Chapter NLP:III

III. Words

- Word-level Phenomena & Text Preprocessing
- Morphological Analysis
- Word Classes

Word-level Phenomena

Phenomena listing

Tokens are the individual occurrences of something

- words
- numbers
- punctuation marks
- emoticons

Words might be consisting of inner word special characters

- `data-source, deep-learning`
- `devil's advocate`
- `Micro$oft`
- `Sk8er`

Multiple words might be composing a single context unit named "Multi Word Expression"

- Names: `John Doe, New York, Bad Düben`
- Idioms: `piece of cake, raining cats and dogs, devil's advocate`

Word-level Phenomena

Phenomena listing (Fortsetzung)

Stop Words are function words which do not carry any special information regarding topics or intends of a text

- `the, from, could, he, she`
- very frequent
- wordforms from closed categories – see POS in the following point

Linguists divide words of a language into classes (sets) of words which show similar syntactic behavior

- parts of speech (POS)
- Most important: noun (people, animal, concept), verb (action), adjective (expression, property)
- open category: many members – see above
- closed category: few members – e.g. prepositions, determiners

Morphology is the systematic production of new wordforms by inflection, derivation and compounding of single lexemes

- `dog, dog-`**`s`**`, call, call-`**`ed`**

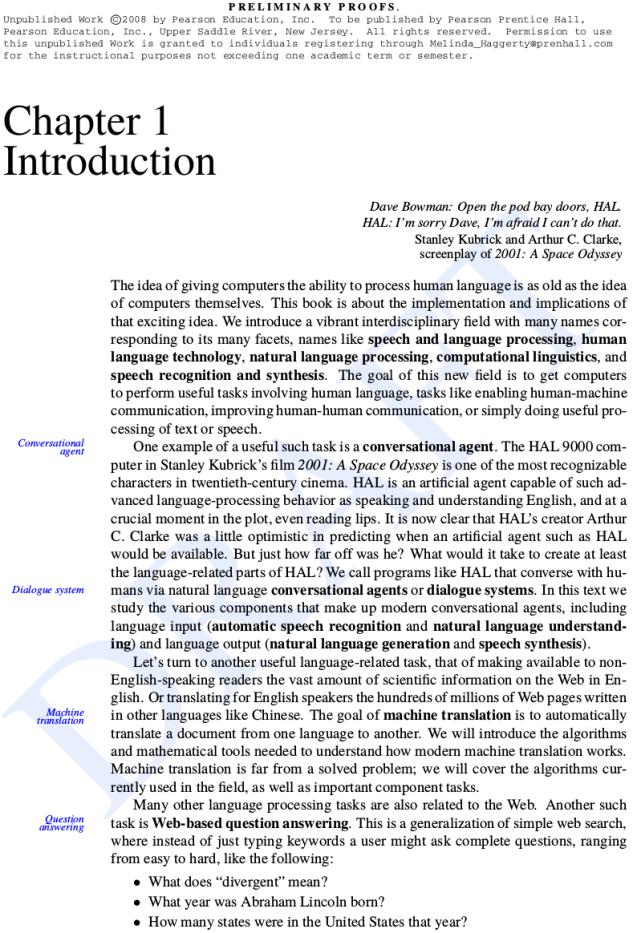
Casing – Words might appear with different case depending on their position in the text

- **`Dogs`** `are barking at people on the street! Why are the` **`dogs`** `doing this?`

Text Preprocessing

Overview

The goal of text preprocessing is its conversion into a canonical form.



PRELIMINARY PROOFS.

Chapter 1
Introduction

Dave Bowman: Open the pod bay doors, HAL.
HAL: I'm sorry Dave, I'm afraid I can't do that.
Stanley Kubrick and Arthur C. Clarke,
screenplay of *2001: A Space Odyssey*

The idea of giving computers the ability to process human language is as old as the idea of computers themselves. This book is about the implementation and implications of that exciting idea. We introduce a vibrant interdisciplinary field with many names corresponding to its many facets, names like **speech and language processing**, **human language technology**, **natural language processing**, **computational linguistics**, and **speech recognition and synthesis**. The goal of this new field is to get computers to perform useful tasks involving human language, tasks like enabling human-machine communication, improving human-human communication, or simply doing useful processing of text or speech.

Conversational agent

One example of a useful such task is a **conversational agent**. The HAL 9000 computer in Stanley Kubrick's film *2001: A Space Odyssey* is one of the most recognizable characters in twentieth-century cinema. HAL is an artificial agent capable of such advanced language-processing behavior as speaking and understanding English, and at a crucial moment in the plot, even reading lips. It is now clear that HAL's creator Arthur C. Clarke was a little optimistic in predicting when an artificial agent such as HAL would be available. But just how far off was he? What would it take to create at least the language-related parts of HAL? We call programs like HAL that converse with humans via natural language **conversational agents** or **dialogue systems**. In this text we study the various components that make up modern conversational agents, including language input (**automatic speech recognition** and **natural language understanding**) and language output (**natural language generation** and **speech synthesis**).

Dialogue system

Machine translation

Let's turn to another useful language-related task, that of making available to non-English-speaking readers the vast amount of scientific information on the Web in English. Or translating for English speakers the hundreds of millions of Web pages written in other languages like Chinese. The goal of **machine translation** is to automatically translate a document from one language to another. We will introduce the algorithms and mathematical tools needed to understand how modern machine translation works. Machine translation is far from a solved problem; we will cover the algorithms currently used in the field, as well as important component tasks.

Question answering

Many other language processing tasks are also related to the Web. Another such task is **Web-based question answering**. This is a generalization of simple web search, where instead of just typing keywords a user might ask complete questions, ranging from easy to hard, like the following:

- What does "divergent" mean?
- What year was Abraham Lincoln born?
- How many states were in the United States that year?

screenshot-jurafsky08-speech-and-language-processing-pdftotext-output.txt

Open Save

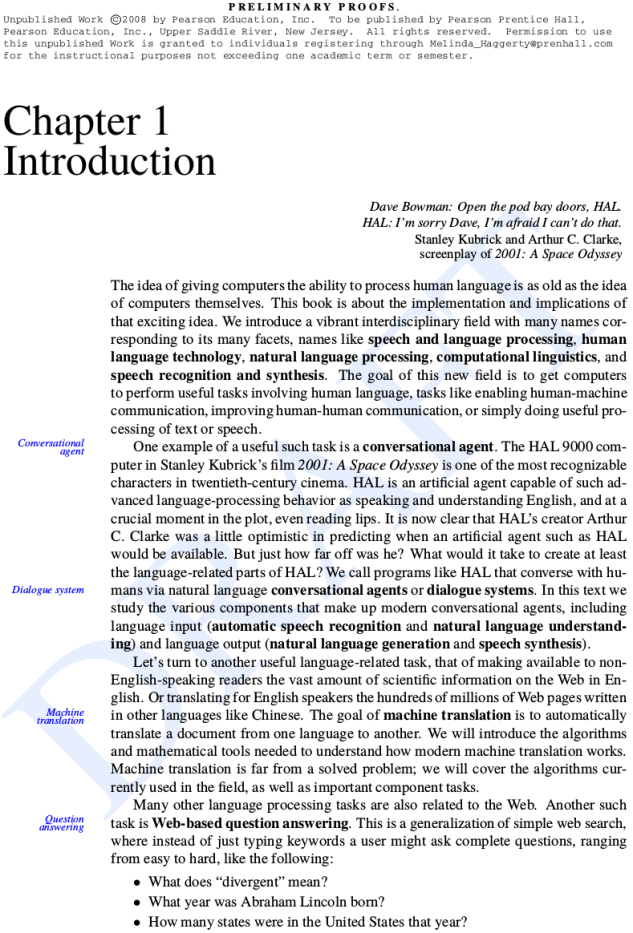
```
PRELIMINARY PROOFS.
c
Unpublished Work ©2008
by Pearson Education, Inc. To be published by Pearson Prentice Hall,
Pearson Education, Inc., Upper Saddle River, New Jersey. All rights reserved. Permission to use
this unpublished Work is granted to individuals registering through Melinda_Haggerty@prenhall.com
for the instructional purposes not exceeding one academic term or semester.

Chapter 1
Introduction
Dave Bowman: Open the pod bay doors, HAL.
HAL: I'm sorry Dave, I'm afraid I can't do that.
Stanley Kubrick and Arthur C. Clarke,
screenplay of 2001: A Space Odyssey

FT

D
RA

Conversational
agent

The idea of giving computers the ability to process human language is as old as the idea
of computers themselves. This book is about the implementation and implications of
that exciting idea. We introduce a vibrant interdisciplinary field with many names corresponding to its
  many facets, names like speech and language processing, human
language technology, natural language processing, computational linguistics, and
speech recognition and synthesis. The goal of this new field is to get computers
to perform useful tasks involving human language, tasks like enabling human-machine
communication, improving human-human communication, or simply doing useful processing of text or speech.
One example of a useful such task is a conversational agent. The HAL 9000 computer in Stanley Kubrick's
  film 2001: A Space Odyssey is one of the most recognizable
characters in twentieth-century cinema. HAL is an artificial agent capable of such advanced language-
  processing behavior as speaking and understanding English, and at a
crucial moment in the plot, even reading lips. It is now clear that HAL's creator Arthur
C. Clarke was a little optimistic in predicting when an artificial agent such as HAL
would be available. But just how far off was he? What would it take to create at least
the language-related parts of HAL? We call programs like HAL that converse with humans via natural
  language conversational agents or dialogue systems. In this text we
study the various components that make up modern conversational agents, including
language input (automatic speech recognition and natural language understanding) and language output
  (natural language generation and speech synthesis).
Let's turn to another useful language-related task, that of making available to nonEnglish-speaking
  readers the vast amount of scientific information on the Web in English. Or translating for English
  speakers the hundreds of millions of Web pages written
in other languages like Chinese. The goal of machine translation is to automatically
translate a document from one language to another. We will introduce the algorithms
and mathematical tools needed to understand how modern machine translation works.
Machine translation is far from a solved problem; we will cover the algorithms currently used in the
  field, as well as important component tasks.
Many other language processing tasks are also related to the Web. Another such
task is Web-based question answering. This is a generalization of simple web search,
where instead of just typing keywords a user might ask complete questions, ranging
from easy to hard, like the following:

Dialogue system

Machine
translation

Question
answering

• What does "divergent" mean?
• What year was Abraham Lincoln born?
• How many states were in the United States that year?
```

Plain Text Tab Width: 2 Ln 1, Col 35 INS

Text Preprocessing

Overview

The goal of text preprocessing is its conversion into a canonical form.



PRELIMINARY PROOFS.

Chapter 1
Introduction

Dave Bowman: Open the pod bay doors, HAL.
HAL: I'm sorry Dave, I'm afraid I can't do that.
Stanley Kubrick and Arthur C. Clarke,
screenplay of *2001: A Space Odyssey*

The idea of giving computers the ability to process human language is as old as the idea of computers themselves. This book is about the implementation and implications of that exciting idea. We introduce a vibrant interdisciplinary field with many names corresponding to its many facets, names like **speech and language processing**, **human language technology**, **natural language processing**, **computational linguistics**, and **speech recognition and synthesis**. The goal of this new field is to get computers to perform useful tasks involving human language, tasks like enabling human-machine communication, improving human-human communication, or simply doing useful processing of text or speech.

Conversational agent

One example of a useful such task is a **conversational agent**. The HAL 9000 computer in Stanley Kubrick's film *2001: A Space Odyssey* is one of the most recognizable characters in twentieth-century cinema. HAL is an artificial agent capable of such advanced language-processing behavior as speaking and understanding English, and at a crucial moment in the plot, even reading lips. It is now clear that HAL's creator Arthur C. Clarke was a little optimistic in predicting when an artificial agent such as HAL would be available. But just how far off was he? What would it take to create at least the language-related parts of HAL? We call programs like HAL that converse with humans via natural language **conversational agents** or **dialogue systems**. In this text we study the various components that make up modern conversational agents, including language input (**automatic speech recognition** and **natural language understanding**) and language output (**natural language generation** and **speech synthesis**).

Dialogue system

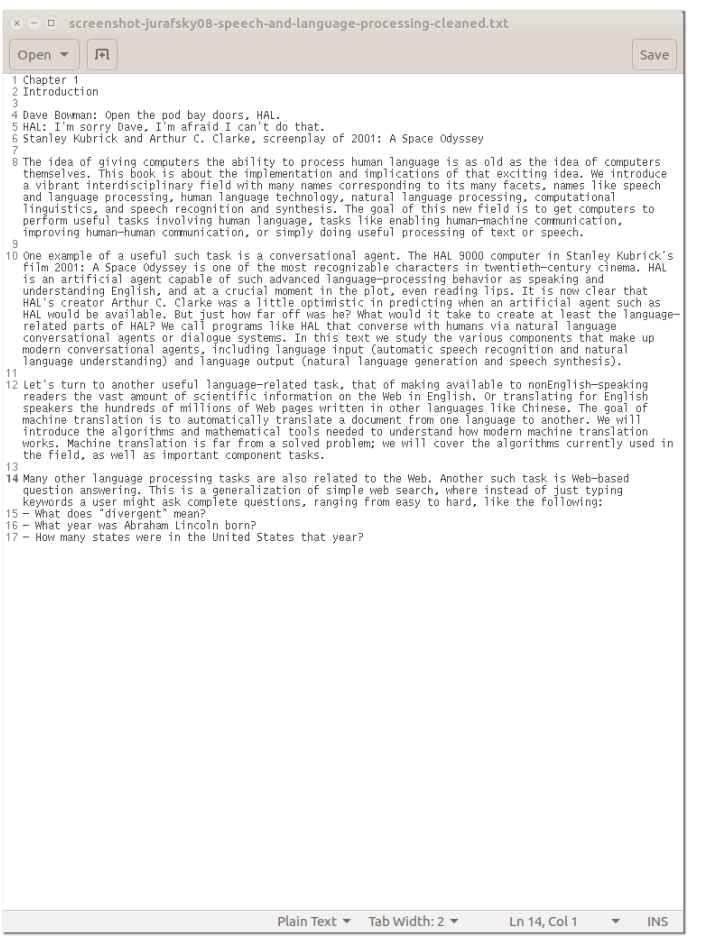
Machine translation

Let's turn to another useful language-related task, that of making available to non-English-speaking readers the vast amount of scientific information on the Web in English. Or translating for English speakers the hundreds of millions of Web pages written in other languages like Chinese. The goal of **machine translation** is to automatically translate a document from one language to another. We will introduce the algorithms and mathematical tools needed to understand how modern machine translation works. Machine translation is far from a solved problem; we will cover the algorithms currently used in the field, as well as important component tasks.

Question answering

Many other language processing tasks are also related to the Web. Another such task is **Web-based question answering**. This is a generalization of simple web search, where instead of just typing keywords a user might ask complete questions, ranging from easy to hard, like the following:

- What does "divergent" mean?
- What year was Abraham Lincoln born?
- How many states were in the United States that year?



Text Preprocessing

Overview

Rationale:

- Ease implementation of subsequent processing steps
  A unified input format simplifies implementing processing steps. Example: In web search engines, all documents are converted to HTML. All indexing steps can expect HTML as input.
- Avoid processing errors and model bias
  Many rule-based and learning-based processing steps in an NLP pipeline may fail or be misled because of random text artifacts. High-level processing steps presume clean text. Examples: Text classifications may learn to exploit PDF-to-text conversion artifacts rather than a text's contents; parsers require grammatical text.

Constraints:

- Task-dependence
  The canonical form depends on the task at hand and its requirements.
- Provenance
  The possibility to determine from where in a raw text corpus, a preprocessed text originated.
- Reversibility
  The capability to render a preprocessed text in human-readable form.

Text Preprocessing

Overview

Common preprocessing steps:

- Conversion to plain text
- Line break encodings: \n – UNIX, \r\n – Windows
- Encoding detection and unification
  e.g. `iconv -f original_charset -t utf-8 originalfile > newfile`
- Extraction of main content and meta information
- Normalization and/or paraphrasing
- Annotation

Plain text formats:

- unformatted, raw
- formatted and/or markup
- inline or external annotations

Text Preprocessing

Overview

Normalization and/or paraphrasing:

- Faithful to the original text
- Departure from the original text
  - Unification across texts
  - Canonicalization
    Whitespace, spelling, grammar; translation of text messages to common text norms
  - Expansion and/or abstraction
    Abbreviations, anaphora, translation to spoken language, canonicalization of tokens

Annotation:

- Syntactic units: phonemes, morphemes, tokens (esp. words), sentences
- Discourse units: paragraphs, sections, chapters
- Typographic units: lines, pages (layout, meta-information), documents
- Meta-information: title, authors, date, properties, ...

Text Preprocessing

Tokenization

Tokenization turns a sequence of characters into a sequence of tokens.

Example:

```
Friends, Romans, Countrymen, lend me your ears !
```

`Friends` `,` `Romans` `,` `Countrymen` `,` `lend` `me` `your` `ears` `!`

Terminology: (simplified)

- A token is a character sequence forming a useful semantic unit.
- A type is to a token what a class is to an object.

Token-granularity:

- Word-level: may or may not include whitespace between words
- Phrase-level: identification of multi-term named entities and common phrases
- Sentence-level: one token corresponds to one clause, or one sentence

Remarks:

- A related philosophical concept is the type-token distinction (see unit about corpus linguistics in this course). Here, a token is a specific instance of a word (i.e., its specific written form), and a type refers to its underlying concept as a whole. This is comparable to the distinction between class and object in object-oriented computer programming. For example, the sentence “A rose is a rose is a rose.” comprises nine token instances but only four types, namely “a”, “rose”, “is”, and “.”. [Wikipedia]
- Tokenization is strongly language-dependent. English is already among the easiest languages to be tokenized, and there are still many problems to be solved. In Chinese, for example, words are not separated by a specific character, rendering the process of determining word boundaries much more difficult.

Text Preprocessing

Tokenization: Special Cases

- **Contractions**
  Apostrophes can be a part of a word, a part of a possessive, or just a mistake: `it's`, `o'donnell`, `can't`, `don't`, `80's`, `men's`, `master's degree`, `shriner's`
- **Hyphenated compounds**
  Hyphens may be part of a word, a separator, and some words refer to the same concept with or without hyphen: `winston-salem`, `e-bay`, `wal-mart`, `active-x`, `far-reaching`, `loud-mouthed`, `20-year-old`.
- **Compounds**
  English: `wheelchair`, German: `Computerlinguistik` for computational linguistics.
- **Other special characters**
  Special characters may form part of words, especially in technology-related text: `M*A*S*H`, `I.B.M.`, `Ph.D.`, `C++`, `C#`, ` `, `http://www.example.com`.
- **Numbers**
  Numbers form tokens of their own, and may contain punctuation as well: `6.5`, `1e+010`.
- **Phrase tokens: named entities, phone numbers, dates**
  `San Francisco`, `(800) 234-2333`, `Mar 11,1983`.

Text Preprocessing

Tokenization Approaches

- Heuristics
  - Whitespace: A token is every character sequence separated by whitespace characters.
  - TREC: A token is every alphanumeric sequence of characters of length > 3, separated by a space or punctuation mark.
- Rule-based

  Applications of rules to a text so that tokens are separated by whitespace from each other. This allows subsequent processing steps to apply the Whitespace Heuristic
- Machine learning-based

  Based on sufficiently large training data of correctly tokenized text, a model can be trained to decide at every character position whether to split tokens.

Text Preprocessing

Rule-based Tokenization

Algorithm: Tokenization with Regular Expressions.

Input: d. Document in the form of a string.
A. Dictionary of abbreviations.

Output: The document with space in-between its tokens.

Tokenize(d, A)

1. `alnum = [A-Za-z0-9]; nalnum = [^A-Za-z0-9]; alwayssep = [?!()";/\|']`
2. `clitic = ('|:|-|'S|'D|'M|'LL|'RE|'VE|N'T|'s|'d|'m|'ll|'re|'ve|n't)`
3. Apply **s/$alwayssep/␣$&␣/g** to d.
4. Apply **s/([^0-9]),/$1␣,␣/g** and **s/,([^0-9])/␣,␣$1/g** to d.
5. Apply **s/^'/$&␣/g** and **s/($nalnum)'/$1␣'/g** to d.
6. Apply **s/$clitic$/␣$&/g** and **s/$clitic($nalnum)/␣$1␣$2/g** to d.
7. Split d by whitespace (**/\s+/**) to obtain a list of tokens T.
8. Apply **s/\.$/␣\./** to $t \in T$ if t matches **/$alnum\./** and is not in A and does not match **/^([A-Za-z]\.([A-Za-z]\.)+)$/**.
9. Optionally expand clitics: **s/'ve/have/** and **s/'m/am/** and so on.
10. Return a whitespace-separated string of T.

Text Preprocessing

Stopping (Token Removal)

Stopping refers to the removal of tokens from a token sequence that are not useful in order to reduce data and improve performance of subsequent tasks:

- Frequent tokens (collection-specific)
  Example: `Wikipedia` when processing Wikipedia.
- Function word tokens (language-dependent)
  `the`, `of`, `and`, etc; strong overlap with frequent tokens.
  Problem: `to be or not to be` would be completely lost.
- Punctuation-only tokens
  Counter-example: `;-)`
- Number-only tokens
- Short tokens
  Short words may be important. Examples: `xp`, `ma`, `pm`, `ben e king`, `el paso`, `master p`, `gm`, `j lo`, `world war II`.

Stop word lists are typically customized to the text domain.

Example of stopword removal

```
The idea of giving computers the ability to process human language is as
old as the idea of computers themselves.  This book is about the
implementation and implications of that exciting idea.  We introduce a
vibrant interdisciplinary field with many names corresponding to its many
facets, names like speech and language processing, human language
technology, natural language processing, computational linguistics, and
speech recognition and synthesis.  The goal of this new field is to get
computers to perform useful tasks involving human language, tasks like
enabling human-machine communication, improving human-human
communication, or simply doing useful processing of text or speech.
```

Example of stopword removal

[redacted] idea [redacted] giving computers [redacted] ability [redacted] process human language [redacted] [redacted] old [redacted] [redacted] idea [redacted] computers themselves[redacted] [redacted] book [redacted] [redacted] [redacted] implementation [redacted] implications [redacted] [redacted] exciting idea[redacted] [redacted] introduce [redacted] vibrant interdisciplinary field [redacted] [redacted] names corresponding [redacted] [redacted] [redacted] facets[redacted] names [redacted] speech [redacted] language processing[redacted] human language technology[redacted] natural language processing[redacted] computational linguistics[redacted] [redacted] speech recognition [redacted] synthesis[redacted] [redacted] goal [redacted] [redacted] new field [redacted] [redacted] [redacted] computers [redacted] perform useful tasks involving human language[redacted] tasks like enabling human-machine communication[redacted] improving human-human communication[redacted] [redacted] simply [redacted] useful processing [redacted] text [redacted] speech[redacted]

Text Preprocessing

Token Normalization

Application of heuristic rules to each token in an attempt to unify them. (e.g. Normalization of social media texts for NLP [Clarke, Araki 2011])

- Lower-casing
  Problem: Capitalization may carry distinctions between word semantics.
  Examples: `Bush` vs. `bush`, `Apple` vs. `apple`.
- Removal of special characters
  Example: `U.S.A.` → `USA`
- Removal of diacritical marks
  Example: `café` → `cafe`
- Spelling correction: `I think my gramma got die of beaties` → `I think my grandma got diabetes`
- Source data may contain "noise"
  - messed up OCR results
  - page headers, page numbers in OCR-scanned books
  - XML-documents may contain copyright information in body texts
  - HTML-text may contain unwanted javascript text
- Morphological analysis
  Lemmatization or stemming

Chapter NLP:III

III. Words

- Word-level Phenomena & Text Preprocessing
- Text Preprocessing
- Morphological Analysis
- Word Classes

Morphological Analysis

Overview [Hancox 1996]

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

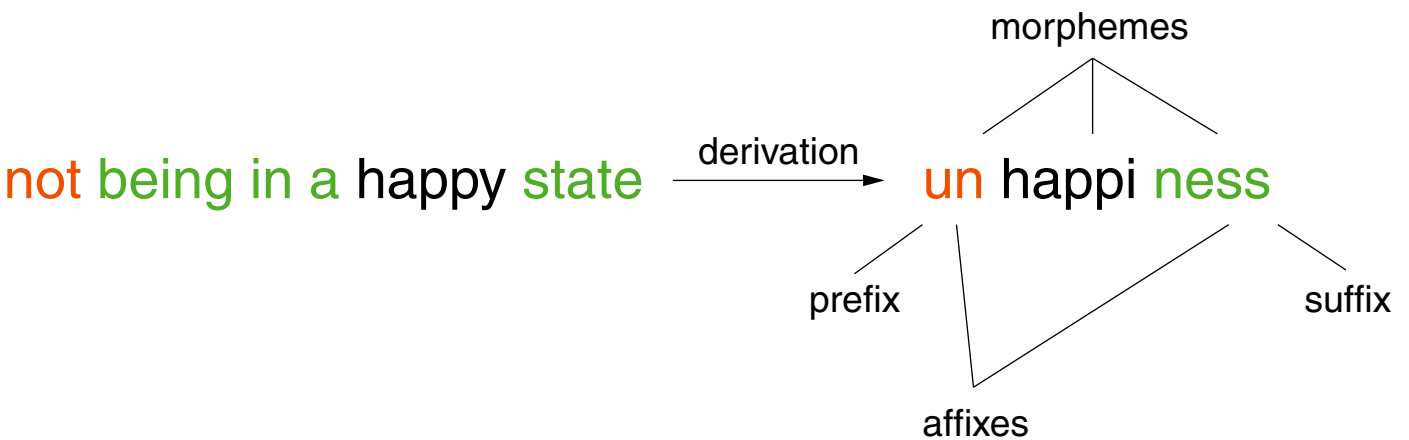
call in the past $\xrightarrow{\text{inflection}}$ call ed

not being in a happy state $\xrightarrow{\text{derivation}}$ un happi ness

Morphological Analysis

Overview [Hancox 1996]

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

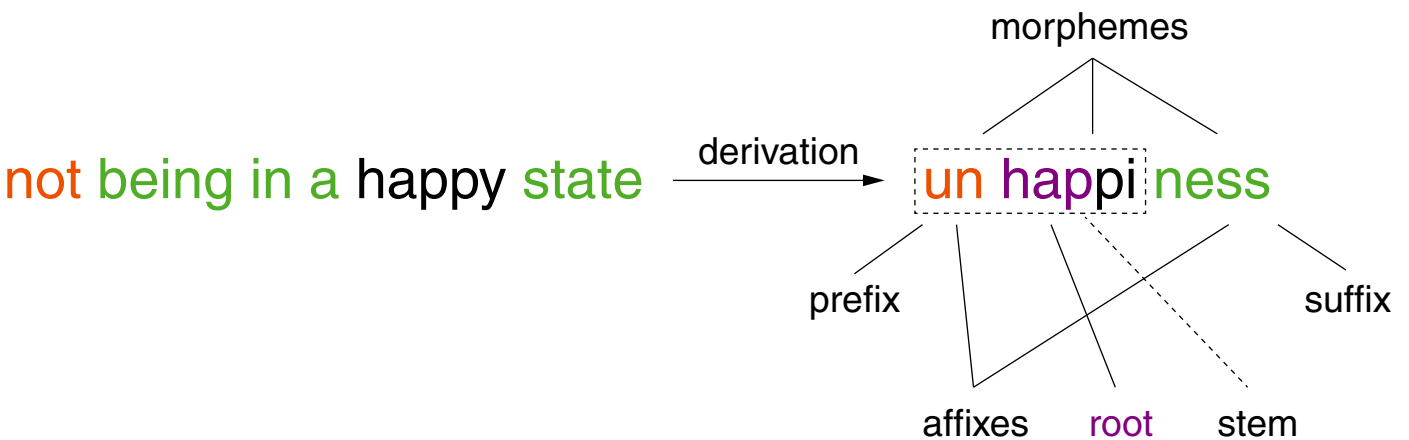


- A morpheme is a “minimal unit of meaning”.
  Free morphemes can also be used as words.
  Bounded morphemes appear only as affixes (prefix, suffix, infix, and more) to words.

Morphological Analysis

Overview [Hancox 1996]

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

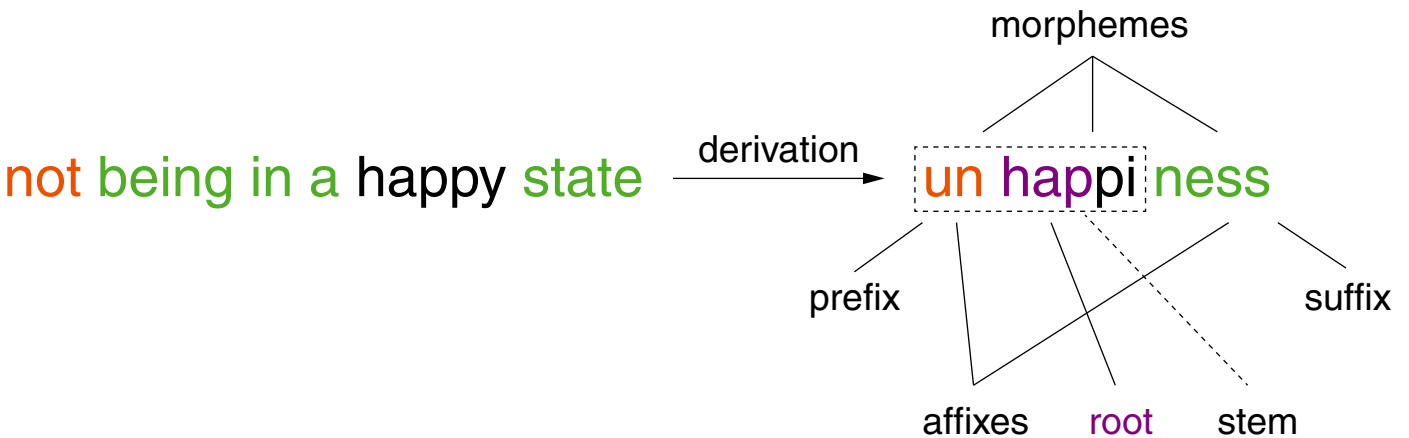


- A morpheme is a “minimal unit of meaning”.
  Free morphemes can also be used as words.
  Bounded morphemes appear only as affixes (prefix, suffix, infix, and more) to words.
- A root is a single morpheme, a stem one or more.
  A root is the derivational base, or type, of a word, a stem its inflectional base.

Morphological Analysis

Overview [Hancox 1996]

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



- A morpheme is a “minimal unit of meaning”.
  Free morphemes can also be used as words.
  Bounded morphemes appear only as affixes (prefix, suffix, infix, and more) to words.
- A root is a single morpheme, a stem one or more.
  A root is the derivational base, or type, of a word, a stem its inflectional base.

➜ Morphological analysis: identification of a word’s morphemes and their role.

Morphological Analysis

Stemming

Mapping of a word token to its word stem by removal of inflection (e.g., affixes).

Inflections:

- noun declination (grammatical case, numerus, gender)
- verb conjugation (grammatical person, numerus, tense, mode, . . .)
- adjective and adverb comparison

Example:

connect	connects
	connected
	connecting
	connection

Morphological Analysis

Stemming: Principles [Frakes 1992]

1. Table lookup:
   Given a word stem, store its inflections in a hash table. Problem: completeness.

2. Affix elimination:
   Rule-based algorithms to identify prefixes and suffixes. Given their efficiency and intuitive workings, these are most commonly used.

3. Character n-grams:
   Usage of 4-grams or 5-grams from tokens as stems. Basic heuristic for English: use the first 4 characters as stem.

4. Successor variety:
   Exploits knowledge about structural linguistics to identify morpheme boundaries. The character sequences of tokens are added to a trie data structure; the outdegrees of inner nodes are analyzed to find suitable stems. Problem: difficult to operationalize.

Morphological Analysis

Stemming: Affix Elimination

Principle: “iterative longest match stemming”

1. Removal of the longest possible match based on a set of rules.
2. Repetition of Step 1 until no rule can be applied, anymore.
3. Recoding to address irregularities captured by the rules.

Morphological Analysis

Stemming: Affix Elimination

Principle: “iterative longest match stemming”

1. Removal of the longest possible match based on a set of rules.
2. Repetition of Step 1 until no rule can be applied, anymore.
3. Recoding to address irregularities captured by the rules.

Notation:

- $\mathtt{c}$ denotes a consonant, $\mathtt{C}$ a non-empty sequence of consonants.
  $\mathtt{v}$ denotes a vowel, $\mathtt{V}$ a non-empty sequence of vowels.
  - ➜ Every word is defined by $[\mathtt{C}](\mathtt{VC})^{\mathtt{m}}[\mathtt{V}]$
- Consonant: Letter that is not a vowel.
- Vowel: Letters A, E, I, O, and U as well as Y after a consonant.
  Example: In TOY the Y is a consonant, in LOVELY the Y is a vowel.

Morphological Analysis

Stemming: Porter Stemmer

Concepts:

- 9 rule sets, each consisting of 1-20 rules
- Rules of each group are sorted, to be applied top to bottom
- Only one rule per set can be applied
- Rules are defined as follows: `<Premise> S1` $\longrightarrow$ `S2`

Morphological Analysis

Stemming: Porter Stemmer

Concepts:

- 9 rule sets, each consisting of 1-20 rules
- Rules of each group are sorted, to be applied top to bottom
- Only one rule per set can be applied
- Rules are defined as follows: `<Premise> S1` $\longrightarrow$ `S2`

Semantics:

If a character sequence ends with `S1` and if the subsequence ahead of `S1` (= word stem) fulfills the `<Premise>`, replace `S1` by `S2`

Premises:

`(m>x)`	Number of vowel-consonant-sequences is larger than `x`.
`(*S)`	Word stem ends with `S`.
`(*v*)`	Word stem contains a vowel.
`(*o)`	Word stem ends with `cvc`, where the second consonant `c` $\notin$ `{W, X, Y}`.
`(*d)`	Word stem ends with two identical consonants.

Morphological Analysis

Stemming: Porter Stemmer

Selection of rules:

Rule set	Premise	Suffix	Replacement	Example
1a	Null	sses	ss	caresses → caress
1a	Null	ies	i	ponies → poni
1b	(m>0)	eed	ee	agreed → agree feed → feed
1b	(*v*)	ed	ε	plastered → plaster bled → bled
1b	(*v*)	ing	ε	motoring → motor sing → sing
1c	(*v*)	y	i	happy → happi sky → sky
2	(m>0)	biliti	ble	sensibiliti → sensible

Morphological Analysis

Stemming: Porter Stemmer

Example:

```
Alan Mathison Turing was an English mathematician, computer scientist,
logician, cryptanalyst, philosopher, and theoretical biologist.  Turing
was highly influential in the development of theoretical computer
science, providing a formalisation of the concepts of algorithm and
computation with the Turing machine, which can be considered a model of a
general-purpose computer.  Turing is widely considered to be the father
of theoretical computer science and artificial intelligence.
```

Morphological Analysis

Stemming: Porter Stemmer

Example:

Alan Mathison Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist. Turing was highly influential in the development of theoretical computer science, providing a formalisation of the concepts of algorithm and computation with the Turing machine, which can be considered a model of a general-purpose computer. Turing is widely considered to be the father of theoretical computer science and artificial intelligence.

Morphological Analysis

Stemming: Porter Stemmer

Example:

```
Alan Mathison Ture  wa  an English mathematician, comput  scientist,
logician, cryptanalyst, philosoph  , and theoret    biologist.  Ture
wa  highli influenti  in the develop    of theoret    comput
scienc , provid   a formalis     of the concepts of algorithm and
computat   with the Ture  machin , which can be consid    a model of a
gener  -purpos  comput  .  Ture  is wide  consid    to be the father
of theoret    comput  scienc  and artifici  intellig   .
```

Morphological Analysis

Stemming: Porter Stemmer

Weaknesses of the algorithm:

- Difficult to modify:
  The effects of changes are hardly predictable.
- Tends to overgeneralize:
  `university/universe, organization/organ`
- Does not capture clear generalizations:
  `European/Europe, matrices/matrix, machine/machineri`

Morphological Analysis

Stemming: Krovetz Stemmer

The Krovetz stemmer combines a dictionary-based approach with rules:

1. Word looked up in dictionary
2. If present, replaced with word stem
3. If not present, word is checked for removable inflection suffixes
4. After removal, dictionary is checked again
5. If still not present, different suffixes are tried

Observations:

- Captures irregular cases such as `is, be, was`.
- Produces words not stems (more readable, similar to lemmatization)
- Comparable effectiveness to Porter stemmer
- Lower false positive rate, somewhat higher false negative rate

Morphological Analysis

Stemming: Stemmer Comparison

Porter Stemmer:

```
Alan Mathison Turing was an English mathematician, computer scientist,
logician, cryptanalyst, philosopher, and theoretical biologist.  Turing
was highly influential in the development of theoretical computer
science, providing a formalisation of the concepts of algorithm and
computation with the Turing machine, which can be considered a model of a
general-purpose computer.  Turing is widely considered to be the father
of theoretical computer science and artificial intelligence.
```

Krovetz Stemmer:

```
Alan Mathison Turing was an English mathematician, computer scientist,
logician, cryptanalyst, philosopher, and theoretical biologist.  Turing
was highly influential in the development of theoretical computer
science, providing a formalisation of the concepts of algorithm and
computation with the Turing machine, which can be considered a model of a
general-purpose computer.  Turing is widely considered to be the father
of theoretical computer science and artificial intelligence.
```

Morphological Analysis

Stemming: Stemmer Comparison

Porter Stemmer:

Alan Mathison Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist. Turing was highly influential in the development of theoretical computer science, providing a formalisation of the concepts of algorithm and computation with the Turing machine, which can be considered a model of a general-purpose computer. Turing is widely considered to be the father of theoretical computer science and artificial intelligence.

Krovetz Stemmer:

Alan Mathison Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist. Turing was highly influential in the development of theoretical computer science, providing a formalisation of the concepts of algorithm and computation with the Turing machine, which can be considered a model of a general-purpose computer. Turing is widely considered to be the father of theoretical computer science and artificial intelligence.

Morphological Analysis

Stemming: Stemmer Comparison

Porter Stemmer:

```
Alan Mathison Ture  wa  an English mathematician, comput  scientist,
logician, cryptanalyst, philosoph  , and theoret    biologist.  Ture
wa  highli influenti  in the develop    of theoret    comput
scienc , provid   a formalis     of the concepts of algorithm and
computat   with the Ture  machin , which can be consid    a model of a
gener  -purpos  comput  .  Ture  is wide  consid    to be the father
of theoret    comput  scienc  and artifici  intellig   .
```

Krovetz Stemmer:

```
Alan Mathison Ture  was an English mathematician, computer scientist,
logician, cryptanalyst, philosopher, and theoretical biologist.  Ture
was highly influential in the development of theoretical computer
science, providing a formalise    of the concepts of algorithm and
computation with the Ture  machine, which can be considered a model of a
general-purpose computer.  Ture  is widely considered to be the father
of theoretical computer science and artificial intelligence.
```

Morphological Analysis

Stemming: Character n-grams [McNamee et al. 2004] [McNamee et al. 2008]

A substring of length n from a longer string is called a character n-gram. A string of length $m \geq n$ has at most $(m - n) + 1$ character n-grams.

Example: `Alan Mathison Turing ...`

- 1-grams: `A, l, a, n, M, a, t, h, i, s, o, n, T, u, r, i, n, g`
- 2-grams: `Al, la, an, Ma, at, th, hi, is, so, on, Tu, ur, ri, in, ng`
- 3-grams: `Ala, lan, Mat, ath, thi, his, iso, son, Tur, uri, rin, ing`
- 4-grams: `Alan, Math, athi, this, hiso, ison, Turi, urin, ring`
- 5-grams: `Alan, Mathi, athis, thiso, hison, Turin, uring`

Morphological Analysis

Stemming: Character n-grams [McNamee et al. 2004] [McNamee et al. 2008]

A substring of length n from a longer string is called a character n-gram. A string of length $m \geq n$ has at most $(m - n) + 1$ character n-grams.

Example: `Alan Mathison Turing ...`

- 1-grams: `A, l, a, n, M, a, t, h, i, s, o, n, T, u, r, i, n, g`
- 2-grams: `Al, la, an, Ma, at, th, hi, is, so, on, Tu, ur, ri, in, ng`
- 3-grams: `Ala, lan, Mat, ath, thi, his, iso, son, Tur, uri, rin, ing`
- 4-grams: `Alan, Math, athi, this, hiso, ison, Turi, urin, ring`
- 5-grams: `Alan, Mathi, athis, thiso, hison, Turin, uring`

Use the first (or all) character n-grams for $n = 4$ or $n = 5$ as pseudo-stems of a word.

Observations:

- Language-independent; good performance for many languages.
- Well-developed stemmers yield better performance (e.g., for English).
- Large overhead in terms of vocabulary size.

Morphological Analysis

Lemmatization

Problems with stemming:

- overstemming: artificial ambiguity
  `{organization, organ}` → `organ`
- understemming: unification fails
  `European` → `european, Europe` → `europ`

Lookup of canonical / dictionary form of a word

- Approach 1: usually retrieved by long dictionary files which contain

inflected_type	**lemma_type**
European	Europe
Europe	Europe
Organizations	Organization

Problems with lookup approach:

- Getting good lemma resources
- Incomplete lemma lookup lists
- Approach 2 Morphology: many taggers also provide lemma output
  `e.g. Tree-tagger, Parzu (for German), SpaCy`