Master Language

Annotation of learner corpus data

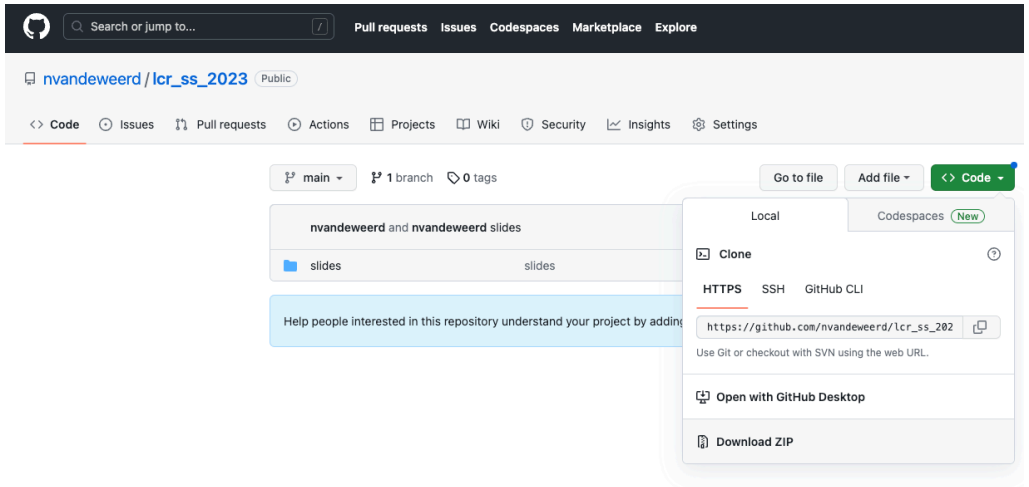
Dr. Nathan Vandeweerd

Vrije Universiteit Amsterdam

March 15th, 2024

Radboud University

Materials and sides available on GitHub



1. Click on `< > Code`.
2. Download ZIP to download all files.

Radboud University

Introduction

Radboud University

Last updated: 2024-03-01 11:48:20.618573 | Slides available here:
github.com/nvandeweerd/ml_seminar_2024_03_15

About me

🏛 Radboud University Nijmegen

🏢 Assistant professor in Language and Communication

🔍 Research interests:

- Phraseological complexity in L2 French
- Accuracy of automatic transcription software for L2 data
- Language development during study abroad
- Crowdsourcing language assessment (CLAP)

Lijdende voorwerpen
(werkwoord + naamwoord)
Bijv.
tenir + compte
'rekening + houden'

About you

✋ How many of you have experience with programming (R, python, etc.,)

Radboud University

Your goals

"the correlation between corpus building and its exploitation"

"most appropriate scientific approaches that one may use in analysing data from a learner corpus"

"I would like to know which new (or less new) software might be used to analyse the corpus. "

"how to build my own machine-learning scheme when using topic modelling?"

🔍 Check out the `topicmodels` package in R and Murakami et al. (2017). But also Shadrova (2021) for a word of caution.

"Secondly, I wish to get some theoretical knowledge & practical skills on programming"

What types of automatic annotation exist?

1. Part of Speech (POS) tagging

Radboud University

What types of automatic annotation exist?

1. Part of Speech (POS) tagging

(1) We_PPIS2 find_VV0 that_CST in_II fact_NN1 these_DD2 people_NN are_VBR the_AT most_RGT exposed_JJ to_II media_NN not_XX to_TO mension_VVI the_AT fact_NN1 that_CST there_EX is_VBZ forever_RT AIDS_NP1 awareness_NN1 campaigns_NN2 launged_VVN through_RP out_RP the_AT county_NN1._

(ICLE-TS-NOUN-0005.1)

(van Rooy, 2015: 80)

🔍 Your research:

- Spanish articles
- morpho-syntactic and syntactic labels
- morphology
- semantic, syntactic and discourse features
- verb valency patterns
- grammatical complexity
- grammatical case
- cohesion and cohesive devices
- stance features
- verb aspect
- adverbs of degree and negation
- verb phrase ellipsis
- colour terms

What types of automatic annotation exist?

1. Part of Speech (POS) tagging

(1) We_PPIS2 find_VV0 that_CST in_II fact_NN1 these_DD2 people_NN are_VBR the_AT most_RGT exposed_JJ to_II media_NN not_XX to_TO mension_VVI the_AT fact_NN1 that_CST there_EX is_VBZ forever_RT AIDS_NP1 awareness_NN1 campaigns_NN2 launged_VVN through_RP out_RP the_AT county_NN1._

(ICLE-TS-NOUN-0005.1)

(van Rooy, 2015: 80)

🔍 Your research:

- Spanish **articles**
- **morpho-syntactic and syntactic labels**
- **morphology**
- semantic, syntactic and **discourse features**
- **verb** valency patterns
- grammatical complexity
- grammatical **case**
- cohesion and **cohesive devices**
- stance features
- **verb aspect**
- **adverbs** of degree and negation
- **verb** phrase ellipsis
- colour terms

What types of automatic annotation exist?

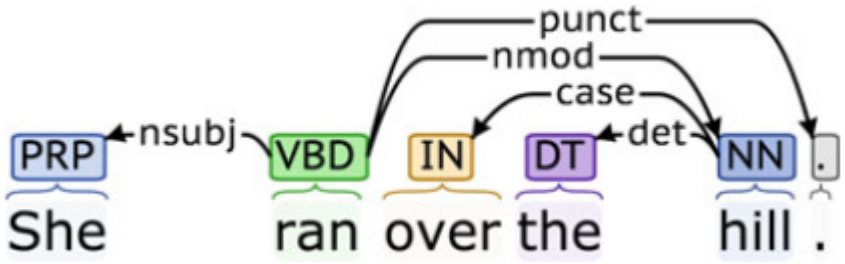
1. Part of Speech (POS) tagging
2. Syntactic parsing

🔍 Your research:

- Spanish **articles**
- **morpho-syntactic and syntactic labels**
- **morphology**
- semantic, syntactic and **discourse features**
- **verb** valency patterns
- grammatical complexity
- grammatical **case**
- cohesion and **cohesive devices**
- stance features
- **verb aspect**
- **adverbs** of degree and negation
- **verb** phrase ellipsis

What types of automatic annotation exist?

1. Part of Speech (POS) tagging
2. Syntactic parsing



(Newman and Cox, 2021: 32)

🔍 Your research:

- Spanish **articles**
- **morpho-syntactic and syntactic labels**
- **morphology**
- semantic, **syntactic** and **discourse features**
- **verb valency patterns**
- **grammatical complexity**
- grammatical **case**
- cohesion and **cohesive devices**
- stance features
- **verb aspect**
- **adverbs** of degree and negation
- **verb phrase** ellipsis

What types of automatic annotation exist?

1. Part of Speech (POS) tagging
2. Syntactic parsing
3. Semantic annotation

🔍 Your research:

- Spanish **articles**
- **morpho-syntactic and syntactic labels**
- **morphology**
- semantic, **syntactic** and **discourse features**
- **verb valency patterns**
- **grammatical complexity**
- grammatical **case**
- cohesion and **cohesive devices**
- stance features
- **verb aspect**
- **adverbs** of degree and negation
- **verb phrase** ellipsis
- colour terms

What types of automatic annotation exist?

1. Part of Speech (POS) tagging
2. Syntactic parsing
3. Semantic annotation

(6) a. The_Z5 ending_T2- of_Z5 the_Z5 poem_Q3 may_A7+ seem_A8 to_Z5 be_A3+ contradictory_A6.1- because_Z5/A2.2 both_N5 girls_S2.1f marry_S4 and_Z5 have_A9+ children_S2mf/T3- ;_PUNC thereby_Z5 filling_N5.1+ the_Z5 traditional_S1.1.1 female_S2.1 role_I3.1 ._PUNC
b. at_T1.1.2[i165.3.1 a_T1.1.2[i165.3.2 time_T1.1.2[i165.3.3

(Newman and Cox, 2021: 35)

🔍 Your research:

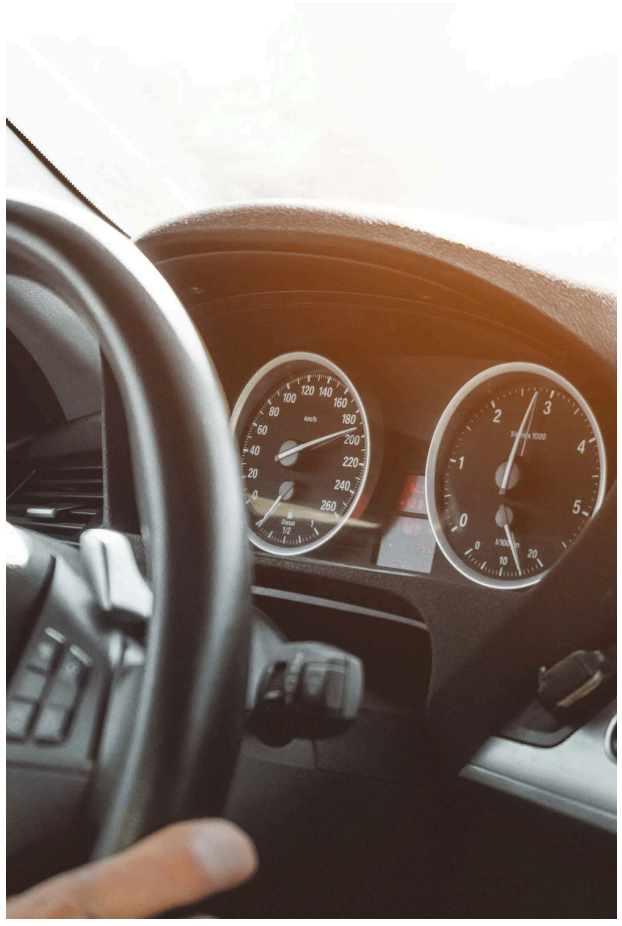
- Spanish **articles**
- **morpho-syntactic and syntactic labels**
- **morphology**
- **semantic**, **syntactic** and **discourse features**
- **verb valency patterns**
- **grammatical complexity**
- grammatical **case**
- cohesion and **cohesive devices**
- **stance features**
- **verb aspect**
- **adverbs** of degree and negation
- **verb phrase** ellipsis

What types of automatic annotation exist?

1. Part of Speech (POS) tagging
2. Syntactic parsing
3. **Semantic annotation**

Radboud University

Why use automatic annotation?



Radboud University

Today's session

- Text-preprocessing
- POS-tagging and lemmatization
- Hands on activity: *POS-tagging and lemmatization*

☕ *Coffee break* (15.30)

- Syntactic annotation
- Hands on activity: *Syntactic annotation*
- The reliability of automatic tools
- Final remarks

Radboud University

How this workshop will work

Option 1. Webtools/Excel

Option 2. R

Radboud University

Text-preprocessing

Radboud University

Last updated: 2024-03-01 11:48:20.618573 | Slides available here:
github.com/nvandeweerd/ml_seminar_2024_03_15

What do you notice about this text?

I agree that successful people try <e>news</e> things and take risk rather than only doing what they already know how to do well,for these reasons; By trying new things allow you to be curious to know how someone did it and you will find out how to do it too, that way it make you make a research. By trying new things allow you to be positive in your mind and to have a great desire to succed no matter how difficult is the situation. By trying new things you no that you should be openminded,go through disscussion with people who have done the same thing to learn their ways of doing thing, you should meet or have conversation with a lot of these people in other to learn from their experiences. By trying new things you take a big risks, like in " french we say if you don ’t risk you don't have anything",risk in a goog way to takle something. We never know if we might succed or not the only way to do it is to risk.Since we do not loose anything when we risk.ed As for me doing the same thing every day become boring,i can say is a waste of energy and time, To conclude,people who succed try new things and take risks rather than only doing what they already know how to do well.

⚠ Ignore spelling mistakes for the time being...

What do you notice about this text?

I agree that successful people try <e>news</e> things and take risk rather than only doing what they already know how to do well,for these reasons; By trying new things allow you to be curious to know how someone did it and you will find out how to do it too, that way it make you make a research. By trying new things allow you to be positive in your mind and to have a great desire to succed no matter how difficult is the situation. By trying new things you no that you should be openminded,go through disscussion with people who have done the same thing to learn their ways of doing thing, you should meet or have conversation with a lot of these people in other to learn from their experiences. By trying new things you take a big risks, like in " french we say if you don 't risk you don't have anything",risk in a goog way to takle something. We never know if we might succed or not the only way to do it is to risk.Since we do not loose anything when we risk.ed As for me doing the same thing every day become boring,i can say is a waste of energy and time, To conclude,people who succed try new things and take risks rather than only doing what they already know how to do well.

(Non-exhaustive) list of things that can cause issues for automatic tools:

- (Inconsistent) file encoding
- Spacing
  - Lack of space between words
  - Unnecessary space between words
  - Double space between words
- 'Stylized' apostrophes or quotation marks
- Accented characters (e.g., *à*)
- Special characters (e.g., *, %, |)
- Inconsistent spelling rules (e.g., email/e-mail)
- Coding schemes (e.g., XML)

What do you notice about this text?

I agree that successful people try <e>news</e> things and take risk rather than only doing what they already know how to do well,for these reasons; By trying new things allow you to be curious to know how someone did it and you will find out how to do it too, that way it make you make a research. By trying new things allow you to be positive in your mind and to have a great desire to succed no matter how difficult is the situation. By trying new things you no that you should be openminded,go through disscussion with people who have done the same thing to learn their ways of doing thing, you should meet or have conversation with a lot of these people in other to learn from their experiences. By trying new things you take a big risks, like in " french we say if you don 't risk you don't have anything",risk in a goog way to takle something. We never know if we might succed or not the only way to do it is to risk.Since we do not loose anything when we risk.ed As for me doing the same thing every day become boring,i can say is a waste of energy and time, To conclude,people who succed try new things and take risks rather than only doing what they already know how to do well.

(Non-exhaustive) list of things that can cause issues for automatic tools:

- (Inconsistent) file encoding
- Spacing
  - Lack of space between words
  - Unnecessary space between words
  - Double space between words
- 'Stylized' apostrophes or quotation marks
- Accented characters (e.g., *à*)
- Special characters (e.g., *, %, |)
- Inconsistent spelling rules (e.g., email/e-mail)
- Coding schemes (e.g., XML)

⚠ Importance of **knowing your corpus**...

Methods of text cleaning/preprocessing

+ Time intensive

− Replicable

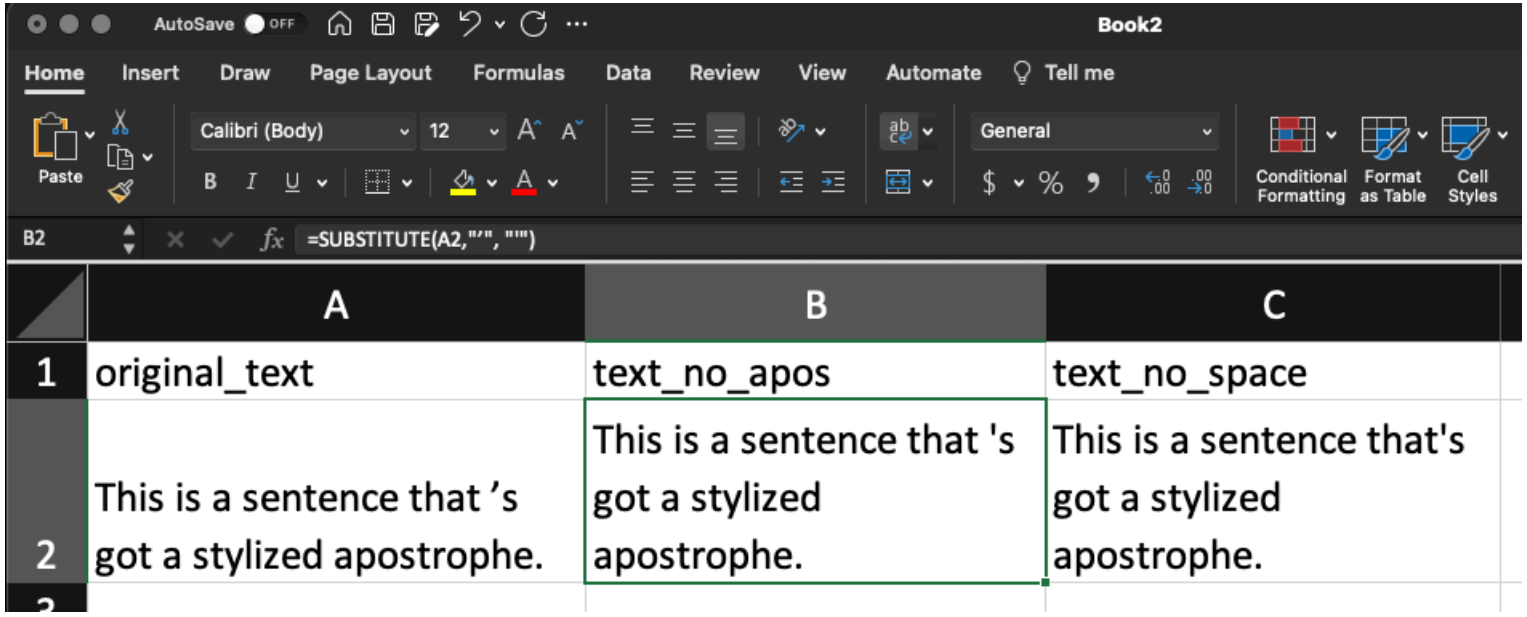
1. Manually
2. Semi-manually using search and replace (e.g., in Excel)
3. Semi-manually using regular expressions (e.g., in a text editor)
4. Semi-automatically using a script with regular expressions (e.g., in R or python)

− Time intensive

+ Replicable

Radboud University

Excel



`SUBSTITUTE()` function (For more details see here)

```
=SUBSTITUTE(text, old_text, new_text, [instance_num])
=SUBSTITUTE(A2,"’", "'")
```

Radboud University

Notepad++ (PC) / Textmate (Mac)

Notepad + +

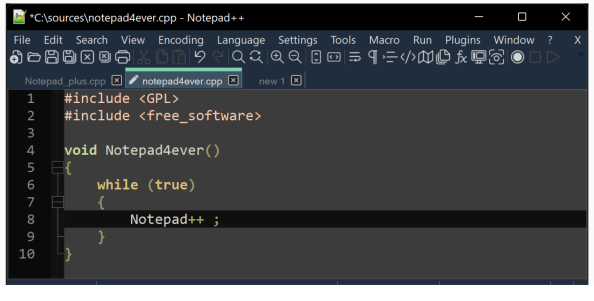
Current Version 8.5.4

- Home
- Download
- News
- Online Help
- Resources
- RSS
- Donate
- Author

What is Notepad++

Notepad++ is a free (as in “free speech” and also as in “free beer”) source code editor and Notepad replacement that supports several languages. Running in the MS Windows environment, its use is governed by GNU General Public License.

Based on the powerful editing component Scintilla, Notepad++ is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size. By optimizing as many routines as possible without losing user friendliness, Notepad++ is trying to reduce the world carbon dioxide emissions. When using less CPU power, the PC can throttle down and reduce power consumption, resulting in a greener environment.



Textmate

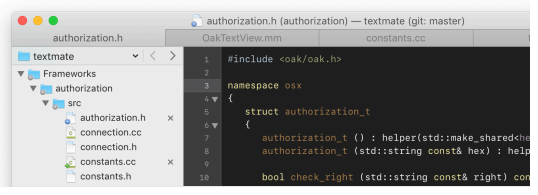
MacroMates — TextMate ▾ News Blog Support

TextMate for macOS

Powerful and customizable text editor with support for a huge list of programming languages and developed as open source.

Download TextMate 2.0

Requires macOS 10.12 or later.



Multiple Carets

Making multiple changes at once, swapping pieces of code, and a lot more is made trivial with TextMate’s easy way to add multiple insertion points.

File Search

Select what you want to search, what you want to search for, and TextMate will present the results in a way that makes it easy to jump between matches, extract matched text, or preview desired replacements.

Version Control

See what files have changes in the file browser view, what lines have changes in the editor view, bring up a diff of the current file’s changes, commit a subset, TextMate supports it all for all the major version control systems.

Scoped Settings

One file mixing languages? Projects using different build systems? Third party code with different formatting preferences? TextMate can handle it all by associating detailed scope selectors with key shortcuts, settings, etc.

Commands

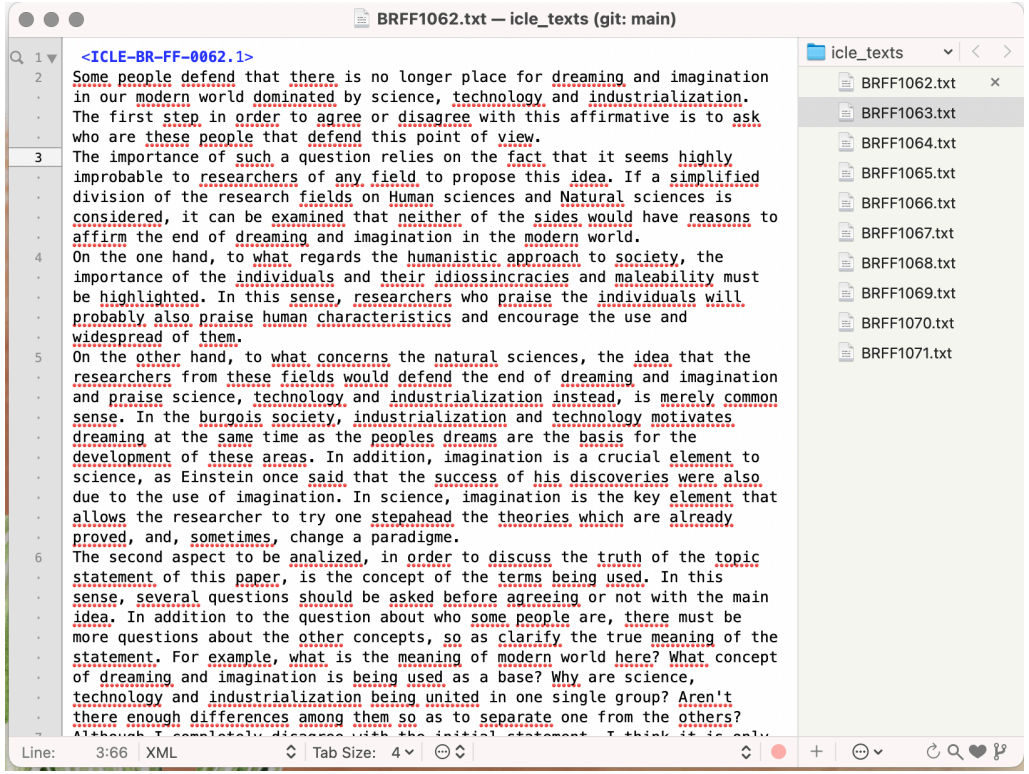
The UNIX underpinnings of macOS allows custom actions to be written in any language that can work with stdin, stdout, and environment variables, and for complex interactions TextMate expose both WebKit and a dialog framework for Mac-native or HTML-based interfaces.

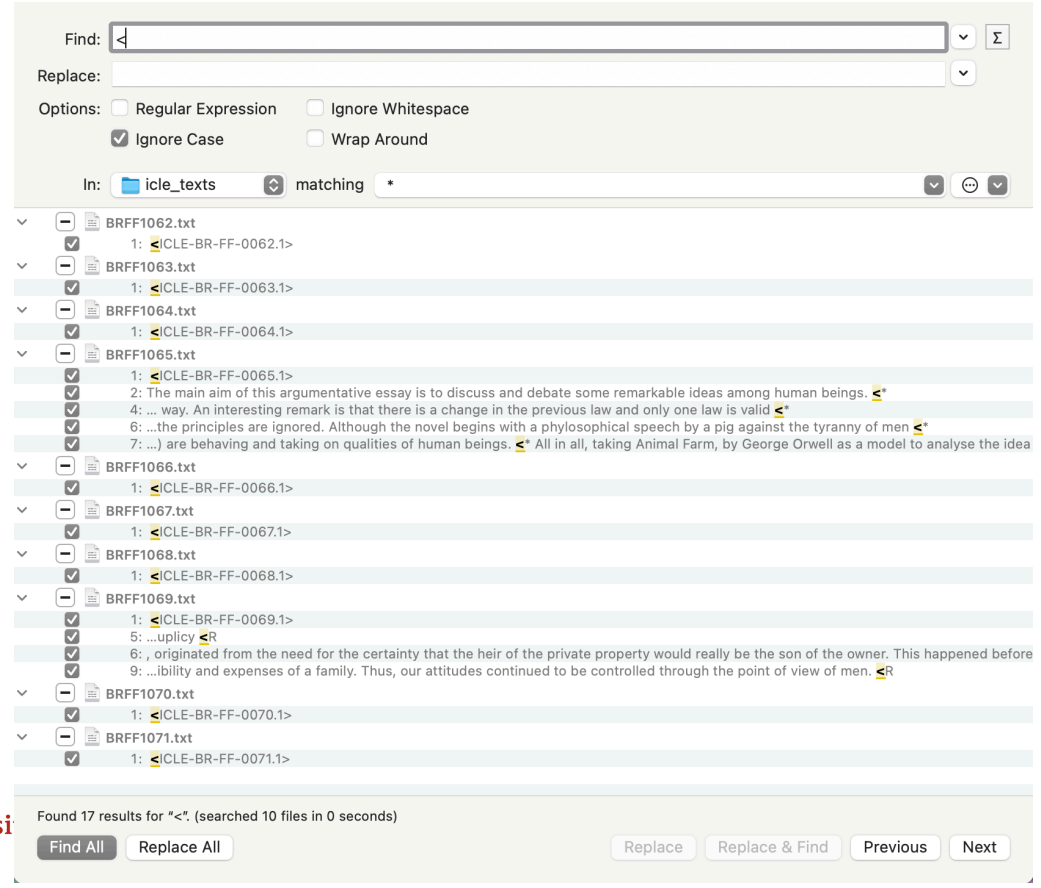
Snippets

Commonly used pieces of text or code can be turned into snippets with placeholders, transformations, and more, to have them adapt according to the context in which they are used.

Radboud University

Textmate (Mac)





Radboud Universi

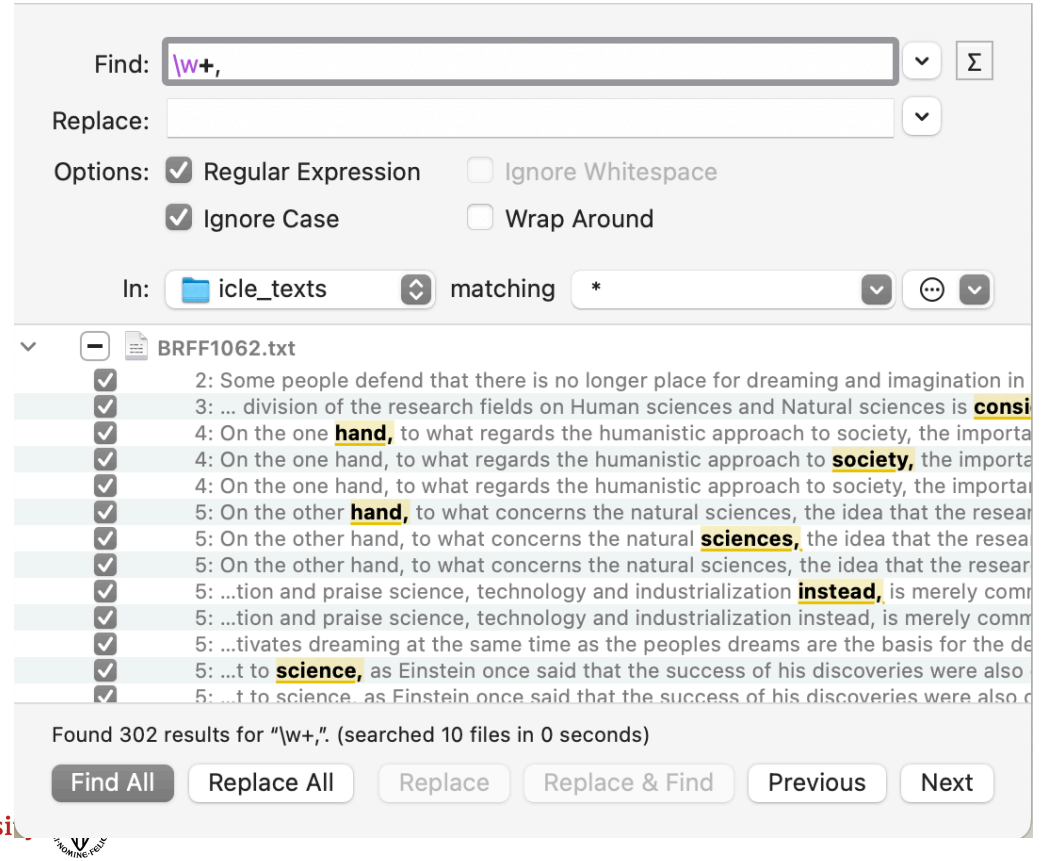
Regular expressions (RegEx)

Special patterns that allow you to search for specific sequences.

Examples:

`\w`: Returns a word character (A-Z, a-z, _)
`\s`: Returns a space character
`+`: Returns one or more of the previous character
`?`: Returns zero or more of the previous character
`.`: Returns any single character

See here for a website where you can test regular expressions.



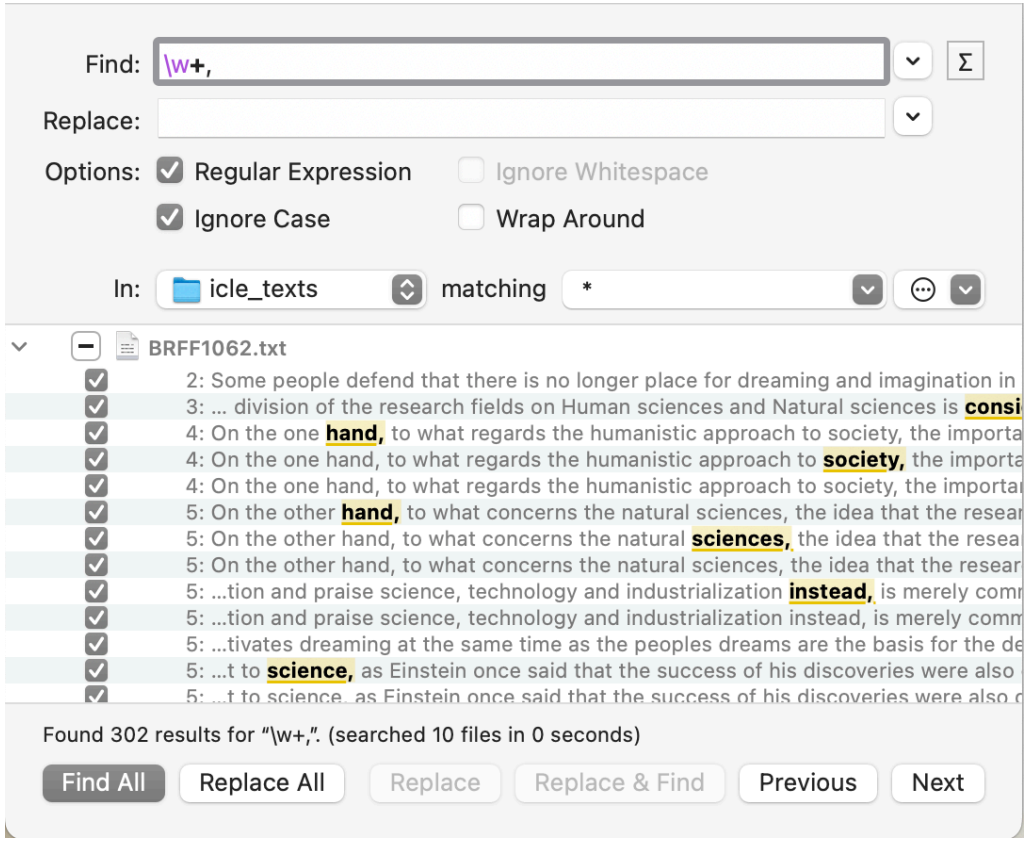
Regular expressions (RegEx)

Special patterns that allow you to search for specific sequences.

Examples:

`\w`: Returns a word character (A-Z, a-z, _)
`\s`: Returns a space character
`+`: Returns one or more of the previous character
`?`: Returns zero or more of the previous character
`.`: Returns any single character

See here for a website where you can test regular expressions.



Try this: What regular expression could be used to remove the 'headers' from the ICLE texts? (e.g., `<ICLE-BR-FF-0062.1>`). Be careful

Some possible solutions

How to find…`<ICLE-BR-FF-0062.1>`

`<\w{4}-\w{2}-\w{2}-\w{4}\.\d>`

/ <\w{4}-\w{2}-\w{2}-\w{4}\.\d> / gm
< matches the character < with index 60_{10} ($3C_{16}$ or 74_{8}) literally (case sensitive)
\w matches any word character (equivalent to [a-zA-Z0-9_])
{4} matches the previous token exactly 4 times
– matches the character – with index 45_{10} ($2D_{16}$ or 55_{8}) literally (case sensitive)
\w matches any word character (equivalent to [a-zA-Z0-9_])
{2} matches the previous token exactly 2 times
– matches the character – with index 45_{10} ($2D_{16}$ or 55_{8}) literally (case sensitive)
\w matches any word character (equivalent to [a-zA-Z0-9_])
{2} matches the previous token exactly 2 times
– matches the character – with index 45_{10} ($2D_{16}$ or 55_{8}) literally (case sensitive)
\w matches any word character (equivalent to [a-zA-Z0-9_])
{4} matches the previous token exactly 4 times
\. matches the character \. with index 46_{10} ($2E_{16}$ or 56_{8}) literally (case sensitive)
\d matches a digit (equivalent to [0-9])
> matches the character > with index 62_{10} ($3E_{16}$ or 76_{8}) literally (case sensitive)

`<[^>]+>`

/ <[^>]+> / gm
< matches the character < with index 60_{10} ($3C_{16}$ or 74_{8}) literally (case sensitive)
Match a single character not present in the list below [^>]
+ matches the previous token between one and unlimited times, as many times as possible, giving back as needed (greedy)
> matches the character > with index 62_{10} ($3E_{16}$ or 76_{8}) literally (case sensitive)
> matches the character > with index 62_{10} ($3E_{16}$ or 76_{8}) literally (case sensitive)
Global pattern flags
g modifier: global. All matches (don't return after first match)
m modifier: multi line. Causes ^ and $ to match the begin/end of each line (not only begin/end of string)

Radboud University

R

```
library(stringr); library(dplyr)
```

```
text <- "This is a sentence that ’s got a stylized apostrophe."

text %>%
  # replace stylized apostrophes
  str_replace_all("’", "'") %>%
  # remove spaces before apostrophes
  # if they are:
  # - preceded by the beginning of a string, space or final punctuation
  # - followed by only one or two letters (e.g., don't, they're)
  # - and then followed by either a space, final punctuation or end of string
  str_replace_all("[\\s\\.\\?\\!^]'([^\\s]{1,2}[\\s\\.\\?\\!$])", "'\\1")
```

```
## [1] "This is a sentence that's got a stylized apostrophe."
```


⚠ Note that you need double slashes for special or escape characters in R (\\).

With both approaches

⚠ Test (and re-test) your pre-processing pipeline!

- Be careful of inadvertent changes (especially when using regular expressions)
  E.g. *don't* vs. *he said 'don't worry'*
- Be aware the different tools require different approaches (e.g., contractions separate or apart)
- Never edit the original corpus files!
- Keep track of any changes you make

```
text <- "'He ’ll be comin' round the 'mountain' when he comes,' I said."

text %>%
  str_replace_all("’", "'") %>%
  str_replace_all("[\\s\\.\\?\\!^]'([^\\s]{1,2}[\\s\\.\\?\\!$])", "'\\1")
```

```
## [1] "'He'll be comin' round the 'mountain' when he comes,' I said."
```

So why is pre-processing so important?

```
 [1] "I"                          "agree"
 [3] "that"                       "successful"
 [5] "people"                     "try"
 [7] "&lt;e&gt;news&lt;/e&gt;"    "things"
 [9] "and"                        "take"
[11] "risk"                       "rather"
[13] "than"                       "only"
[15] "doing"                      "what"
[17] "they"                       "already"
[19] "know"                       "how"
[21] "to"                         "do"
[23] "well,for"                   "these"
[25] "reasons;"                   ""
[27] "By"                         "trying"
[29] "new"                        "things"
```

Garbage in...garbage out

Tokenization

Token: "the smallest unit of a corpus" (Krause, Lüdeling, Odebrecht, and Zeldes, 2012: 2)
= words, numbers, punctuation marks, quotation marks etc.
= syllable, phoneme, etc...

Easiest method (for English):

- split tokens at spaces
- split sentences at periods (.), exclamation marks (!) or question marks (?)

What problems do you see with this method?

Potential problems:

- clitics (isn't, ain't)
- missing whitespace
- periods (etc., U.S.A., fig.)
- ordinal numbers
- multiword expressions (New York-based, 10 000, as well as)
- word-internal punctuation (relationship(s), "Rambo"-type)
- (de)hypthenation (preprocessing vs. pre-processing)
- quoted speech (“You still don’t have an accountant?” Ellis said.)
- ideographic languages (e.g., Chinese, Japanese)

🔍 For more information about tokenization see Zeldes (2020) and Schmid (2008).

POS-tagging and lemmatization

Radboud University

Last updated: 2024-03-01 11:48:20.618573 | Slides available here:
github.com/nvandeweerd/ml_seminar_2024_03_15

Overview

Each word in the corpus is 'tagged' (labelled) with information about its grammatical category.

🔧 Under the hood:

1. All tokens with unambiguous POS labels are assigned tags (e.g., on the basis of a dictionary)
2. Contextual features (e.g., surrounding tags, morphological endings) used in a statistical model to predict the tags of ambiguous items

(Kyle, 2021: 6)

She 🏷 `pronoun`
sells 🏷 `verb`
seashells 🏷 `noun`
by 🏷 `preposition`
the 🏷 `determiner`
seashore 🏷 `noun`
. 🏷 `punctuation`

Tagsets for English

- CLAWS (Constituent Likelihood Automatic Word-tagging System)
  - CLAWS 5 = 60 tags
  - CLAWS 7 = 160 tags
- PENN Treebank Tagset
- BNC Tagset
- Universal POS tags

Different tagsets, (subtly) different theories of grammar

Table 2.1 Four tagging solutions for English *rid*

	I am now completely rid of such things	*You are well rid of him*	*I got rid of the rubbish*
CLAWS7 tagger[a]	Past participle	Past participle	Past participle
Infogistics[b]	Verb base	Verb base	Past participle
FreeLing[c]	Adjective	Verb base	Past participle
(Brill-based) GoTagger[d]	Adjective	Adjective	Adjective

(Newman and Cox, 2021: 21)

Select the tagset that is right for your data/research question.

Example

(1) We_PPIS2 find_VV0 that_CST in_II fact_NN1 these_DD2 people_NN are_VBR the_AT most_RGT exposed_JJ to_II media_NN not_XX to_TO mension_VVI the_AT fact_NN1 that_CST there_EX is_VBZ forever_RT AIDS_NP1 awareness_NN1 campaigns_NN2 launged_VVN through_RP out_RP the_AT county_NN1._

(ICLE-TS-NOUN-0005.1)

(2) We find that in fact these people are the most exposed to media not to mension the fact that there is forever AIDS awareness campaigns launged through out the county.

(van Rooy, 2015: 80-81)

Verbs: launged_VVD, find_VV0, is_VBZ, are_VBR

Nouns: media_NN, fact_NN1, AIDS_NN1, county_NN1, awareness_NN1, people_NN, mension_NN1, campaigns_NN2, fact_NN1

= CLAWS C7 Tagset

? What does the tag PPIS2 refer to?

? What do you notice about the learner errors? (e.g., 'mension', 'is awareness campaigns')

Types of output: *Horizontal*

```
We_PPIS2 find_VV0 that_CST in_II fact_NN1 these_DD2 people_NN are_VBR the_AT most_RGT
exposed_JJ to_II media_NN not_XX to_II mension_NN1 the_AT fact_NN1 that_CST there_EX
is_VBZ forever_RT AIDS_NN1 awareness_NN1 campaigns_NN2 launged_VVD through_RP out_RP
the_AT county_NN1 ._.
```

Types of output: *(Pseudo)-XML*

```
<w id="2.1" pos="PPIS2">We</w>
<w id="2.2" pos="VV0">find</w>
<w id="2.3" pos="CST">that</w>
<w id="2.4" pos="II">in</w>
<w id="2.5" pos="NN1">fact</w>
<w id="2.6" pos="DD2">these</w>
<w id="2.7" pos="NN">people</w>
<w id="2.8" pos="VBR">are</w>
<w id="2.9" pos="AT">the</w>
<w id="2.10" pos="RGT">most</w>
<w id="2.11" pos="JJ">exposed</w>
<w id="2.12" pos="II">to</w>
<w id="2.13" pos="NN">media</w>
<w id="2.14" pos="XX">not</w>
<w id="2.15" pos="II">to</w>
<w id="2.16" pos="NN1">mension</w>
<w id="2.17" pos="AT">the</w>
<w id="2.18" pos="NN1">fact</w>
<w id="2.19" pos="CST">that</w>
<w id="2.20" pos="EX">there</w>
```

Radboud University

Types of output: *Vertical*

```
##    idx sntc   token tag lttr      wclass
## 1    1    1      We  PP    2     pronoun
## 2    2    1    find VBP    4        verb
## 3    3    1    that  IN    4 preposition
## 4    4    1      in  IN    2 preposition
## 5    5    1    fact  NN    4        noun
## 6    6    1   these  DT    5  determiner
## 7    7    1  people NNS    6        noun
## 8    8    1     are VBP    3        verb
## 9    9    1     the  DT    3  determiner
## 10  10    1    most RBS    4      adverb
## 11  11    1 exposed VBN    7        verb
## 12  12    1      to  TO    2          to
## 13  13    1   media NNS    5        noun
## 14  14    1     not  RB    3      adverb
## 15  15    1      to  TO    2          to
## 16  16    1 mension  NN    7        noun
## 17  17    1     the  DT    3  determiner
## 18  18    1    fact  NN    4        noun
## 19  19    1    that  IN    4 preposition
## 20  20    1   there  EX    5 existential
```

Lemmatization

Lemma: "a 'base form', which provides a level of abstraction from any inflection that might appear in the original orthographic word."

(Newman and Cox, 2021: 29)

(2) We find that in fact these people are the most exposed to media not to mension the fact that there is forever AIDS awareness campaigns launged through out the county.

we find that in fact these people be the most expose to medium not to mension the fact that there be forever AIDS awareness campaign launged through out the county .

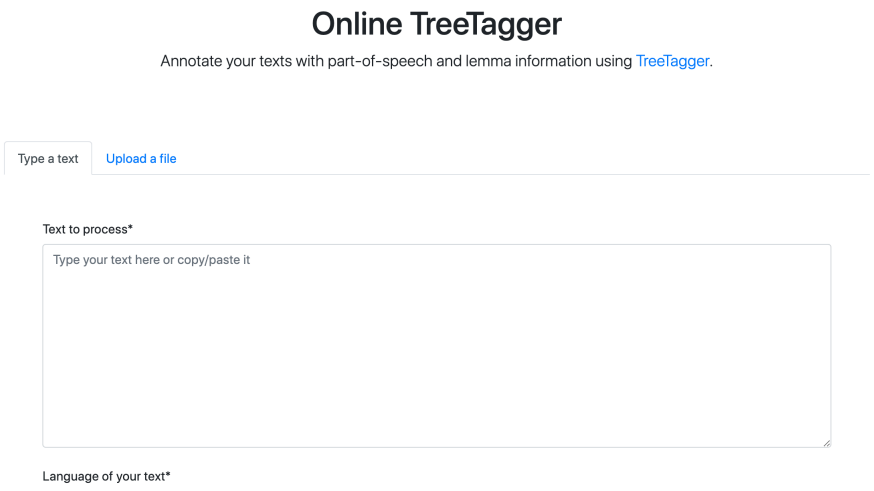
Why might this be useful?

Example

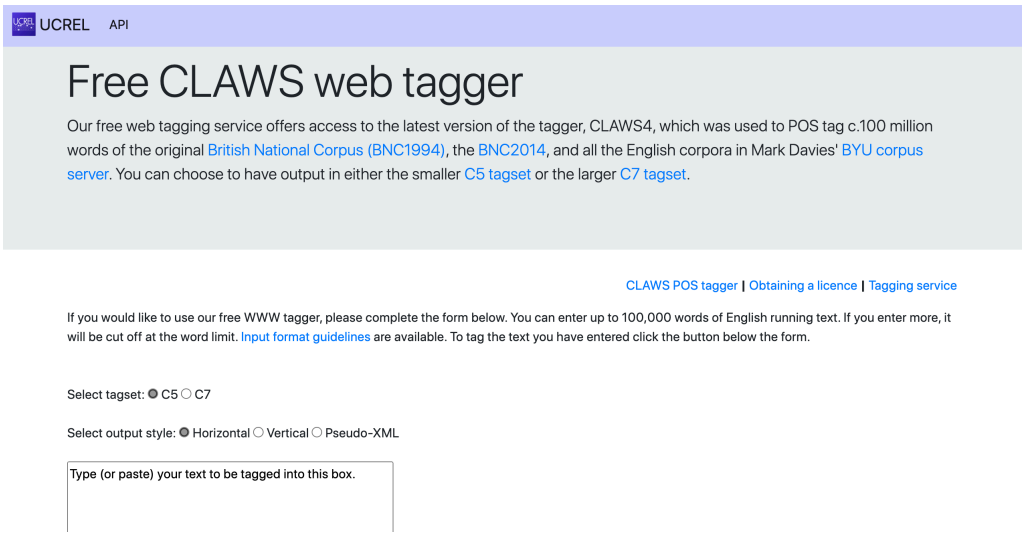
```
##    idx sntc   token tag     lemma lttr      wclass
## 1    1    1      We  PP        we    2     pronoun
## 2    2    1    find VBP      find    4        verb
## 3    3    1    that  IN      that    4 preposition
## 4    4    1      in  IN        in    2 preposition
## 5    5    1    fact  NN      fact    4        noun
## 6    6    1   these  DT     these    5  determiner
## 7    7    1  people NNS    people    6        noun
## 8    8    1     are VBP        be    3        verb
## 9    9    1     the  DT       the    3  determiner
## 10  10    1    most RBS      most    4      adverb
## 11  11    1 exposed VBN    expose    7        verb
## 12  12    1      to  TO        to    2          to
## 13  13    1   media NNS    medium    5        noun
## 14  14    1     not  RB       not    3      adverb
## 15  15    1      to  TO        to    2          to
## 16  16    1 mension  NN <unknown>    7        noun
## 17  17    1     the  DT       the    3  determiner
## 18  18    1    fact  NN      fact    4        noun
## 19  19    1    that  IN      that    4 preposition
## 20  20    1   there  EX     there    5 existential
```

Webtools

TreeTagger



CLAWS



Radboud University

R

```r
library(koRpus); library(koRpus.lang.en)

file <- "data/example_texts/ICLE-TS-NOUN-0

treetag(
  file,
  treetagger="manual",
  lang="en",
  TT.options=list(
    # Change this to the location where Tr
    path="/Applications/tree-tagger",
    preset="en"
  ),
  doc_id=basename(file)
)
```

```r
library(koRpus); library(koRpus.lang.en)

text <- "This is a sentence."

treetag(
  text,
  # Need to add the 'format' argument.
  format = "obj",
  treetagger="manual",
  lang="en",
  TT.options=list(
    # Change this to the location where Tr
    path="/Applications/tree-tagger",
    preset="en"
  ),
  # Not necessary anymore
  #doc_id=basename(file)
)
```

R

```r
library(koRpus); library(koRpus.lang.en)

file <- "data/example_texts/ICLE-TS-NOUN-0

treetag(
  file,
  treetagger="manual",
  lang="en",
  TT.options=list(
    # Change this to the location where Tr
    path="/Applications/tree-tagger",
    preset="en"
  ),
  doc_id=basename(file)
)
```

⚠ Note: For this to work, both TreeTagger and the appropriate tagsets must first be installed

```r
library(koRpus); library(koRpus.lang.en)

text <- "This is a sentence."

treetag(
  text,
  # Need to add the 'format' argument.
  format = "obj",
  treetagger="manual",
  lang="en",
  TT.options=list(
    # Change this to the location where Tr
    path="/Applications/tree-tagger",
    preset="en"
  ),
  # Not necessary anymore
  #doc_id=basename(file)
)
```

Activity 1: POS-tagging

Webtools/Excel Option

Open `activity_01_pos-tagging.docx` and follow the instructions.

R Option

Open `activity_01_pos-tagging.R` and follow the instructions.

Remember that all materials and sides available on GitHub.

1. Click on `< > Code`.
2. Download ZIP to download all files.

d University

Activity 1: POS-tagging

Webtools/Excel Option

Open `activity_01_pos-tagging.docx` and follow the instructions.

R Option

Open `activity_01_pos-tagging.R` and follow the instructions.

Remember that all materials and are sides available on GitHub

1. Click on `< > Code`.
2. Download ZIP to download all files.

Questions

1. What do you notice about the ICLE texts? What pre-processing steps (if any) might be necessary before using automatic annotation tools?
2. R: What 'special characters' do you notice?
3. R: How many 'words' and 'sentences' does BRFF1065.txt contain?
4. R: What is the average sentence length of BRFF1065.txt?
5. WT: How many tokens were tagged with CLAWS (v5)?
6. WT: What is the tag for the base form of a lexical verb in the C5 tagset?
7. WT: What is the tag for the base form of a lexical verb in the C7 tagset?
8. WT: What is meant by '[VVZ/86] NN2/14'?
9. WT: What does the code '\@card\@' mean?
10. How many adjectives (JJ) are there in BRFF1065.txt?
11. How many common nouns (NN, NNS) are there in BRFF1065.txt?
12. How should you best deal with unknown lemmas?
13. What are the most frequent verb lemmas in the corpus?

Syntactic parsing

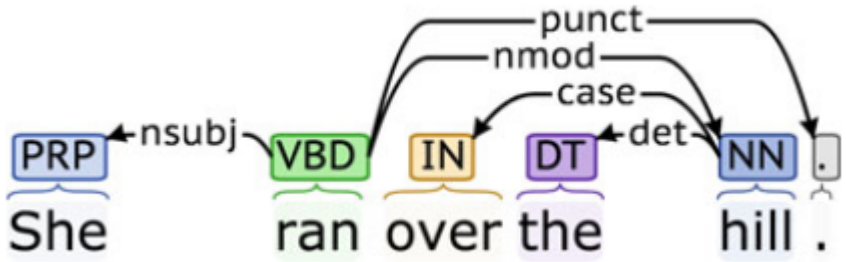
Last updated: 2024-03-01 11:48:20.618573 | Slides available here:
github.com/nvandeweerd/ml_seminar_2024_03_15

Overview

Labels of the syntactic connections between words (heads and dependents)

🔧 Under the hood:

1. Texts are POS-tagged.
2. POS tags used in conjunction with phrase-structure rules (generated from training algorithms on large corpora) to generate several possible *parse trees* for each sentence.
3. Statistical or machine learning algorithms are used to select the most probable parse tree for a given sentence.



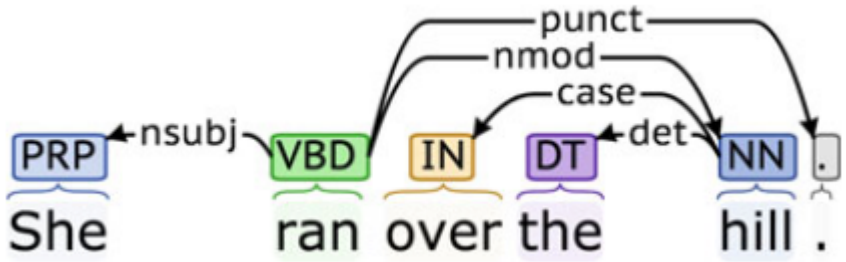
(Kyle, 2021: 7)

Overview

Labels of the syntactic connections between words (heads and dependents)

🔧 Under the hood:

1. Texts are POS-tagged.
2. POS tags used in conjunction with phrase-structure rules (generated from training algorithms on large corpora) to generate several possible *parse trees* for each sentence.
3. Statistical or machine learning algorithms are used to select the most probable parse tree for a given sentence.



(3) Parse of *She ran over the hill.*

```
(ROOT
  (S
    (NP (PRP She))
    (VP (VBD ran)
      (PP (IN over)
        (NP (DT the) (NN hill))))
    (. .)))
```

(Kyle, 2021: 7)

Dependency models for English

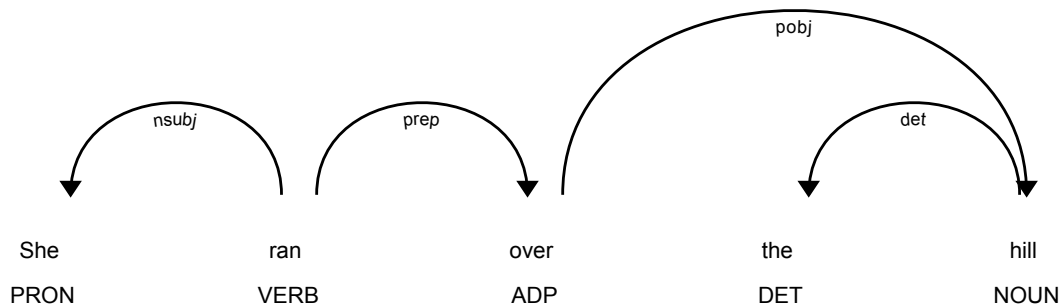
- Stanford CoreNLP
- spaCy
- Universal dependencies

Different models, (subtly) different theories of grammar

Types of output: *CoNLL*

= Conference on Natural Language Learning

	doc_id	sentence_id	token_id	token	lemma	pos	head_token_id	dep_rel
1	text1	1	1	She	she	PRON	2	nsubj
2	text1	1	2	ran	run	VERB	2	ROOT
3	text1	1	3	over	over	ADP	2	prep
4	text1	1	4	the	the	DET	5	det
5	text1	1	5	hill	hill	NOUN	3	pobj
6	text1	1	6	.	.	PUNCT	2	punct



Types of output: *FoLiA XML*

= Format for Linguistic Annotation

```
<p xml:id="example.p.1">
 <t>He hits Mr. Smith. That came quite expected!</t>
 <s xml:id="example.p.1.s.1">
  <t offset="0">He hits Mr. Smith.</t>
  <w xml:id="example.p.1.s.1.w.1"><t offset="0">He</t></w>
  <w xml:id="example.p.1.s.1.w.2"><t offset="3">hits</t>
    <morphology>
        <morpheme class="lexical" function="lexical">
         <t offset="0">hit</t>
        </morpheme>
        <morpheme class="suffix" function="inflectional">
         <t offset="3">s</t>
        </morpheme>
    </morphology>
   </w>
  <w xml:id="example.p.1.s.1.w.3"><t offset="8">Mr.</t></w>
  <w xml:id="example.p.1.s.1.w.4" space="no"><t offset="10">Smith</t></w>
  <w xml:id="example.p.1.s.1.w.5"><t offset="15">.</t></w>
 </s>
  ..
</p>
```

Radboud University

Types of output: *json*

```
 [1] "{"
 [2] "  \"text\": \"She ran over the hill\","
 [3] "  \"ents\": [],"
 [4] "  \"sents\": ["
 [5] "    {"
 [6] "      \"start\": 0,"
 [7] "      \"end\": 21"
 [8] "    }"
 [9] "  ],"
[10] "  \"tokens\": ["
[11] "    {"
[12] "      \"id\": 0,"
[13] "      \"start\": 0,"
[14] "      \"end\": 3,"
[15] "      \"tag\": \"PRP\","
[16] "      \"pos\": \"PRON\","
[17] "      \"morph\": \"Case=Nom|Gender=Fem|Number=Sing|Person=3|PronType=Prs\","
[18] "      \"lemma\": \"she\","
[19] "      \"dep\": \"nsubj\","
[20] "      \"head\": 1"
[21] "    },"
```

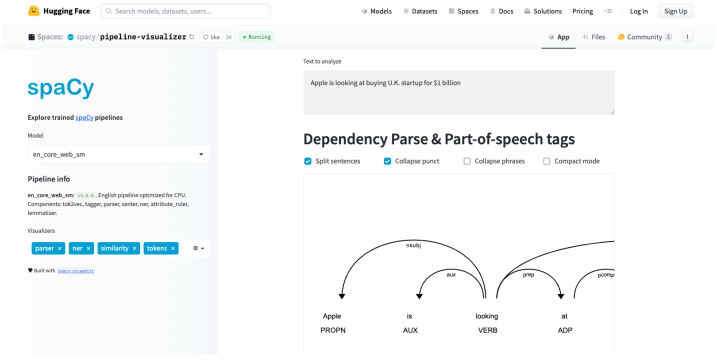
Can be converted into CSV/Excel format.

Webtools

CoreNLP

⚠ Note: May not be available due to planned outage from June 24th-July 3rd.

Hugging Face spaCy visualizer



Radboud University

R

```r
library(spacyr)

spacy_initialize(model = "en_core_web_sm")

text <- "She ran over the hill."

spacy_parse(text, dependency = TRUE)
```

⚠ Note: For this to work, spaCy must be installed locally on your computer but this can be done within the spaCy package using the `spacy_install()` function.

💡 See this vignette for more information about the `spacyr` package.

Activity 2: Syntactic parsing

Webtools/Excel Option

Open `activity_02_parsing.docx` and follow the instructions.

R Option

Open `activity_02_parsing.R` and follow the instructions.

Remember that all materials and sides available on GitHub.

1. Click on `< > Code`.
2. Download ZIP to download all files.

d University

Activity 2: Syntactic parsing

Webtools/Excel Option

Open `activity_02_parsing.docx` and follow the instructions.

R Option

Open `activity_02_parsing.R` and follow the instructions.

Remember that all materials and sides available on GitHub.

1. Click on `< > Code`.
2. Download ZIP to download all files.

Questions

1. Which word is the final period dependent on?
2. What type of dependency relationships are marked by 'amod' and 'dobj'?
3. How many amod dependencies are there in this text?
4. What adjective modifies the word 'difference'?
5. What is the object of the verb 'control'?
6. R: What is the most frequency dependency relation in the corpus?
7. R: What is the average length of noun phrases in the corpus?

d University

The reliability of automatic tools

Radboud University

Last updated: 2024-03-01 11:48:20.618573 | Slides available here:
github.com/nvandeweerd/ml_seminar_2024_03_15

The effect of learner errors

(4) a. . . . like body that will **loss** its **ballence** when one of two **organe** get **dameged**, a country also can **loss** its **ballence** when part of its people live in sadness.

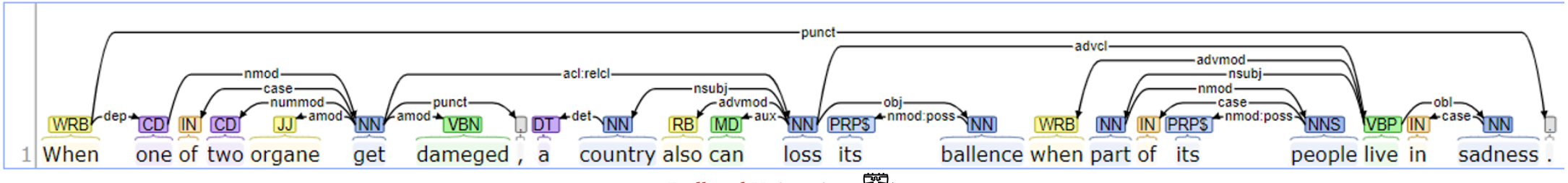
b. . . . **peoples** who I met is not good . . .

c. So when I **admit** to korea university, I decide what i find my own way.

d. Also, the people in it very friendly.

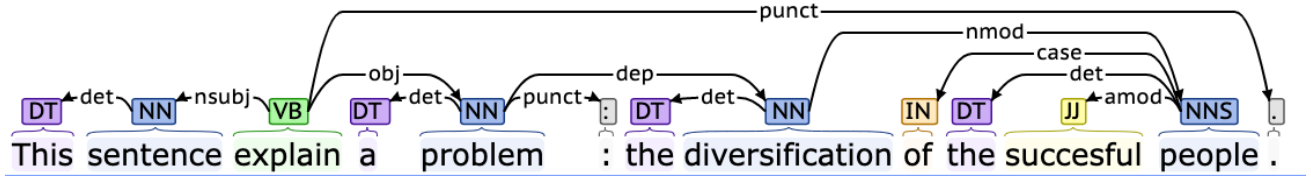
(Ragheb and Dickinson, 2012)

What problems do you see here?



Radboud University

Not all errors are equally as problematic



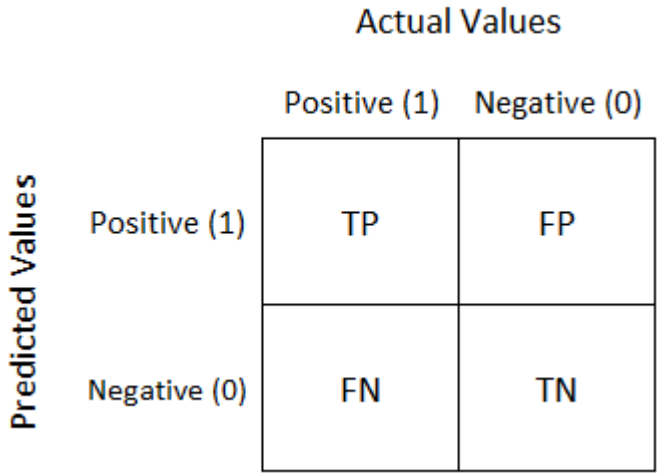
(Ragheb and Dickinson, 2012)

Evaluating the *reliability* of automatic tools

Confusion matrix: A tabulation of the agreement between manual and automatic annotation.

Example:

```
##       unit human computer
## 1    apple  noun     noun
## 2      run  verb     noun
## 3     cake  noun     verb
## 4   coffee  noun     noun
## 5    drink  verb     noun
## 6     swim  verb     verb
## 7     tart  noun     verb
## 8     walk  verb     noun
## 9      ice  noun     noun
## 10     pen  noun     noun
```



Radboud University

Evaluating the *reliability* of automatic tools

Confusion matrix: A tabulation of the agreement between manual and automatic annotation.

Example:

```
##      unit human computer
## 1   apple  noun     noun
## 2     run  verb     noun
## 3    cake  noun     verb
## 4  coffee  noun     noun
## 5   drink  verb     noun
## 6    swim  verb     verb
## 7    tart  noun     verb
## 8    walk  verb     noun
## 9     ice  noun     noun
## 10    pen  noun     noun
```

```
##            Reference
## Prediction noun verb
##       noun    4    2
##       verb    3    1
```

Radboud University

Evaluating the *reliability* of automatic tools

Precision: the extent to which the retrieved objects in a query are correctly tagged

Recall/Sensitivity: the extent to which the objects matching the query retrieve all the target objects in the corpus

F-score/F1-score the balance between precision and recall (what is normally reported)

Radboud University

Evaluating the *reliability* of automatic tools

Precision: the extent to which the retrieved objects in a query are correctly tagged

$$\frac{TP}{(TP + FP)}$$

Recall/Sensitivity: the extent to which the objects matching the query retrieve all the target objects in the corpus

$$\frac{TP}{(TP + FN)}$$

F-score/F1-score the balance between precision and recall (what is normally reported)

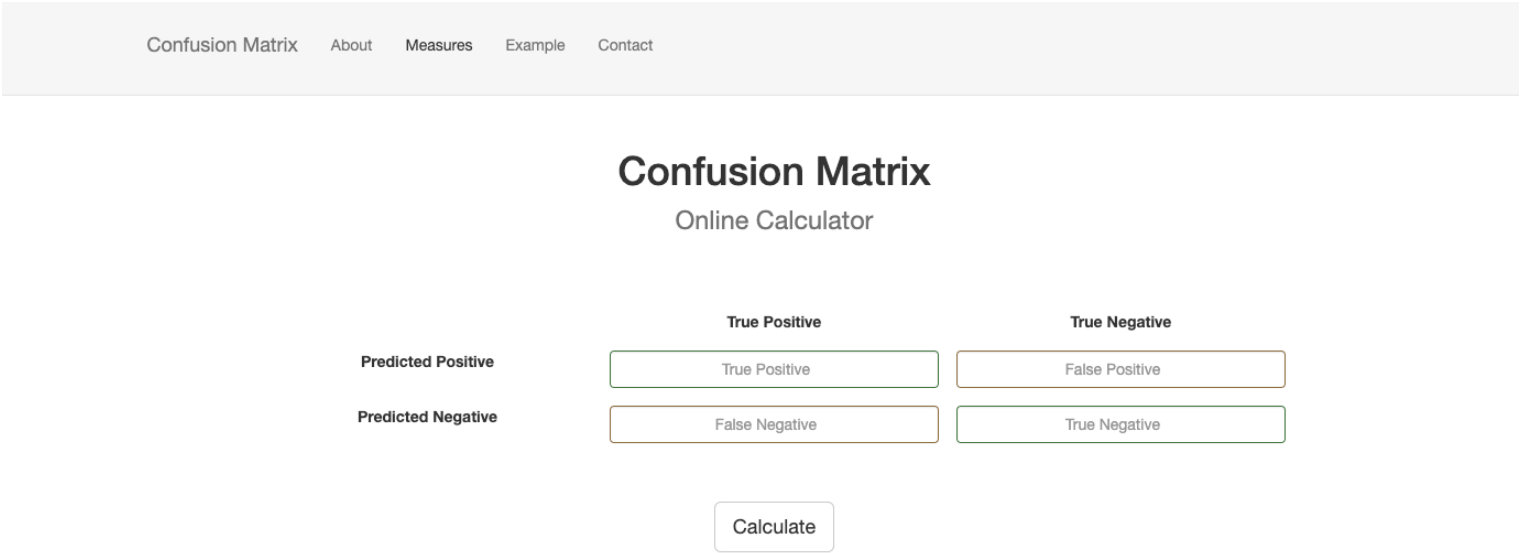
$$\frac{2 * (P * R)}{P + R}$$

Reliability of automatic annotation for fsca tool (Vandeweerd, 2021: 265)

Unit	Precision	Recall	F-score
Sentences	0.96	0.97	0.97
Clauses	0.75	0.72	0.74
Dependent Clauses	0.63	0.58	0.60
Coordinated Clauses	0.54	0.51	0.53
T-units	0.83	0.82	0.83
Noun Phrases	0.84	0.84	0.84
Verb Phrases	0.78	0.78	0.78

Webtools

Confusion Matrix Online Calculator



Radboud University

R

```
library(caret)

confusionMatrix(data, reference)
```

See this vignette for more information about the **caret** package.

```
# Confusion Matrix and Statistics
#
#           Reference
# Prediction  M  R
#          M 21  7
#          R  6 17
#
#                Accuracy : 0.745
#                  95% CI : (0.604, 0.857)
#     No Information Rate : 0.529
#     P-Value [Acc  NIR] : 0.00131
#
#                   Kappa : 0.487
#
#  Mcnemar's Test P-Value : 1.00000
#
#             Sensitivity : 0.778
#             Specificity : 0.708
#          Pos Pred Value : 0.750
#          Neg Pred Value : 0.739
```

Final remarks

Recap

1. What are some types of automatic annotation that can be applied to learner texts?
2. Why is it important to clean and pre-process your texts before using automatic tools?
3. What is POS-tagging?
    - What are some POS tagging tools?
4. What is syntactic parsing?
    - What are some tools for syntactic parsing?
5. How can you evaluate the reliability of automatic annotation tools?

Your research

How can you apply automatic tools to your research?

- Spanish articles
- morpho-syntactic and syntactic labels
- morphology
- semantic, syntactic and discourse features
- verb valency patterns
- grammatical complexity
- grammatical case
- cohesion and cohesive devices
- stance features
- verb aspect
- adverbs of degree and negation
- verb phrase ellipsis
- colour terms
- meta-discourse markers
- formulaic language

Q

Further reading

Kyle, K. (2021). Natural language processing for learner corpus research. International Journal of Learner Corpus Research, 7(1), 1–16.

Murakami, A., Thompson, P., Hunston, S., & Vajn, D. (2017). "What is this corpus about?": Using topic modelling to explore a specialised corpus. Corpora, 12(2), 243–277.

Newman, J., & Cox, C. (2021). Corpus annotation. In A practical handbook of corpus linguistics (pp. 25–48). Springer. https://doi.org/10.4324/9780429269035-7

Schmid, H. (2008). Tokenizing and part-of-speech tagging. In Corpus Linguistics: An International Handbook. de Gruyter.

Shadrova, A. (2021). Topic models do not model topics: epistemological remarks and steps towards best practices. Journal of Data Mining & Digital Humanities, 2021, 7595.

van Rooy, B. (2015). Annotating learner corpora. In S. Granger, G. Gilquin, & F. Meunier (Eds.), The Cambridge handbook of learner corpus research (pp. 79–106). Cambridge University Press. https://doi.org/10.1017/CBO9781139649414.005

Zeldes, A. (2020). Corpus Architecture. In M. Paquot & S. Th. Gries (Eds.), A Practical Handbook of Corpus Linguistics (pp. 49–73). Springer International Publishing. https://link.springer.com/10.1007/978-3-030-46216-1_3