Natural Language Processing

Prof. Dr. Jannik Strötgen
jannik.stroetgen@h-ka.de

Summer 2024

Hochschule Karlsruhe
University of
Applied Sciences

Fakultät für
Informatik und
Wirtschaftsinformatik

[2] Pre-processing

Outline

- [2] Pre-processing
  - Text segmentation
    - Sentence splitting
    - Tokenization
  - Token normalization
    - Stemming
    - Lemmatization

From the Introduction

1. Pre-processing

Ahdistinctlyirememberitwasinthebleakdecemberandeachseparatedyingember
wroughtitsghostuponthefloореagerlyiwishedthemorrowvainlyIhadsoughttobor
rowfrommybookssurceaseofsorrowsorrowforthelostlenorefortherareandradia
ntm heangelsnamelenorenamelesshereforevermore

adj: distinct
adv: distinctly

Ah, distinctly I remember it was in the bleak December;
And each separate dying ember wrought its ghost upon the floor.
Eagerly I wished the morrow;—vainly I had sought to borrow
From my books surcease of sorrow—sorrow for the lost Lenore—
For the rare and radiant maiden whom the a Lenore—
Nameless here for evermore.

to seek
1. person sg.
past perfect

March 19, 2024 Natural Language Processing @ HKA - Prof. Dr. Jannik Strötgen 10

Sentence Splitting (from last week)

Using whitespaces and punctuation, sentences should not be too difficult to detect… right?

- ! and ? are relatively unambiguous
- A period "." is quite ambiguous, but only has a few cases such as
  - Sentence boundaries
  - Abbreviations
  - Numbers
  - etc.

...at least as long as we have punctuation.

- Classical Greek and Latin texts use no punctuation
- Classical Chinese considers punctuation as optional
- Thai is using whitespaces instead of punctuation

Tokenization (from last week)

Token:

- The occurrence of a word in a text

Tokenization:

- Segmentation of an input stream into an ordered sequence of tokens

Tokenizer:

- A system that splits texts into word tokens

Example:

- Input text: John likes Mary and Mary likes John.
- Tokens: {"John", "likes", "Mary", "and", "Mary", "likes", "John", "."}
- Types: {“John”, “likes”, “Mary”, “and”, “.”}

[2] Pre-processing: Token Normalization

Stemming and Lemmatization

Recall:

- Tokenization (ideally) returns word occurrences in a text
- Word occurrence frequencies are Zipf-distributed
- Due to morphological variants, many word occurrences will be unique, e.g., `computerization`, `computer`, `computing`, `computation`

Stemming and Lemmatization

To make these words comparable, we want to **normalize** them by grouping them into equivalence classes. This can be done by reducing them to a root morpheme by

- **stemming**, which reduces a word to its stem (`comput`) or
- **lemmatization**, which reduces a word to its lemma (`compute`)

Since {`computerization`, `computer`, `computing`, `computation`} are all derived from the same root, we can then combine their occurrences in our corpus statistics – which for some tasks might be what we want.

Porter Stemmer

A simple approach:

- We effectively chop off the end of the word!
- Only suffixes are considered for removal
- Frequently used algorithm (e.g., in Information Retrieval)
- Results are quite ugly (from a linguistic perspective)

Manually designed rules for suffix stripping are applied, e.g.:

`-sses → ss`	`caresses → caress`
`-ies → i`	`libraries → librari`
`s → ∅`	`dogs → dog`

Porter Stemmer

Core idea

- Consecutively remove / replace suffixes
- The number of iterations depends on the syllables of the word
- In each step, one rule from a fixed set of rules is applied
- No rule may be applied twice
- Once no rule can be applied, the algorithm stops

Porter Stemmer: Measure of a Word

Number of syllables is approximated heuristically by the number of vowel-consonant sequences between (optional) leading consonants and (optional) trailing vowels. Formally, let

- V denote a sequence of vowels and
- C denote a sequence of consonants.

Then each word can be modeled as: **[C](VC)^m[V]** and **m** is called the **measure** of the word. We use **m** to approximate the number of syllables.

Examples:

m=0: `to` `[t][o]`

m=1: `brick` `[br](ick)`

m=2: `eastern` `(eas t)(ern)`

Porter Stemmer: Application of Rules

Each rule comes with a constraint that determines when it can be applied, which uses the measure. For example:

- Rule: If (m>0): ATION -> ATE
  - `medication` → `medicate`
  - `nation` → `nation` (without "`ation`" the word is a single consonant: m=0)

"Computing" Example:

- Input = `computational`
- Replace `-ational` with `-ate` → `computate`
- Replace `-ate` with nothing → `comput`

Stemming: Evaluation

There are two types of error during stemming:

- Over-stemming
  - Two inflected words are stemmed to the same root when they should have been treated as separate
  - Example: `universal`, `university`, `universe` → `univers`

Under-stemming

- Two separate inflected words should be stemmed to the same root but are stemmed to different roots
  - Example: `alumnus` → `alumnu`
    `alumni` → `alumni`
    `alumna` → `alumna`

Lemmatization

A set of more sophisticated approaches to finding the root of a word, that include some of the following techniques:

- Using an understanding of inflectional morphology (e.g., ≈ reverse inflection for a verb)
- Using a set of rules for the detachment of morphemes
- Using an exception list for irregular inflections
- Utilizes word collocation information
- Utilizes **part-of-speech** tagging (more on that later)
- Comparing the results to linguistic resources such as **WordNet** (more on that later)
- Keeping the original form if transformations do not match a dictionary
- Many more involved techniques...

Lemmatization

Disadvantages of lemmatization: complexity and need for knowledge and understanding of the context.

- **Example:** `saw`

→ (`to`) `see` | (`the`) `saw`

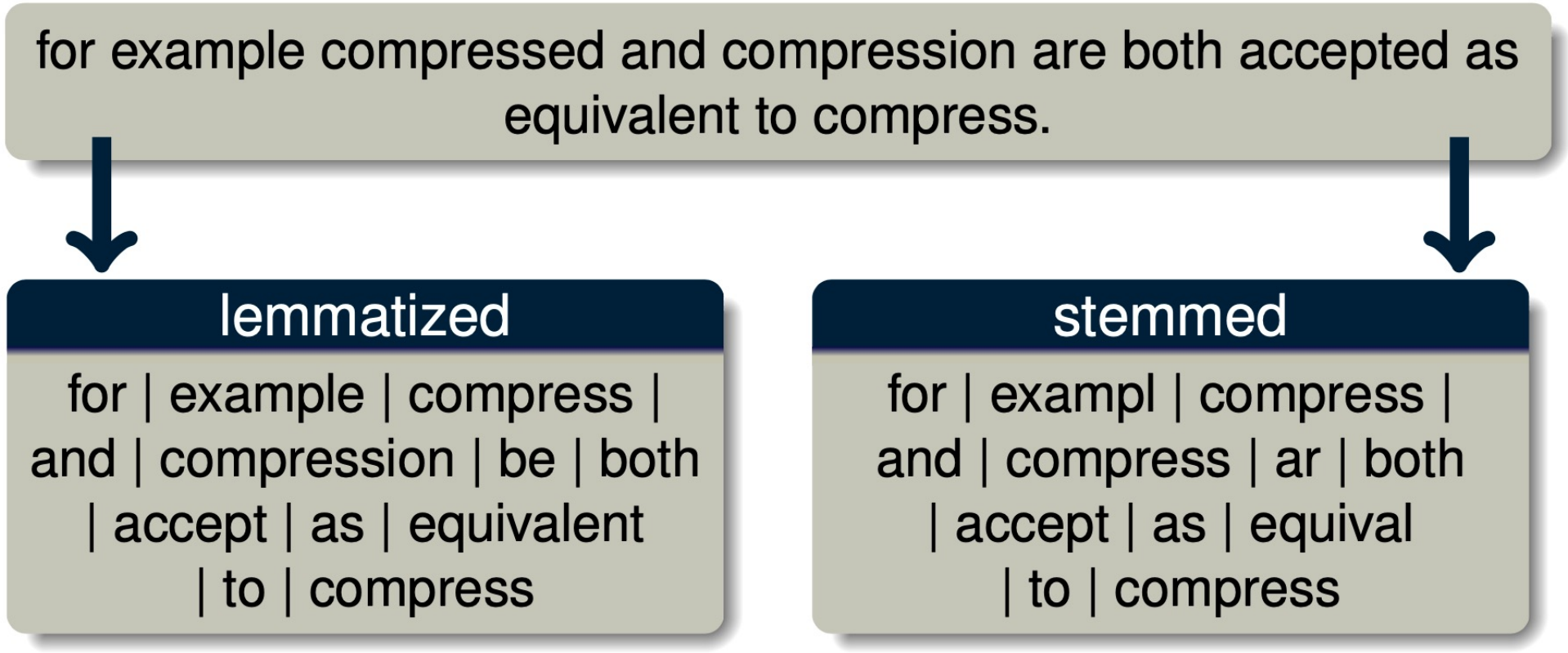
https://www.meinmed.at/gesundheit/auge-anatomie/1470

https://www.krippenursel.de/saege-gr.html

Stemming vs. Lemmatization

Input	WordNet Lemmatizer	Porter Stemmer
leaves	leaf leave	leav
acceptable	acceptable	accept

Stemming vs. Lemmatization



Stopword Removal

Stopword Removal

- **Stopwords** (e.g., `a`, `the`, `of`) are words, which carry only little information
- They occur in pretty much every document of a language

- **In information retrieval (i.e., search), removing stopwords**
  - Reduces the number of terms, which need to be indexed
  - Improves response times
  - Can improve value of search results (e.g., `a song of fire and ice`)
  - Can decrease value of search results (e.g., `the who`)
- In **data analytics**, stopwords are also not meaningful as they do not characterize any document

Stopword Removal

Stopword Removal

- Based on a manually defined list of stopwords (sometimes with domain-specific stopwords)

```
a, an, and, are, as, at, be, by, for,
has, he, in, is, it, its, of, on, that,
the, to, was, where, will, with
```

- List of stopwords can be automatically constructed and contains all terms which occur frequently in pretty much all documents of a document collection.

Online Tools and Resources

http://text-processing.com/demo/

http://textanalysisonline.com/nltk-porter-stemmer

http://textanalysisonline.com/nltk-wordnet-word-lemmatizer

http://textanalysisonline.com/nltk-wordnet-lemmatizer

Further Watching Material

Word Tokenization

https://youtu.be/dzSQ0-SEqxQ?list=PLoROMvodv4rOFZnDyrlW3-nI7tMLtmiJZ

Word Normalization and Stemming

https://youtu.be/rHWCHeDmXFc?list=PLoROMvodv4rOFZnDyrlW3-nI7tMLtmiJZ

Summary: Foundations and Preprocessing

Difficulties of NLP:

- Language enables communication between humans, not computers
- Language presupposes world knowledge / prior knowledge
- Taming the complexity that emerges when words are combined to text

Evolution of three major types of NLP methods:

- Rule-based models
- Statistical models / corpus linguistics
- (Deep) neural network models

Summary: Foundations and Preprocessing

Distributional semantics can help us understand words purely based on their context.

“You shall know a word by the company it keeps”

J. R. Firth, 1957

Methods from **corpus linguistics** enable us to derive context statistics that can then be used to construct algorithms that solve NLP tasks.

Summary: Foundations and Preprocessing

Elements of language:

- Morphology: Internal composition of words (morphemes)
- Syntax: Composition of sentences from words

Language	
Sound	1. Phonetics
Grammar	2. Phonology 3. Morphology 4. Syntax
Meaning	5. Semantics

Text pre-processing:

- Tokenization
- Sentence splitting
- Stemming and Lemmatization
- Stopword Removal

Thank you for your attention!

Questions?