NLP processing from R

Wouter van Atteveldt

CCS Hannover, Feb 2018

Programme

Tuesday:

- AM: Intro, Text analysis with R
- PM: Topic Modeling: application and validation

Wednesday:

- AM: Technical Details
- PM: Structural Topic Modeling

Thursday:

- AM: Linguistic processing

Short AmCAT demo

- `https://amcat.nl`, `https://github.com/amcat/amcat`
- Server-based corpus management tool
  - Project management
  - Permissions (inc. copyright-sensitive)
  - Keyword analysis
  - Quantitative coding
  - API and connects to R, python, nlpipe
  - Open source, docker container available

AmCAT benefits

- Easier overview and sharing of corpora
- Query texts without full text access
- Standardize multi-lingual processing via NLPipe
- Shared starting point for R/Python analyses
- (hit underlying elastic/postgres if you want)

AmCAT and R

- Install AmCAT-r:
  - `install.packages("devtools")`
  - `devtools::install_github("amcat/amcat-r")`
- Connecting to AmCAT:
  - `library(amcat)`
  - `amcat.save.password(...)`
  - `conn = amcat.connect(...)`
- Querying AmCAT:
  - `amcat.aggregate(conn, sets=.., ...)`
  - `amcat.hits(conn, sets=.., ...)`
  - `amcat.hits(conn, project=.., sets=.., col=c("headli`
  - `amcat.getarticlemeta(conn, set=.., ...)`

Why linguistic processing

- Stemming sucks (sorry)
  - Not too badly for English
- Computational Linguistics built some great tools
  - to extract basic structure of text
  - to help filter out uninteresting features
  - to help enrich words
- We can use these to improve our analyses

Steps in linguistic processing

- (cleaning UTF, HTML etc)
- Tokenization
- POS tagging
- Lemmatization
- Entity Recognition
- Dependency parsing
- Coreference resolution

POS tagging

Identify Part-of-speech (POS) of words:

- noun
- name
- verb
- pronoun
- etc.

Great for focussing on certain classes, filtering out function words

Lemmatization

Reduce word to lemma
flew -> fly went -> go
Like stemming, but better^{TM}

Entity recognition

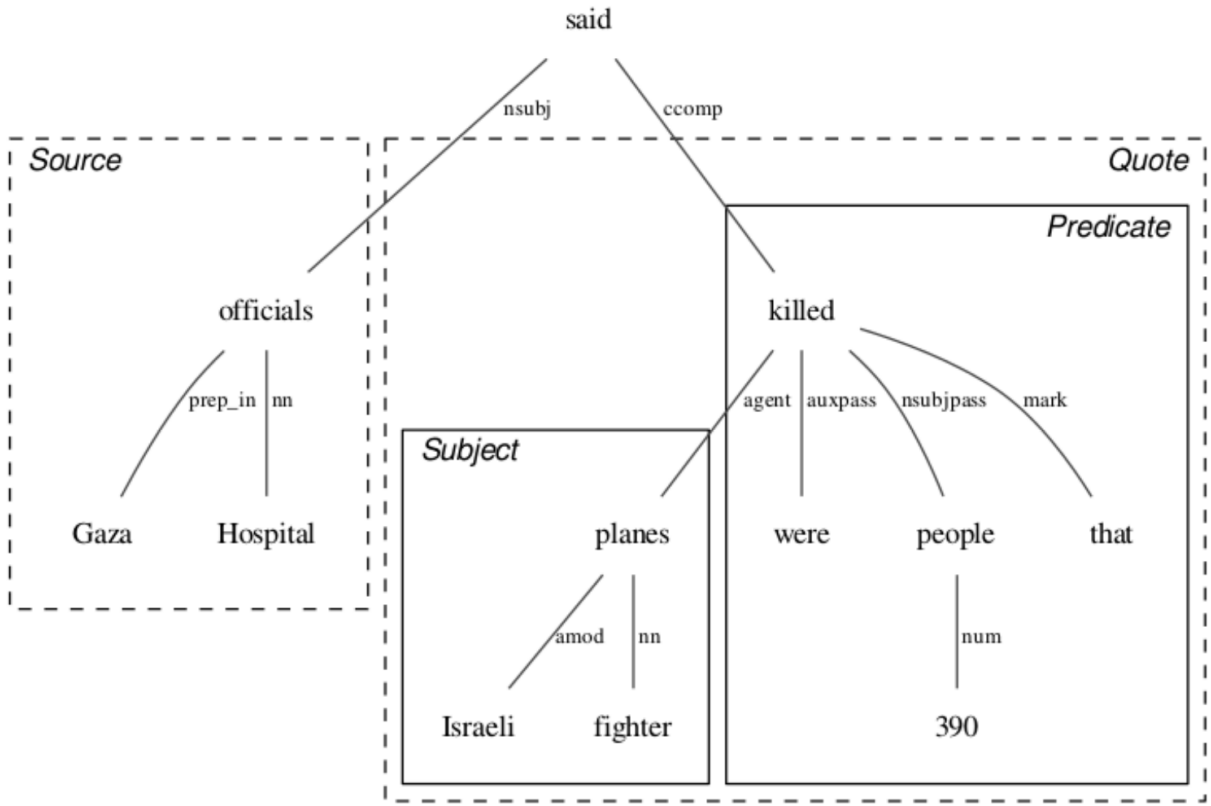
Recognize named entities

- Person
- Organization
- Location
- etc.

Great for building e.g. actor networks, exploring data

Dependency Parsing

- Grammatical structure of sentences



Hospital officials in Gaza said that 390 people were killed by Israeli fighter planes

Coreference Resolution

- (aka anaphora resolution)
- What do 'he', 'she', 'the president' etc refer to?
- Very important in text-level analyses

Running NLP in R

- Is complicated (sorry)

Running NLP in R

- Is complicated (sorry)
- Different NLP groups publish different programs
- Attempts are made to unify them (spacy, coreNLP)
- And R packages exists (coreNLP, spacyr)
- But it can be non-trivial to install

Running NLP in R: your options

- Spacy + spacyr
- coreNLP
- for Dutch: frog + frogr
- nlpiper

Spacy + spacyr

- POS, Lemmatization, parsing for 7 languages
- Install: (`https://spacy.io/usage/`)
  1. Windows: install python (e.g. anaconda)
  2. Windows: install VS express; Mac: install xcode
  3. Using python/pip or conda, install spacy package
  4. Using python, download language model
- Run:

```
library(spacyr)
spacy_initialize("de", executable="/path/to/python")
tokens = spacy_parse('ich bin ein Berliner')
```

coreNLP

```
library(coreNLP)
downloadCoreNLP()
initCoreNLP(type='english')
output = annotateString("I love Hannover")
getToken(output)
```

frog + frogr

1. Install docker
2. Run frog:

```
sudo docker run -dp 9887:9887 proycon/lamachine \      f
```

1. Install and run frogr:

```
devtools::install_github("vanatteveldt/frogr")
library(frogr)
tokens = frogr::call_frog("Tulpen uit Amsterdam",
                          port=9887)
```

nlpipe(r)

```
library(nlpiper)
id = process_async("corenlp_lemmatize",
                   "This is a test")
status(id)
tokens = result("corenlp_lemmatize", id,
                format='csv'
```

Analysing tokens

- Result of NLP processing is tokens dataframe
  - One row per word ('token')
  - Attributes of words in columns (word, lemma, etc)
- You can analyse this using normal dataframe tools
  - esp. `subset`
- Then do actual analysis on filtered tokens

Corpustools

- `corpustools` designed to work with tokens files
  - Remembers word order
- Create a `tcorpus` object from tokens and possible metadata
- Do preprocessing
- Convert to normal dfm/dtm or analyse within corpustools
  - Filter dtm on fragments near a word
  - Windowed associations (semantic networks) and KWIC
  - Dictionary search with proximity / conditions

Corpustools

```
devtools::install_github("kasperwelbers/corpustools")
tc = tokens_to_tcorpus(tokens)
dfm = tc$dtm('lemma', form="quanteda_dfm")
```

See corpustools handout

Conclusion

- R has a lot of tools for text analysis
  - `quanteda`, `topicmodels`, `stm`, `corpustools`
- Important take home lessons
  - Think about what you want to measure
  - 90% of your time is spent gathering and cleaning the data
  - Try out multiple tools as appropriate
  - Validate on the right metric
- From here:
  - Go have fun with these tools!
  - "R for Data Science" (O'Reilly)
  - Check out supervised machine learning, e.g. 'e1071' package (SVM)