# Natural Language Processing

## Lecture 1
## Introduction to NLP
## Methods of Morphological Analysis

# Course program and requirements

12 classes:

- 10 classes - lectures;
- 2 classes - reports;

\+ some practical tasks

= attestation

Languages:

- Natural languages: *English*, *Chinese*, *Russian* etc.;
- Formal languages: *programming languages* etc.;
- Artificial languages: *Esperanto*, *Elvish languages* etc.

# Natural Language Processing

**Natural-language processing** (**NLP**) is a field of computer science, artificial intelligence concerned with the interactions between computers and human (natural) languages.

- ***1950*** - Turing test
- ***1954*** - Georgetown experiment (Machine Translation)
- ***1970s*** - conceptual ontologies
- ***1980s*** - ***1990s*** - statistical revolution
- ***Currently*** - Deep Learning algorithms

# Computer Linguistics Tasks:

1. Information Retrieval: *Google*, *Yahoo!*;
2. Information Extraction: *RCO Fact Extractor*;
3. Machine Translation: *PROMT*, *Google Translate*;
4. Automatic Text Summarization: *TextAnalyst*, *Extractor*, *Text Miner*;
5. Corpus Linguistics: *RusCorpora*, *OpenCorpora*;
6. Expert Systems: *IBM Watson*, *Wolfram Alfa*;
7. Question Answering Systems: *IBM Watson*, *Siri*;
8. Electronic dictionaries, thesaurus, onthology creation;
9. Optical Character Recognition: *Fine Reader*;
10. Automatic Speech Recognition: *plug-in in Google Chrome*;
11. Text-To-Speech: *Google Translate*

# Stages to build NLP system:

1. Analysis of graphemes (character level);
2. Morphological analysis (word level);
3. Fragmentational analysis (phrase level);
4. Syntax analysis (sentence level);
5. Semantic analysis (text level).

Discourse analysis - ?

# Analysis of graphemes: Tokenization

**Tokenization** is words, digits, punctuation marks, formula etc. extraction from the text.

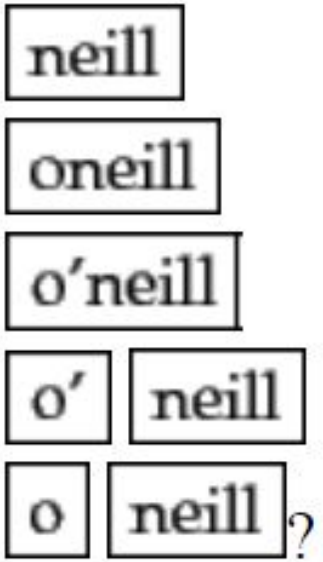
**Tokens** are elements extracted from the text.

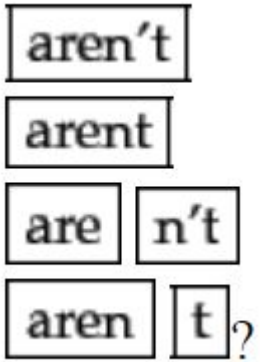
Input: Friends, Romans, Countrymen, lend me your ears;

Output: | Friends | | Romans | | Countrymen | | lend | | me | | your | | ears |

# Tokenization: tricky cases

Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing.



# Tokenization: tricky cases

1. **Programming Languages**: C++, C#;
2. **Aircraft names**: B-52;
3. **Email addresses**: jblack@mail.yahoo.com;
4. **Web URLs**: http://stuff.big.com/new/specials.html;
5. **Numeric IP addresses**: 142.32.48.231;
6. **Package tracking numbers**: (1Z9999W99845399981)
7. and more...

# Tokenization: hyphenation

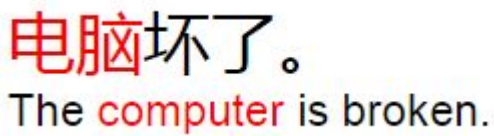
Example 1: *co-education*

Example 2: *Hewlett-Packard*

Example 3: *the hold-him-back-and-drag-him-away maneuver*

# Tokenization: other languages

- French: *l'ensemble*, *donne-moi* ‘give me’;
- German: *Computerlinguistik* `computational linguistics'; *Lebensversicherungsgesellschaftsangestellter* `life insurance company employee'
- East Asian Languages (e.g., Chinese, Japanese, Korean, and Thai)



# Analysis of graphemes: Segmentation

**Segmentation** is the retrieval of words boundaries in the text without spaces (e.g. Chinese or Japanese texts).

Example: *Itiseasytoreadtextwithoutspaces - It is easy to read text without spaces*

Possible solutions and important notes:

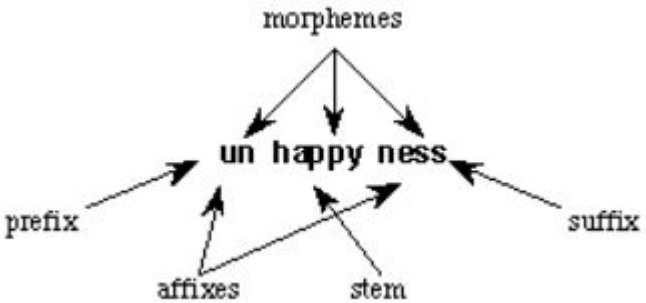
- statistical decision-making;
- large dictionaries;
- consideration of syntax and semantic constraints;
- taking into account specific domain

# Morphological Analysis

# Morphology

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

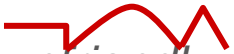
Its most important unit is the **morpheme**, which is defined as the "minimal unit of meaning".




**Free morpheme** can appear on its own
**Bound morphemes** have to be attached to a free morpheme

**Morpheme** is a minimal meaningful unit of a word.

**Root** is a morpheme with lexical meaning of a word. *unfriendly*



**Affix** is a morpheme which modifies the lexical meaning of a word (e.g. prefix, suffix).

**Allomorph** is some complementary **morphs** (the phonetic realization of morpheme), which manifest a morpheme in its different morphological or phonological environments.

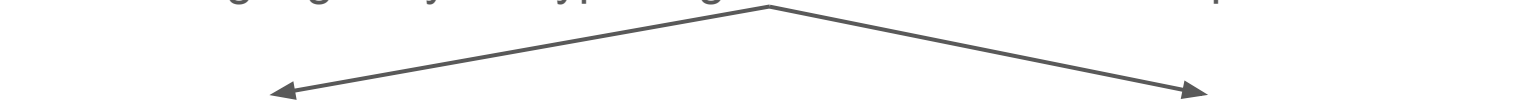
Lexemes: *illegal*, *impatient*, *irregular*, *inconsiderate*

Allomorphs: *il*, *im*, *ir*, *in* Morpheme: *in*

**Paradigm** is a list of all word forms.

Paradigm for verb *to be*: *am*, *is*, *are*, *was*, *were*, *will*

Languages by the type of grammatical features expression




Index of synthesis = M / W,
M - number of morphs in text;
W - number of words in text.

For analytical languages index < 2.0 (e.g. for English 1.68)

For synthetic languages index 2.0 - 3.0 (e.g. for Russian 2.33 - 2.45)

# Languages by the type of morphological structure

- **Isolating** languages: isolated morphemes as a word (e.g. *Chinese*);
- **Agglutinative** languages: a lot of affixes in word, each affix has its own meaning (e.g. *Turkish*);
- **Inflectional** languages: affixes are homonymous (e.g. *Russian*)

# Isolating languages (e.g. Mandarin Chinese)

Transliterated sentence: *gou bú ài chi qingcài*

may be literally translated as: *dog not like eat vegetable*

Depending on the context, it can mean any of the four following sentences:

- *the dog did not like to eat vegetables*
- *the dogs do not like to eat vegetables*
- *the dogs did not like to eat vegetables*
- *dogs do not like to eat vegetables*

# Agglutinative languages (e.g. Turkish)

- *ler* = plural
- *i* = possessive (e.g. *his*, *her*, *its*)
- *den* = ablative (e.g. a grammatical "case" ending showing a source, e.g. *from a house*)
    - ev: house
    - evler: houses
    - evi: his/her house
    - evleri: his/her houses, their houses
    - evden: from the house
    - evlerden: from the houses
    - evinden: from his/her house
    - evlerinden: from his/her houses, from their houses

# Inflecting languages (e.g. Latin)

*amo = I love*

Ending *o* is used to express the meanings:

- first person ("*I*" or "*we*"),
- singular,
- present tense,
- and also other meanings.

**Stemming** is the process of reducing inflected words to their word stem or root (the stem need not be identical to morphological root of the word): *‘stems’*, *‘stemmer’*, *‘stemming’*, *‘stemmed’* → *‘stem’ .*

**Lemmatization** is the process of getting the base form of the word: *‘tables’* → *‘table’*, *‘written’* → *‘write’*.

Tagging a wordform with its grammems: *‘table’*: [Noun, sing]; *‘book’*: [Noun, sing], [Verb, 1/2 person, sing/plur, Pr.Simple] / [Verb, 3 person, plur, Pr.Simple].

**Paradigm derivation** is the process of derivation all word forms from the base form.

# Morphological analyzers

- **Dictionary-based**: using a table (a dictionary), which contains mapping from set of words on set of lemmas. For Russian Zaliznyak's dictionary is used. Downside: it is impossible to get information for word if the dictionary does not contain it.
- **Analytical**: using a set of rules for morphological transformations. Don't cope with all morphological tasks, but good for stemming, lemmatization and getting paradigm.

# Lovins' algorithm (Lovins, 1968)

- 294 endings are defined;
- 29 conditions for removing one of the endings;
- 35 rules of wordform transformation after the ending removing

Example: '*nationally*' → '*nat*'. Two endings can be removed: '*ationally*' and '*ionally*'. But the first can't be removed because of the restriction: stem should be longer than 3 characters.

Downside: the algorithm requires linguists for rules and exceptions creating.

# Porter’s algorithm (Porter, 1980)

Rule: <*condition*>, <*ending*> → <*new ending*>

Contains ~ 60 rules, each of them is applied to the input wordform.

Example:

(*m* > 0) *eed* → *ee*  *agreed* → *agree*

# Algorithm of Paice&Husk (Paice/Husk, 1990)

Table of rules for ending transformations (removing or replacement).

Rule contains:

- inverted ending;
- integrity mark “*” (optional);
- length of the removing ending (including 0);
- string with length > 1, which has to be added (optional);
- symbols ‘>’ (switching to the pointed entry) or ‘.’ (stopping).

Example: “*nois4j>*”

# Comparison

| | |
|---|---|
| Original sentence | *Such an analysis can reveal features that are not easily visible from the variations in the individual genes.* |
| Lovins' algorithm | *Such an analysis can reve featur that ar not eas vis from th vari in th individu gen* |
| Porter's algorithm | *Such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene.* |
| Algorithm of Paice&Husk | *Such an analysis can rev feat that are not easy vis from the vary in the individ gen* |

# Why does morphology matter?

- Information retrieval:
  - A query for ***phones*** should match both ***phone*** and ***phones***
- Language modeling:
  - If we have seen ***scrutinize***, we can predict ***scrutinized***
- Machine translation:
  - Swedish ***bilen*** corresponds to English ***the car***
- etc.

# Morphological analyzers

- **Morphology software for English**: https://aclweb.org/aclwiki/Morphology_software_for_English
- **AOT**: http://aot.ru/technology.html
- **PyMorphy**: *documentation*: http://pymorphy.readthedocs.io/en/v0.5.6/index.html , *code*: https://github.com/kmike/pymorphy2
- **TreeTagger**: http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
- **MyStem**: https://tech.yandex.ru/mystem/
- more: https://nlpub.ru/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%B0

Thank you for your attention!