Part I: Intro NLP & Task at Hand

Outline

i. Intro NLP
ii. Working Data
iii. Task at Hand
iv. Quanteda Universe

Part I: Intro NLP & Task at Hand

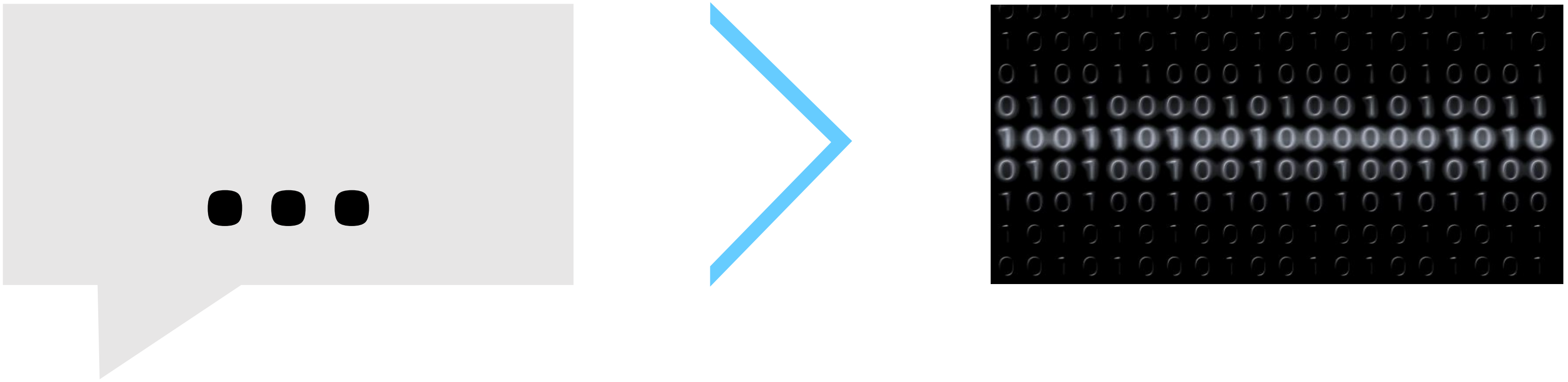
Intro NLP

Intro NLP What is NLP?

> **Natural Language Processing (NLP)** is a theoretically motivated range of *computational techniques* for analyzing and representing *naturally occurring texts* at one or more *levels of linguistic analysis* for the purpose of achieving *human-like language processing* for a *range of tasks or applications* (Liddy, 2001).

Intro NLP Human-like Language Processing

- How to make human language comprehensible to machines?
  - Numerical **vector** representation
  - Characterization by **probabilities**



Intro NLP Naturally Occurring Texts

- Basically, any form of human communication
  - Written text
  - Speech
- Different types in different levels of formality
  - News articles
  - Customer reviews
  - Social media posts
  - ...
- Different languages

Intro NLP Levels of Linguistic Analysis

- **Morphological** – how are words composed?
- **Lexical** – what do single words mean?
- **Syntactic** – what is the grammatical structure of a sentence?
- **Semantic** – what meaning does a sentence convey?
- **Discourse** – how do sentences interact to form a text?
- **Pragmatic** – what is there between the lines?

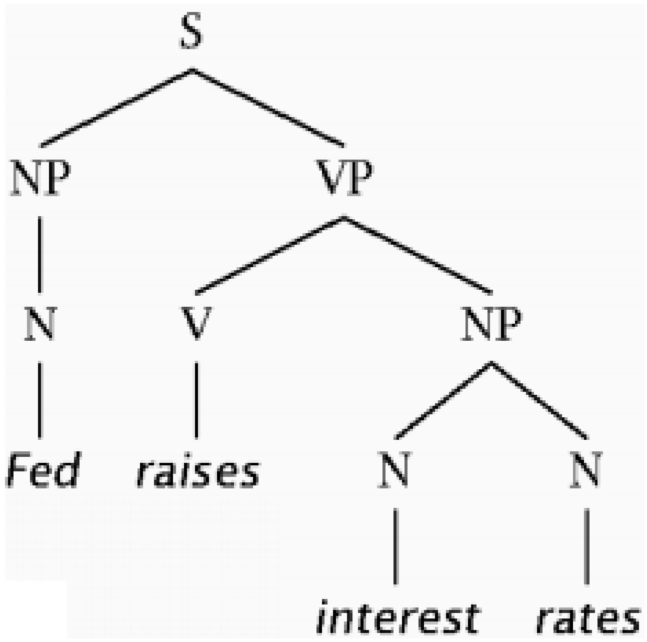
Intro NLP **Tasks**

- High-level tasks
  - Speech recognition
  - Word-sense disambiguation (WSD)
  - Named entity recognition (NER)
  - Relationship extraction
  - Error identification and recovery
  - Automatic summarization
  - Machine translation
  - **Topic extraction**
  - **Sentiment analysis**

+ *many more*

Intro NLP **Tasks**

- Low-level tasks
  - Sentence boundary detection
  - Tokenization
  - Part-of-speech (POS) tagging
  - Stemming
  - Lemmatization
  - Shallow parsing
  - ...



Intro NLP Computational Techniques

- Available techniques largely depending on the task to solve
  - Standard **machine learning** techniques for classification tasks
    → E.g., sentiment analysis
  - **Generative models** for unsupervised tasks
    → E.g., topic modeling
  - **Deep learning** models for various tasks
    → E.g., translation with RNN

- State of the art: **transformer models** (BERT, GPT-3)
  - Idea: teach them as much as possible about the language as a whole (pre-training) and fine-tune to specific tasks

Intro NLP **Challenges**

- Variety of languages
  - Around 7,000 living tongues
  - Many low-resource languages
  - Large differences in grammatical structure, alphabet, scripting systems
- Irregularities
  - Synonyms
  - Homonyms
  - Genera
  - Cases

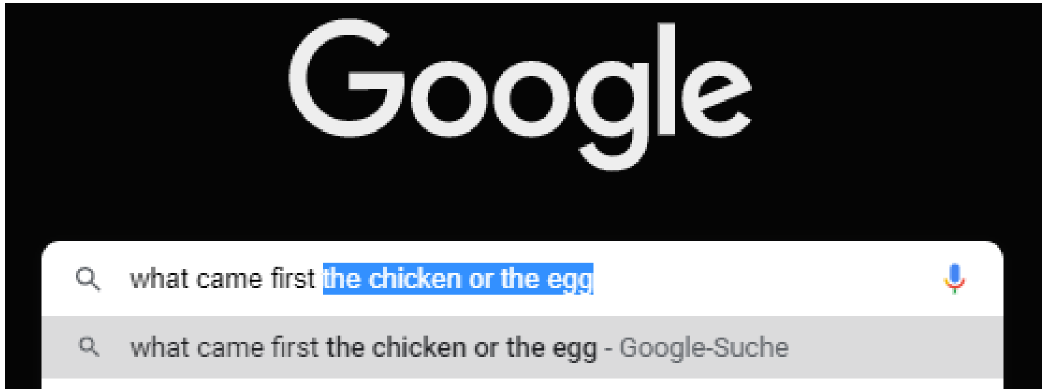
„das Wachstum“ vs „der Reichtum“

Intro NLP **Challenges**

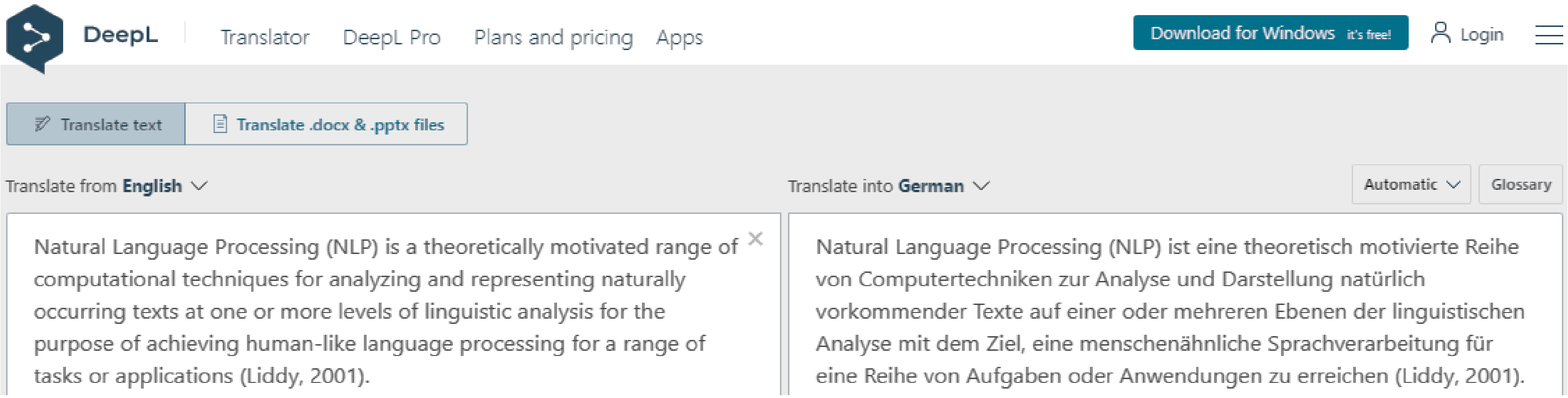
- Contextual dependencies
  - Ambiguities
  - Domain-specific vocabulary
  - Varying formality
- Complex constructs
  - Humor
  - Irony
  - Sarcasm
  - Colloquialisms
- Individual expression
  - Style
  - Emotion
- Errors
  - Transcription/translation errors
  - Misspelling

\+ **Evaluation of NLP tasks**

Intro NLP Applications



ad, unpaid



Part I: Intro NLP & Task at Hand

Working Data

Working Data **Generation**

- All data generated by **scraping** the web

scraping is legal so long as it does not involve breaking security barriers explicitly in place to guard against such automatic data extraction

- Various sources:
  - https://www.bundestag.de/abgeordnete
  - Individual party websites
  - Twitter API

Working Data Structure

- Required information (on MP level)
  - Name
  - Party
  - Electoral district & associated meta data
  - Twitter username
  - Posted tweets
    - Date
    - Text
    - Number of likes, retweets
    - Number of followers

Working Data **Structure**

Variable	Type	Description
`last_name`	`chr`	MP‘s last name
`first_name`	`chr`	MP‘s first name
`wahlkreis_name`	`chr`	MP‘s electoral district
`party`	`factor`	MP‘s political party
`bundesland`	`factor`	Federal state of MP‘s electoral district
`unemployment_rate`	`num`	Unemployment rate in MP‘s electoral district during 2017 election
`share_pop_migration`	`num`	Share of migrant population in MP‘s electoral district during 2017 election
`username`	`chr`	MP‘s username on Twitter
`followers_count`	`num`	MP‘s number of followers on Twitter at scraping time
`created_at`	`date`	Time stamp of tweet creation
`text`	`chr`	Tweet text
`favorite_count`	`num`	Number of likes for tweet at scraping time
`retweet_count`	`num`	Number of retweets for tweet at scraping time

Working Data **Example**

"Merkel-Regierung geht vor Erdogan in die Knie. Auf meine Frage, ob nach Auffassung der Bundesregierung die Ermordung der Armenier 1915/16 ein „Völkermord“ war, eiert sie nur rum. Ihr sei die Position des Bundestages dazu „bekannt“. Sie selbst hat dazu keine. #erbärmlich #feige https://t.co/bkwSflCJan"

Working Data **Particularities**

- Twitter idiosyncrasies
  - Extremely short texts
  - Often in response to recent event without explicitly naming it
  - Informal language with tendency to containing spelling mistakes
  - Special tokens: emojis, hashtags
- Political context
  - Specific vocabulary
  - Sometimes rather formal after all (and few emojis)
  - Many solely informative tweets
  - Tendency toward negative sentiment

\+ *German language*

Part I: Intro NLP & Task at Hand

Task at Hand

Task Analytical Objective

Twitter + socioeconomic data on German MPs

Pre-processing

Topic extraction

Sentiment classification

Sentiment $s \in$ {positive, negative} toward topic $k \in \{1, 2, \ldots, K\}$

Task **Topic Extraction**

... more on this later

- **Topic extraction** aka **topic modeling**: finding latent thematic clusters within a collection of texts
- **Goal**: assign each document a topic probability vector / topic label
- Used for
  - Information retrieval
  - Clustering
  - Supporting upstream tasks

for instance, sentiment analysis

- **Unsupervised task**: both topics and their number unknown

Task Sentiment Analysis

... more on this later

- **Sentiment analysis**: identifying and analyzing affective states
- Relevant subtask: **polarity detection**
- **Goal**: assign each document a polarity label ∈ {positive, negative}
- Used for
  - Customer relationship management
  - Social media analysis

alternative, rule-based approaches exist

- **Supervised task**: requiring labeled training data (typically)

Task Topic-Specific Sentiment Analysis

- **Idea:** domain- / topic-dependence of sentiment predictors

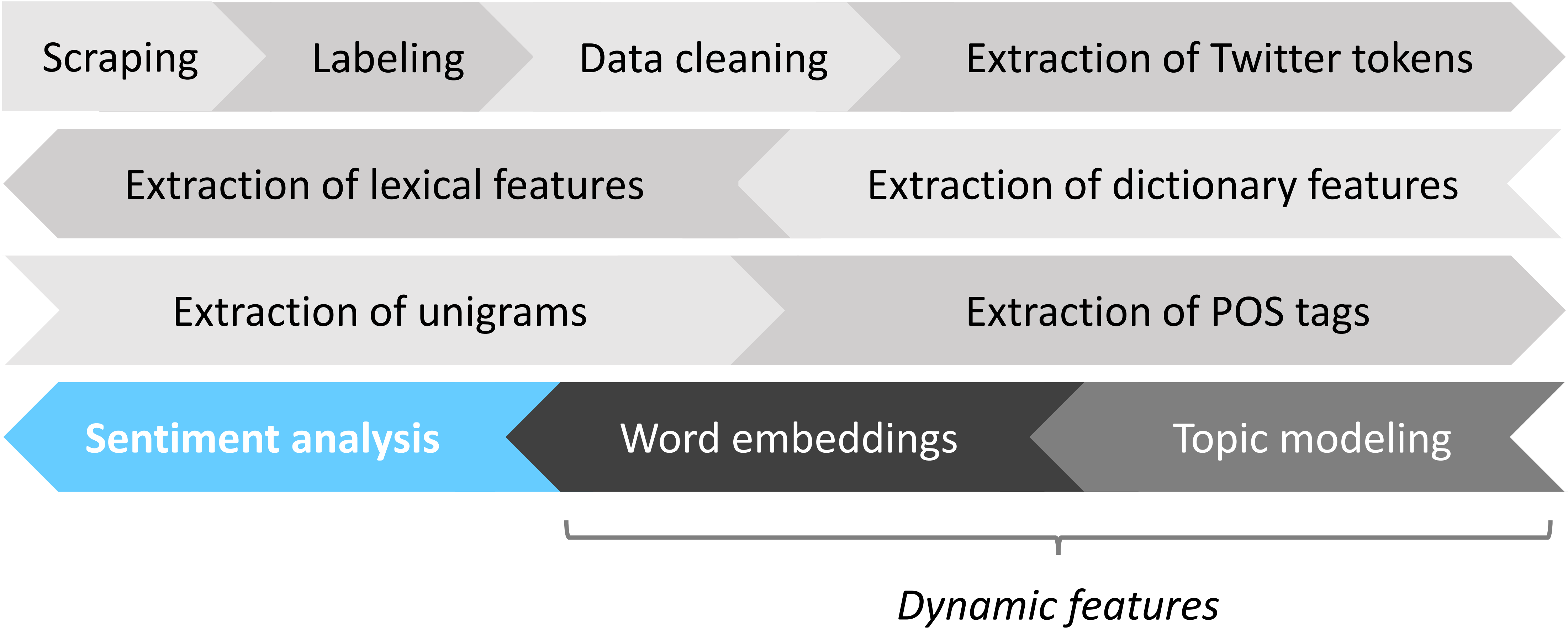
e.g., „Sozialleistungen“ possibly positively connotated in social security context but negatively connotated in asylum politics

→ Combine topic extraction (1) and sentiment analysis (2)

- Implementation
  - **R**: word embeddings per topic
  - **BERT**: aspect-based sentiment analysis

underlying assumption: one aspect per document

ML Pipeline **Analytical Sequence (R)**



ML Pipeline **Static vs Dynamic Features**

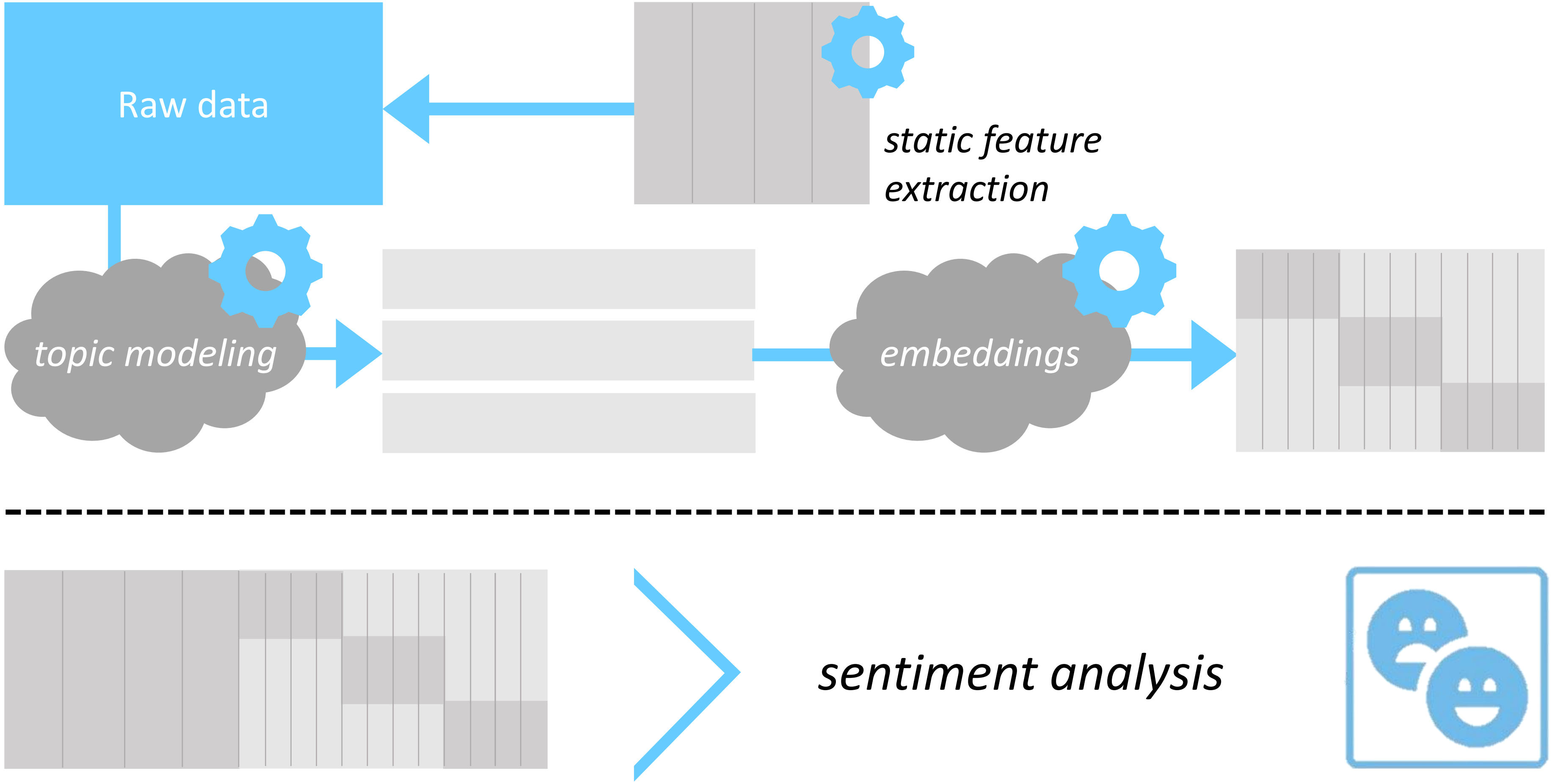
- Fundamental principle in machine learning: dichotomy between **training and test sphere**
  $\rightarrow$ Avoid **bias** in performance estimation

- **Static** features
  - Solely determined on single-observation level
  - E.g., POS tags
- **Dynamic** features
  - Affected by surrounding observations
  - E.g., topic labels

may be computed before training

must be computed during training

ML Pipeline **Static vs Dynamic Features**



Part I: Intro NLP & Task at Hand

Quanteda Universe

Quanteda Universe **Package**

- Benoit et al. (2018)
- Convenient text handling in R
  - Designated **classes** for textual data (with easy conversion to and from `data.frame` & friends)
  - **User-friendly** syntax
  - **Fast** computation
  - Compatibility with `spacyr` package (Benoit et al., 2020)
    → Wrapper for Python's popular `spaCy` package used for, i.a., **POS tagging**

tutorials for getting started on https://tutorials.quanteda.io/

Quanteda Universe **Basic Classes**

[Word = smallest entity of text $\rightarrow$ **words**]
[Sentence = sequence of w words $\rightarrow$ **sentences**]
[Paragraph = sequence of s sentences $\rightarrow$ **not relevant**]
[Document = sequence of p paragraphs $\rightarrow$ **tweets**]

- `corpus`
  - Most basic class to handle text data
  - Collection of documents + document-level variables $\rightarrow$ **tweets + meta data**

lower-level corpora, e.g., as collections of paragraphs, also possible

Quanteda Universe **Basic Classes**

- `tokens`
  - Representing documents as a collection of tokens
    → **tokens per tweet + meta data**
  - **Token**: sequence of characters grouped together as a useful semantic unit
    → Single words, n-grams, ...
  - During tokenization, we will often
    - Remove punctuation
    - Remove stopwords
    - Omit cases (e.g., lowercase everything)
    - Perform stemming / lemmatization

    text normalization – to be continued
  - **Goal**: representation of texts by tokens that co-occur across documents

Quanteda Universe **Basic Classes**

```
 doc_id  text                                                              author                 nationality
-------  ----------------------------------------------------------------  ---------------------  -----------
      1  Politics have no relation to morals.                               Niccolo Machiavelli    Italian
      2  Politics is too serious a matter to be left to the politicians.   Charles de Gaulle      French
      3  In politics stupidity is not a handicap.                           Napoleon Bonaparte     French
```

```
Corpus consisting of 3 documents and 2 docvars.
1 :
"Politics have no relation to morals."

2 :
"Politics is too serious a matter to be left to the politicia..."

3 :
"In politics stupidity is not a handicap."
```

```
Tokens consisting of 3 documents and 2 docvars.
1 :
[1] "Politics" "relation" "morals"

2 :
[1] "Politics"    "serious"     "matter"      "left"        "politicians"

3 :
[1] "politics"   "stupidity" "handicap"
```

Quanteda Universe **Basic Classes**

- `dfm`
  - **Document-feature matrix**
  - Token count per document → **word occurrence per tweet + meta data**
  - Methods
    - **Weighting** schemes, such as tf-idf
    - Counting **matches** with a list of words
    - Extracting **top** features
    - Performing dictionary **look-ups**

```
Document-feature matrix of: 3 documents, 9 features (59.3% sparse) and 2 docvars.
    features
docs politics relation morals serious matter left politicians stupidity handicap
   1        1        1      1       0      0    0           0         0        0
   2        1        0      0       1      1    1           1         0        0
   3        1        0      0       0      0    0           0         1        1
```

Quanteda Universe **Basic Classes**

- `fcm`
  - **Feature co-occurrence matrix**
  - Tokens co-occurrence count across corpus → **co-occurrence across tweets**

```
Feature co-occurrence matrix of: 9 by 9 features.
              features
features       politics relation morals serious matter left politicians stupidity handicap
  politics            0        1      1       1      1    1           1         1        1
  relation            0        0      1       0      0    0           0         0        0
  morals              0        0      0       0      0    0           0         0        0
  serious             0        0      0       0      1    1           1         0        0
  matter              0        0      0       0      0    1           1         0        0
  left                0        0      0       0      0    0           1         0        0
  politicians         0        0      0       0      0    0           0         0        0
  stupidity           0        0      0       0      0    0           0         0        1
  handicap            0        0      0       0      0    0           0         0        0
```

Quanteda Universe **Basic Classes**

- `dictionary`
  - Essentially, named list
  - Specifying dimensions with associated items
  - Look-up on document level → **dictionary item count per tweet**

```
Dictionary object with 2 key entries.
- [political]:
  - politics, politicians
- [critical]:
  - morals, stupidity, handicap
```

```
Document-feature matrix of: 3 documents, 2 features (16.7% sparse) and 2 docvars.
    features
docs political critical
   1         1        1
   2         2        0
   3         1        2
```

Quanteda Universe **Scope**

- Purpose of `quanteda`: handling text corpora and performing basic analysis of their components

- **Within scope**
  - Organizing text documents
  - Tokenization
  - Descriptive analyses

  pre-processing with `quanteda`

- **Out of scope**
  - Higher-level text analysis, such as topic modeling or sentiment analysis

  downstream analyses with other tools

Part I: Intro NLP & Task at Hand

Literature and References

Eisenstein, J. (2019): Introduction to Natural Language Processing, MIT Press.

Liddy, E.D. (2001): Natural Language Processing, *in*: Encyclopedia of Library and Information Science, 2nd ed., NY. Marcel Decker, Inc.

Nadkarni, P. M., Ohno-Machado, L., and Chapman W. (2011): Natural Language Processing: An Introduction. *Journal of the American Medical Informatics Association 18(5)*, 544–551, https://doi.org/10.1136/amiajnl-2011-000464.

Vayansky, I., and Kumar S.A.P. (2020): A Review of Ttopic Modeling Methods. *Information Systems*, doi: https://doi.org/10.1016/j.is.2020.101582.

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., and Matsuo, A. (2018): quanteda: An R package for the Quantitative Analysis of Textual Data. *Journal of Open Source Software 3(30)*, 774, https://doi.org/10.21105/joss.00774.