{\#\ correctly\ extracted\ items}{Total\ \#\ of\ gold\ items} \quad (2)$$

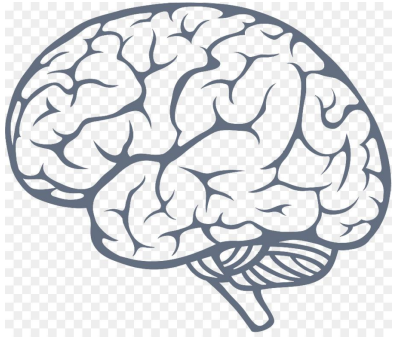
$$F = \frac{2 \times P \times R}{P + R} \quad (3)$$

Information Extraction performance

- Performance depends on:
  - Complexity of relations/events
  - Nature and length of text
  - Domain and language
- Message Understanding Conference (MUC-7) scores were typically:
  - 95% for NER
  - 60-80% for co-reference
  - 70-85% for relation extraction
  - 50-70% for event extraction

Regular expressions

- A formal language for specifying text strings
- How can we search for any of these?
  - *brain*
  - *brains*
  - *Brain*
  - *Brains*



https://www.debuggex.com/cheatsheet/regex/python

https://www.nltk.org/book/ch03.html#sec-regular-expressions-word-patterns

Regular expressions: Disjunctions

- Letters inside square brackets []

  [Bb]rain → *brain, Brain*

  [1234567890] → *Any digit*

- Ranges

  [A-Z] → *An upper case letter*

  [a-z] → *A lower case letter*

  [0-9] → *A single digit*

Regular Expressions: Negation in disjunction

- Negations [^Ss], only when the first character in the disjunction

[^A-Z]	→	*Not an upper case letter*
[^Ss]	→	*Neither S nor s*
[^e^]	→	*Neither e nor the literal ^*

Regular expressions: More disjunction

- *Encephalon* is the medical term for the brain
- The pipe | is also used for disjunction

  brain | *encephalon*

  a|*b*|*c* = *[abc]*

  [Bb]rain | *[Ee]ncephalon*

Regular Expressions: ? * + .

colou?r	→	*Optional previous character*	*color colour*
*oo*h!*	→	*Zero or more*	*oh! ooh! Oooh!*
o+h!	→	*One or more*	*oh! ooh! Oooh!*
beg.n	→	*Any character*	*begin begun beg3n*

Regular expressions: Hypernym/hyponym extraction

[Hearst1992]

NP↑ such as NP {, NP}* {(and|or) NP}

such NP↑ as {NP,}* {(or|and)} NP

NP {, NP}* {,} or other NP↑

NP {, NP}* {,} and other NP↑

NP↑ {,} including {NP, }* {(or|and)} NP

NP↑ {,} especially {NP, }* {(or|and)} NP

Next courses

- 15 November: Statistical Natural Language Processing
- 15 December: Natural Language Processing using Deep Learning

NLP tools

- R with the TM package
- Java (or Scala, Clojure, etc.) with OpenNLP
- Python NLP tools
  - spaCy, Gensim, Core NLP, Pattern, Polyglot, Text Blob, AllenNLP, Flair
  - Hugging Face Transformers
- Python NLTK
  - matplotlib: for drawing graphs
  - scipy / numpy: mathematics
  - scikit-learn: machine learning

Hands on with Natural Language Toolkit (NLTK)

1. Read the Jupyter notebook

   jupyter lab

2. Follow and run examples
3. Implement and run exercises
4. Check solutions

Bibliography

[Hearst 1992] Marti Hearst, “Automatic acquisition of hyponyms from large text corpora” (1992)

Christopher D. Manning and Hinrich Schütze, “Foundations of Statistical Natural Language Processing”

Steven Bird, Ewan Klein, and Edward Loper, “Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit”
https://www.nltk.org/book/

10 Best Python Libraries for NLP
https://www.qblocks.cloud/blog/best-nlp-libraries-python