UNIVERSITÄT
LEIPZIG

Forschungseminar CSS – Text as Data I

GWZ H2 1.15, 13.11.2025

Felix Lennert, M.Sc.

OUTLINE

- A collection of thoughts on what it means to measure things with text
- Measuring things with text and the Bag of Words
  - Preprocessing
  - Sentiment Analysis
  - TF-IDF
  - POS, NER, Dependency Parsing

DISTANT READING

“The extraction of implicit, previously unknown and potentially useful information from large amounts of textual resources.” (Bizer 2019: 4)

- Text analysis methods distill generalizations from language
  ⟹ new data is produced
- (Potential) end goals:
  - Numeric representation of your text (e.g., labels)
  - Extract and count terms you are interested in

STOLTZ & TAYLOR 2024: TEXT MAPPING

- Identification of patterns in text (theory-driven)
- Map texts systematically according to these patterns
  - Which topic are they dealing with
  - What narratives can be found in there
  - What’s their tone
- Later, connect these patterns to context variables
  - Who wrote the text
  - When was it written
  - What are the consequences?

A NEW THING?

1910: Max Weber's "Universal Press Project" – **systematic analysis of the media and the values the texts contain**

1934: Lasswell produces first "keyword count" – "exact" **quantitative science** as opposed to qualitative "impressionism"

~1950: Turing foresees developments in AI

1950s: Gottschalk connects psychoanalysis with content analysis – **quantitative, systematic coding of patients' responses**

1952: first book about **content analysis** (Berelson 1952)

1954: "Georgetown-IBM Experiment" – automated **text translation**

1963: Mosteller and Wallace (1963) analyze federalist papers – harness a **Bayesian approach using "marker words"** to

determine authorship

1966: General Inquirer (Stone, Bales, Namenwirth, and Ogilvie 1962) – **combination of dictionaries**

1981: Weintraub counts **"parts of speech"** (Weintraub 1981)

1986: Pennebaker develops LIWC (Linguistic Inquiry and Word Count) (Tausczik and Pennebaker 2010)

2003: Blei, Ng, and Jordan (2003) develop LDA – **unsupervised topic modeling**

2010: Hopkins and King (2010) bring **supervised ML** into the "social science mainstream" (ReadMe)

2013: word2vec (Mikolov et al. 2013) – **distributive hypothesis**

2017: "attention is all you need" (Vaswani et al. 2017) – new way of processing text
2022: ChatGPT launches for public

GRIMMER, ROBERTS, AND STEWART (2022)

Six Principles:

- Theory still matters for research design
- Text analysis augments humans
- Text analysis methods distill generalizations from language
- Choose the method based on the task
- Validation is essential and theory- and task-dependent
- Building, refining, and testing social science theories requires iteration and cumulation

THEORY MATTERS

when designing your research, ask yourself the following questions:

- what data are relevant?
- how do I measure the concept? (see also principle #5!)
- which results do I expect?
- how do they matter?

⟹ **theory-dependent**

TEXT ANALYSIS AUGMENTS HUMANS

TEXT ANALYSIS AUGMENTS HUMANS

humans are still decisive part of the research process:

- supervised methods: they need to “instruct” the computer, validate the results
- unsupervised methods: they need to make sense of the outcome

⇒ computers offer a “different way of reading”

⇒ both the “instruction” in supervised ML and the “sense making” in unsupervised methods is **qualitative work**

- “For example, manually coding topics from 40 million scientific abstracts could take a thousand researcher-years, but automatic coding by a trained model might require only a few computer-days.” (Evans & Aceves 2016: 5)

TEXT ANALYSIS METHODS DISTILL GENERALIZATIONS FROM LANGUAGE

“all models are wrong – but some are useful”

text is high-dimensional – even beyond words

⇒ we need to reduce dimensionality in order to get…

- interpretability – e.g., use topic models to reduce the number of documents to use/read
- analyzability – remove uninformative noise (i.e., words), e.g., for predictions using text – overfitting!
- back to theory – usually low-dimensional, e.g., left-right scale of parties

TEXT ANALYSIS METHODS DISTILL GENERALIZATIONS FROM LANGUAGE

"all models are wrong – but some are useful"

How does it look in practice?

- supervised methods: classifying documents into distinct categories (positive/negative, containing concept A/B/C/D…), giving documents a value on a continuous scale (e.g., ideology) based on similarity to pre-selected texts, etc.
- unsupervised methods: organizing documents into groups based on their content

BEST METHOD DEPENDS ON THE TASK

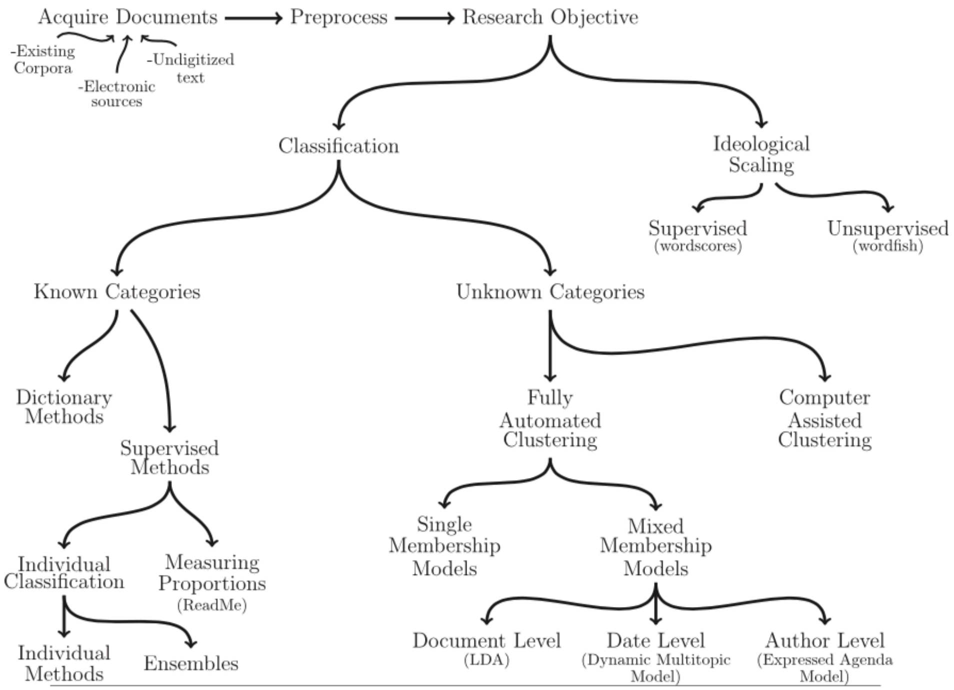


Fig. 1 An overview of text as data methods.

no silver bullets

BEST METHOD DEPENDS ON THE TASK

no silver bullets

examples:

- topic detection in newspaper articles – topic model, e.g., LDA
- sentiment classification – dictionary based, classifiers
- measurement of ideology – supervised (wordscores), unsupervised (wordfish), semi-supervised (LSS)
- All these things have also been shown to be achieved using LLMs…

⟹ depends on data characteristics (topic detection in tweets vs. newspapers), goal/ task, and performance and validity of analysis

VALIDATE VALIDATE VALIDATE

humans need to make sure that they measure what they want to measure
⇒ for the first step, this usually requires reading a set of documents and then checking the results

- supervised methods: annotating a full set and subsequently split into training vs. held out test set
- unsupervised methods: check the documents in the respective clusters, read them – does the classification “make sense”?; also: measures of model fit

VALIDATE VALIDATE VALIDATE

humans need to make sure that they measure what they want to measure
⇒ next step: how are measures aggregated across documents? – is there systematic bias?

example: spam filter

- goal is to send few important mails to spam folder (avoid false positives)
- therefore, the classifier might become less sensitive – higher threshold to send email to spam folder to not upset the user
- number of spam emails might be underestimated – trade-off

BUILDING, REFINING, AND TESTING SOCIAL SCIENCE THEORIES REQUIRES ITERATION AND CUMULATION

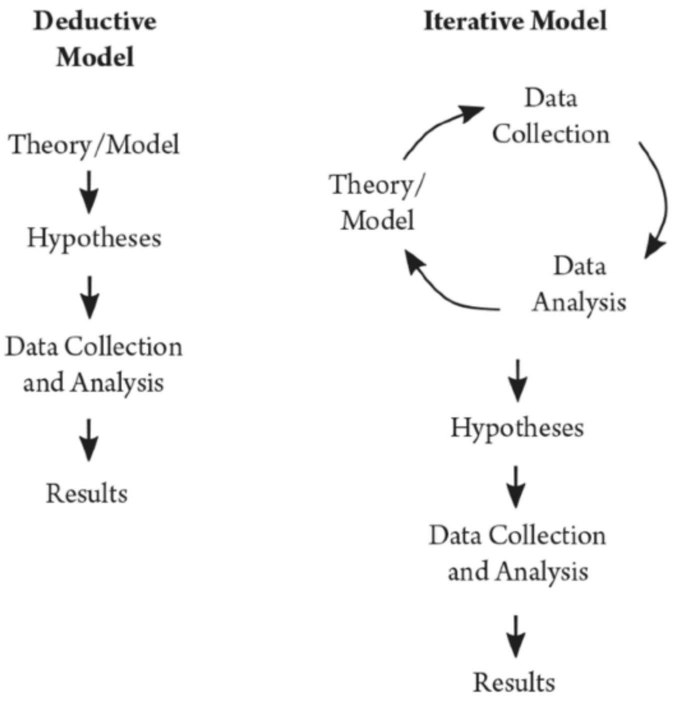


Figure 2.1. Flowcharts for the standard deductive model of research (left) as compared to the iterative model of research (right).

corpus

Bundeswehr

Die fliegenden Spione

18. April 2024, 17:14 Uhr | Lesezeit: 4 min | 8 Kommentare

Von Georg Ismar und Paul-Anton Krüger, Berlin

author

feature/token/word

Die Bundeswehr weiß nicht erst seit dem Lauschangriff auf ein Gespräch hochrangiger Offiziere, dass sie im Fokus russischer Operationen steht. Vor allem im Bereich Drohnen ist der Aufholbedarf so groß, dass man kaum weiß, wo man anfangen soll - und das betrifft neben dem militärischen Einsatz auch die Abwehr von Spionage.

document

SZ Plus Augenheilkunde

Was tun gegen Kurzsichtigkeit?

Je schlechter jemand sieht, desto höher ist das Risiko für schwere Augenschäden bis zur Erblindung. Worauf Betroffene achten sollten.

Von Celine Chorus

SZ Plus Bezahlen

Echt oder unecht? Wenn die Kreditkarte im Ausland zum Problem wird

Wer mit seiner kostenfreien Debitkarte in die Ferne reist, stößt schnell an Grenzen. Mietwagen abholen oder ins Hotel einchecken kann zum Problem und am Ende teuer werden. Meist hilft eine "echte" Zweitkarte.

Von Berrit Gräber

Kryptowährung

Wie das Halving bei Bitcoin funktioniert

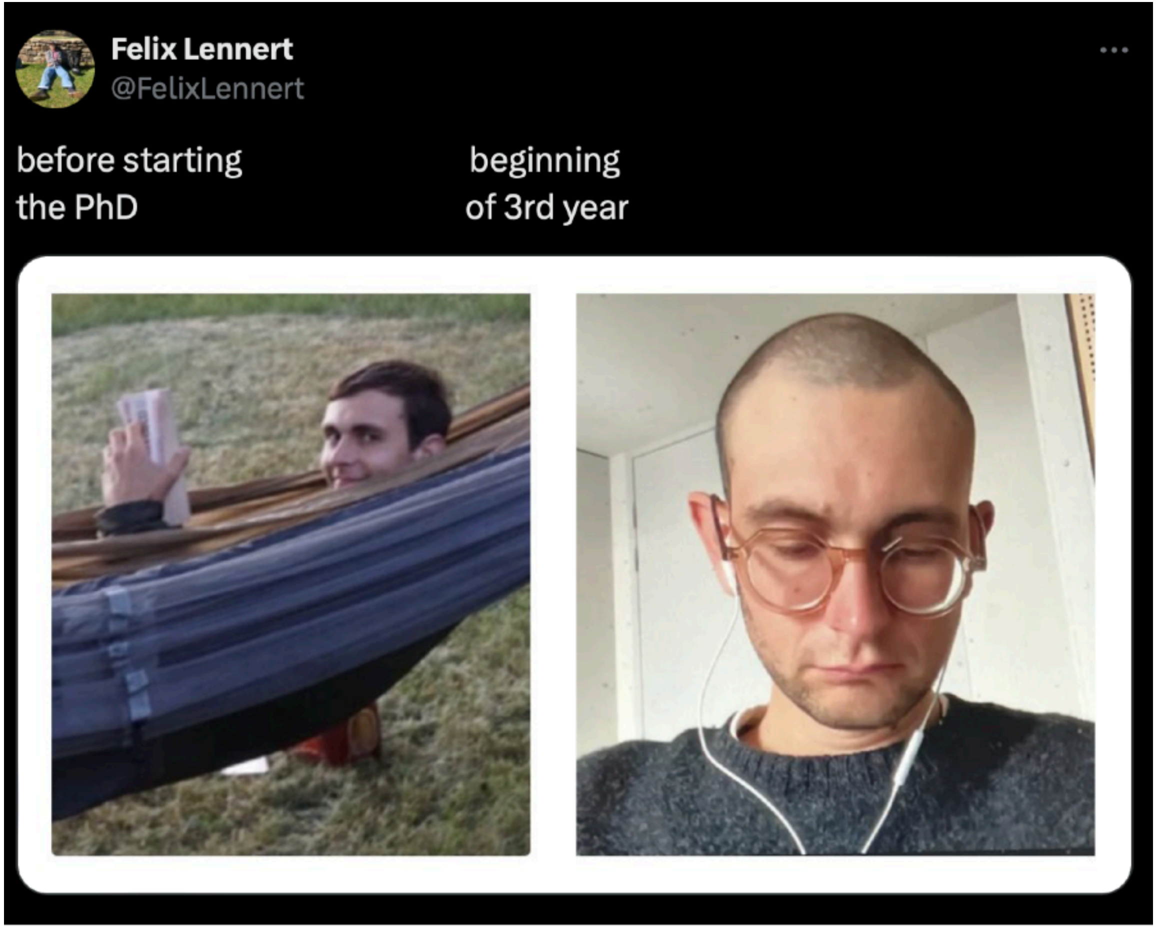
Am Samstagmorgen halbiert sich der Nachschub an neuen Bitcoin, die in großen Rechnerfarmen geschürft werden. Das Event dürfte einige Auswirkungen haben - auch auf den Preis der Währung.

Von Max Muth

What is:

- author
- document
- feature/token/word

What could a corpus look like?

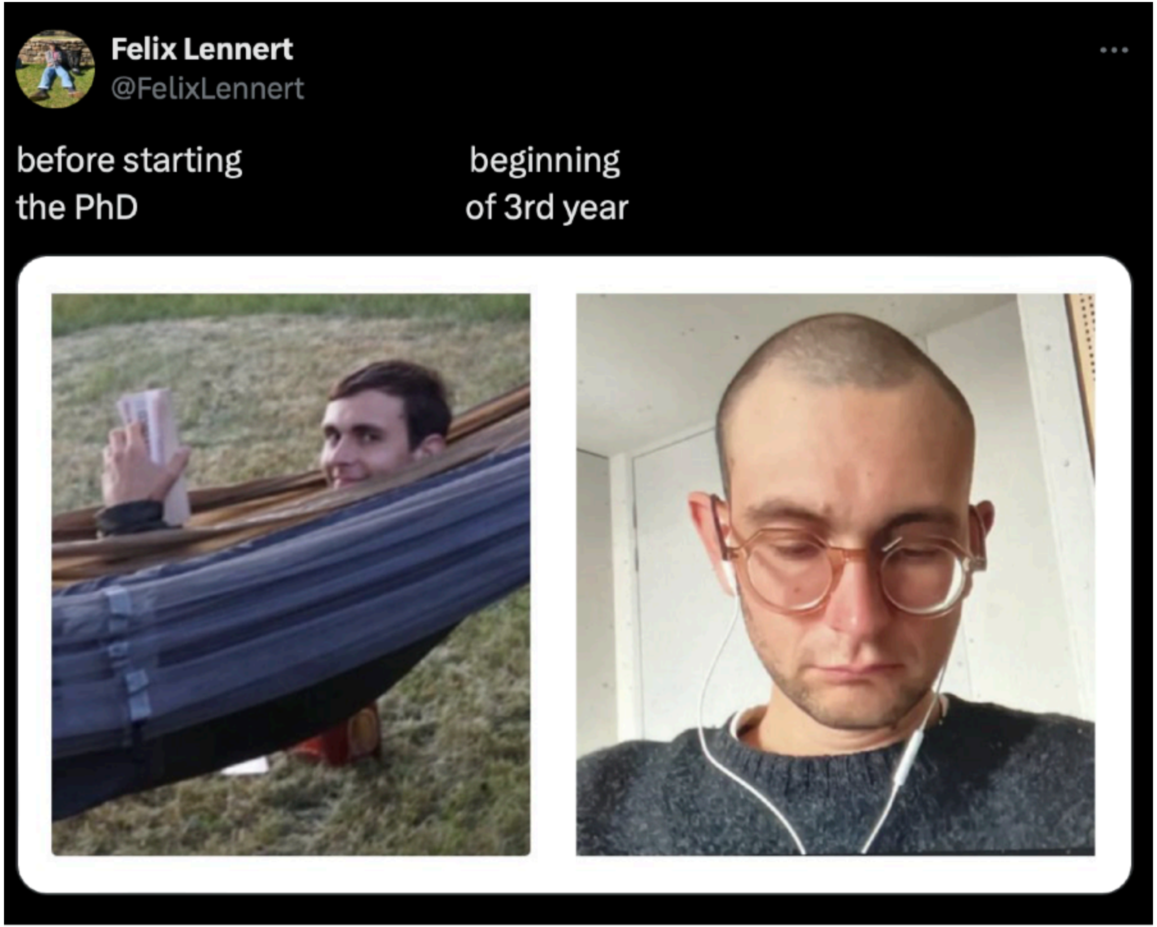


What is:

- author – ME
- document – the tweet
- feature/token/word – the text; perhaps a description of the picture; split up into words

What could a corpus look like?

- some sample of tweets (e.g., timeline)



HOW TO REPRESENT TEXT

How does a computer see text?

- Collections of characters (letters, numbers, special characters, etc.) – STRING
- Possible operations: comparisons (“string A is the same as string B”)

Our goal:

- We want to perform math on this text
- We need to transform text to numbers

HOW TO REPRESENT TEXT

One way to introduce numbers:

- Count features/tokens/words (“featurization”)
- Represent each document as the counts of its *unique* words
- “Bag of Words”

NO RIGHT WAY TO REPRESENT TEXT

From Wikipedia:
“The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity.”

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

REPRESENTATION – DTM

```r
dtm <- movie_review |>
  enframe(name = "sentence", value = "text") |>
  unnest_tokens("words", "text") |>
  count(sentence, words) |>
  cast_dtm(doc, words, n)
```

```
> as.matrix(dtm)[, 1:10]
    Terms
Docs i love movie this but humor it's satirical sweet with
   1 1    1     1    1   0     0    0         0     0    0
   2 0    0     0    0   1     1    1         1     1    1
   3 0    0     0    0   0     0    0         0     0    0
   4 0    0     0    0   0     0    0         0     0    0
   5 1    0     0    0   0     0    0         0     0    0
   6 1    0     0    0   0     0    0         0     0    0
```

TEXT TO DATA

- Each document is represented by its words – 6 points (here: sentences) in a 54 dim (the number of different words) space
- Problem: dimensionality
  ⇒ the dimensionality grows a the more documents you introduce
  - A lot of the words is just noise

- The next slides will introduce you how to remove complexity
- We will get rid of:
  - 🫡 Word order ("bag of words")
  - 🫡 Uppercase characters
  - 🫡 Special characters
  - Inflections ("lemmatization", "stemming")
  - Too frequent words ("stopwords")
  - Infrequent words

TEXT TO DATA

- Stemming and lemmatization
- Goal: bring the words into their basic forms – stem or lemma (– basic form)
- stemming is rule-based and “stupid” – but fast and efficient
- lemmatization is more sophisticated and model-based, hence reliable – but slow

```
> tictoc::tic()
> wordStem(rep(special_cases, 10000)) |> head()
[1] "studi" "buri"  "studi" "buri"  "studi" "buri"
> tictoc::toc()
0.013 sec elapsed
```

```
> tictoc::tic()
> spacy_parse(rep(special_cases, 10000)) |>
+   pull(lemma) |>
+   head()
[1] "study" "bury"  "study" "bury"  "study" "bury"
> tictoc::toc()
12.911 sec elapsed
```

PREPROCESSING – STEMMING

	studies	buried	study	buries	studied
doc 1	1	2	0	1	2
doc 2	1	0	0	3	0
doc 3	2	1	3	0	1
doc 4	0	0	2	0	1

```
> tictoc::tic()
> wordStem(rep(special_cases, 10000)) |> head()
[1] "studi" "buri"  "studi" "buri"  "studi" "buri"
> tictoc::toc()
0.013 sec elapsed
```

```
> tictoc::tic()
> spacy_parse(rep(special_cases, 10000)) |>
+   pull(lemma) |>
+   head()
[1] "study" "bury"  "study" "bury"  "study" "bury"
> tictoc::toc()
12.911 sec elapsed
```

	studies	buried	study	buries	studied
doc 1	1	2	0	1	2
doc 2	1	0	0	3	0
doc 3	2	1	3	0	1
doc 4	0	0	2	0	1

	study/ studi	bury/ buri
doc 1	3	3
doc 2	1	3
doc 3	6	1
doc 4	3	0

TEXT TO DATA

- Stemming and lemmatization
- Goal: bring the words into their basic forms

```
> dtm_stemmed |> dim()
[1]  6 53
> as.matrix(dtm_stemmed)[, 1:10]
    Terms
Docs i love movi thi but humor it' satir sweet with
   1 1    1    1   1   0     0   0     0     0    0
   2 0    0    0   0   1     1   1     1     1    1
   3 1    0    0   0   0     0   0     0     0    0
   4 0    0    0   0   0     0   0     0     0    0
   5 1    0    0   0   0     0   0     0     0    0
   6 1    0    0   0   0     0   0     0     0    0
```

TEXT TO DATA

🫡 – Word order (“bag of words”)
🫡 – Uppercase characters
🫡 – Special characters
🫡 – Inflections (“lemmatization”, “stemming”)
– Too frequent words (“stopwords”)
– Infrequent words

TEXT TO DATA

- One of the oldest mysteries in linguistics: Zipf’s law – the most common term appears (roughly) twice as often as the second-most common term which appears twice as often as the third-most, etc.

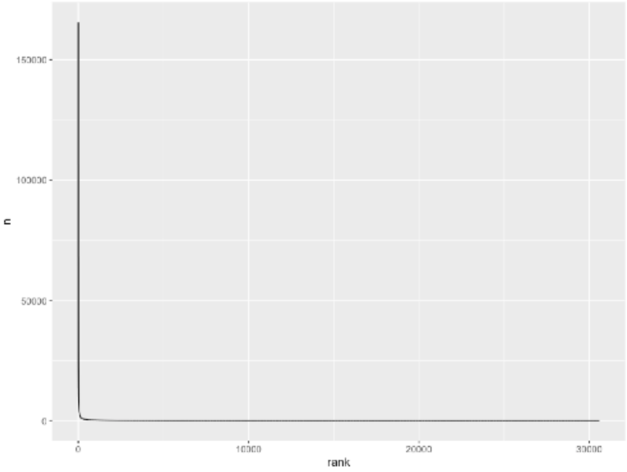
```
> zipf_example <- sotu_text |>
+   enframe(name = NULL, value = "text") |>
+   unnest_tokens(token, text) |>
+   count(token) |>
+   arrange(-n) |>
+   rowid_to_column("rank")
```

```
> zipf_example
# A tibble: 30,585 × 3
    rank token       n
   <int> <chr>   <int>
 1     1 the    165601
 2     2 of     106402
 3     3 and     68063
 4     4 to      68037
 5     5 in      43429
 6     6 a       31342
 7     7 that    24113
 8     8 for     21701
 9     9 be      20449
10    10 our     19598
# … with 30,575 more rows
```

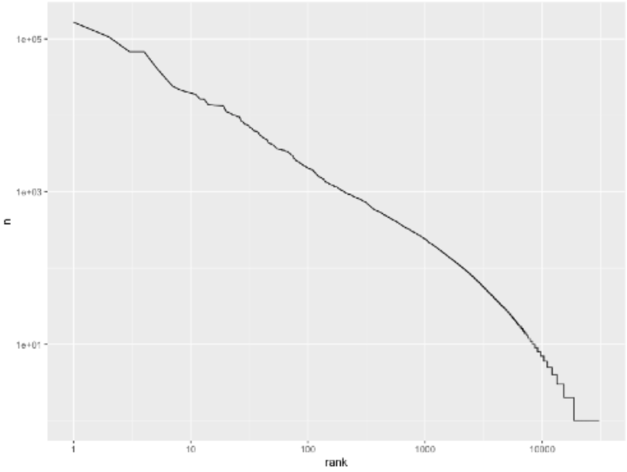
TEXT TO DATA

- One of the oldest mysteries in linguistics: Zipf's law – the most common term appears (roughly) twice as often as the second-most common term which appears twice as often as the third-most, etc.

```
> ggplot(zipf_example) +
+   geom_line(aes(x = rank, y = n))
```




```
> ggplot(zipf_example) +
+   geom_line(aes(x = rank, y = n)) +
+   scale_x_log10() +
+   scale_y_log10()
```



TEXT TO DATA

- Reason: mix of syntax and semantics (Lestrade 2017)
- What this also implies: a bunch of words occur in almost every document – they bear no particular meaning, and can hence be safely removed

  ⟹ "Stopwords"

- BUT BEWARE: they might carry meaning (e.g., gender)

```
> stopwords::stopwords() |> head(21)
 [1] "i"          "me"         "my"         "myself"     "we"         "our"        "ours"
 [8] "ourselves"  "you"        "your"       "yours"      "yourself"   "yourselves" "he"
[15] "him"        "his"        "himself"    "she"        "her"        "hers"       "herself"
```

TEXT TO DATA

```
> as.matrix(dtm)[, 1:10]
    Terms
Docs i love movie this but humor it's satirical sweet with
   1 1    1     1    1   0     0    0         0     0    0
   2 0    0     0    0   1     1    1         1     1    1
   3 0    0     0    0   0     0    0         0     0    0
   4 0    0     0    0   0     0    0         0     0    0
   5 1    0     0    0   0     0    0         0     0    0
   6 1    0     0    0   0     0    0         0     0    0
```

```
> dtm_stemmed_nostop |> dim()
[1]  6 21
> as.matrix(dtm_stemmed_nostop)[, 1:10]
    Terms
Docs love movi humor satir sweet adventur dialogu fun scene convent
   1    1    1     0     0     0        0       0   0     0       0
   2    0    0     1     1     1        0       0   0     0       0
   3    0    0     0     0     0        1       1   1     1       0
   4    0    0     0     0     0        0       0   0     0       1
   5    0    0     0     0     0        0       0   0     0       0
   6    0    0     0     0     0        0       0   0     0       0
```

⇨ BEFORE: 6 54

TEXT TO DATA

🫡 – Word order (“bag of words”)
🫡 – Uppercase characters
🫡 – Special characters
🫡 – Inflections (“lemmatization”, “stemming”)
🫡 – Too frequent words (“stopwords”)
– Infrequent words

TEXT TO DATA

- Vice versa: there are incredibly many infrequent words
- These may also not bear any particular meaning/value but induce plenty of noise
- Hence, you may consider removing them, too

⟹ Not in our example

TEXT TO DATA

- Same holds for special characters

- However, some may bear value:
  - Identify questions [?]
  - Identify sentences/paragraphs [.\n!?:]
  - Identify sentiment (:’-) -.- ;-) ¯_(ツ)_/¯)

  - etc.

⟹ THE BEST PREPROCESSING DEPENDS ON THE CASE

FINALLY: WHAT CAN WE DO WITH THE BOW/DTM?

(1) Use columns as inputs for different algorithms
⇾ e.g., each word (count) constitutes a variable to predict an outcome

(2) Use linear algebra to determine similarity of documents and words
⇾ *documents*: embedded in space based on word overlap – the more words they share, the closer
⇾ *words*: embedded in space based on document overlap – the more they appear in same documents, the closer // alternatively: the other words they co-appear with (context-cooccurrence matrix – CCM; *wait for embeddings session*)

(3) use it as input for networks
⇾ documents connected based on word overlap

SO WHAT NOW?

- We have a mathematical representation of a document
- But, remember, we need something even more low-dimensional
  - A numeric value, e.g., indicating sentiment (positive, negative)
  - “Special” terms:
    - Words that describe it well ⇨ distinct terms
    - Words that matter for us ⇨ named entities
    - Words that take a particular role in the text ⇨ Parts-of-Speech, Dependency-parsing

DICTIONARY-BASED ANALYSIS

- A numeric value/label, e.g., indicating sentiment (positive, negative)
- Most basic approach: pre-define terms that stand for the sentiment

  ⟹ Positive or negative terms

SENTIMENT

- Example: which terms say something about whether the person liked or disliked the movie?

I love this movie! It’s sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun… It manages to be whimsical and romantic, while laughing at the conventions of the fairytale genre. I would recommend it to just about anyone. I’ve seen it several times, and I’m always happy to see it again whenever I have a friend who hasn’t seen it yet!

SENTIMENT ANALYSIS

Idea: sentiment of document can be measured by counting positive and negative terms

I **love** this movie! It's **sweet**, but with satirical **humor**. The dialogue is **great** and the adventure scenes are **fun**… It manages to be whimsical and **romantic**, while laughing at the conventions of the fairytale genre. I would **recommend** it to just about anyone. I've seen it several times, and I'm always **happy** to see it again whenever I have a friend who hasn't seen it yet!

$$t_i = \sum_{m=1}^{M} \frac{s_m W_{im}}{N_i}$$

				t_i
	i	am	happy	
s	0	0	1	0.33
	i	am	sad	
s	0	0	-1	-0.33

t_i = tone of document i

m = term

s_m = Sentiment value

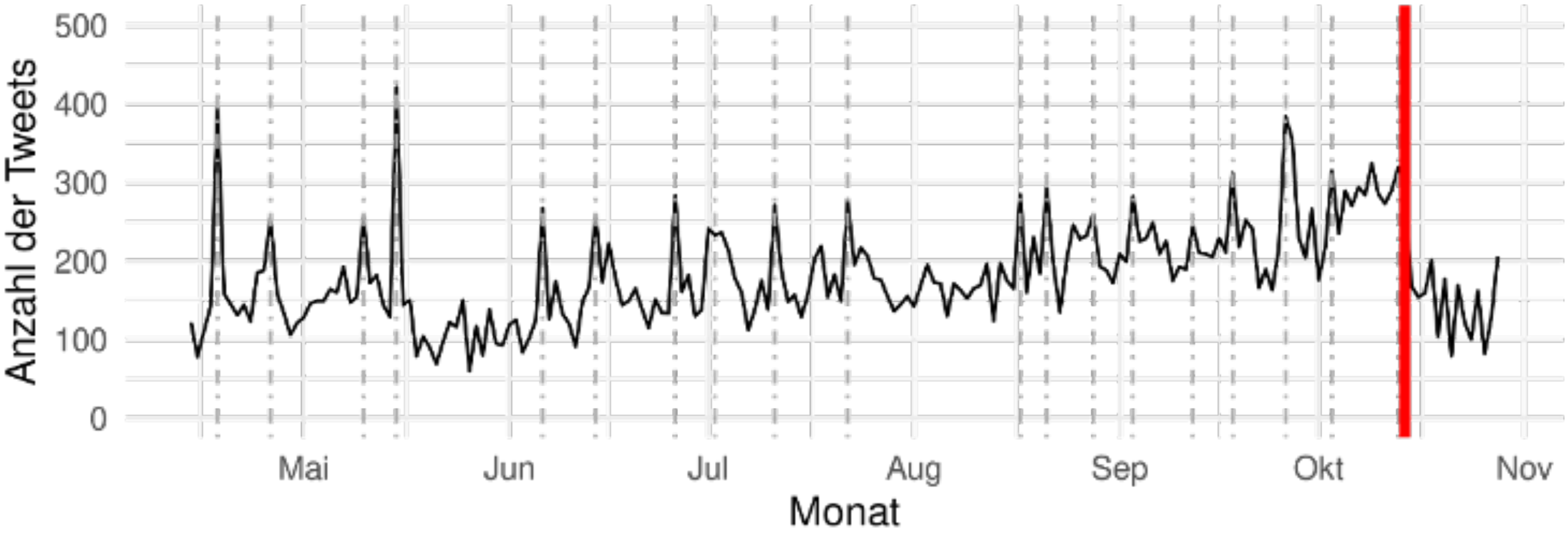
W_{im} = number of appearances of m in i

N_i = number of terms in i; sometimes also operationalized as number of terms bearing sentiment

LENNERT (2023): ANALYZING THE TWITTER DISCOURSE OF BAVARIAN POLITICIANS

- “Wahlkampf in Sozialen Medien – Eine Inhaltsanalyse der Twitter-Kommunikation politischer Eliten zur Landtagswahl in Bayern 2018”
- Descriptive study of the elite discourse during the election campaigns in Bavaria
- Sample: all candidates of different parties
- What are politicians discussing on Twitter?
  ⇒ Strategy: look at terms that are exclusive for documents

LENNERT (2023): ANALYZING THE TWITTER DISCOURSE OF BAVARIAN POLITICIANS



TFIDF

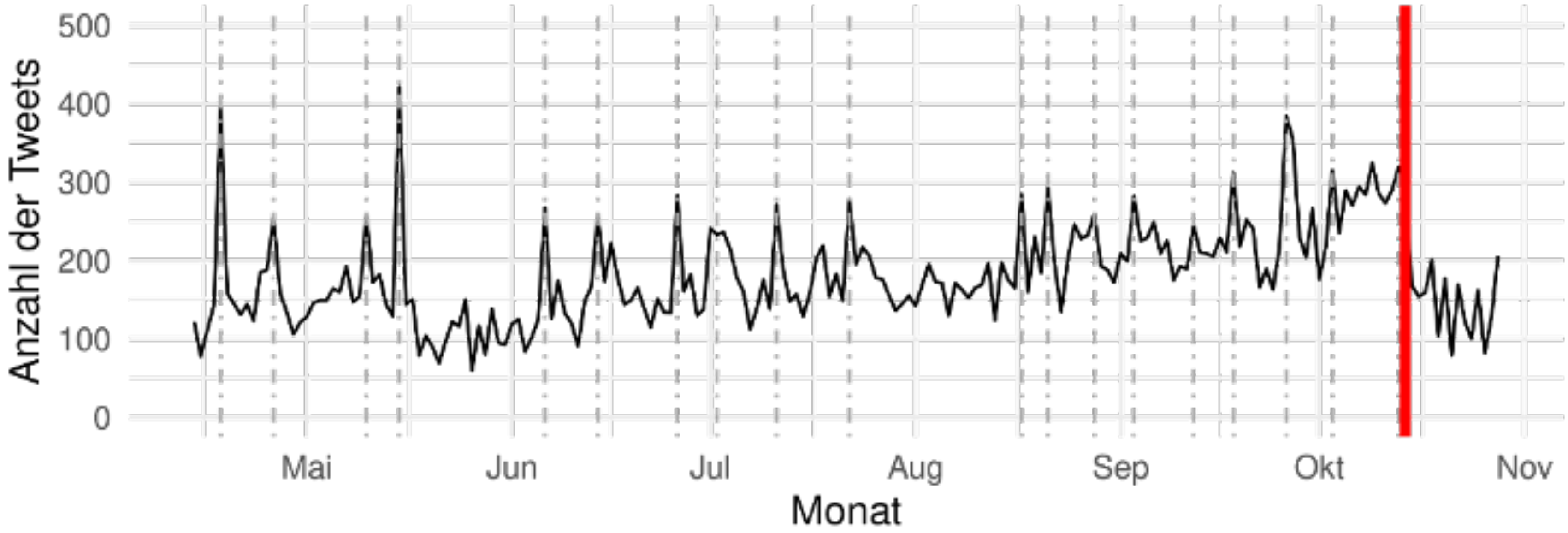
$\Longrightarrow$ Strategy: look at terms that are exclusive for documents

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

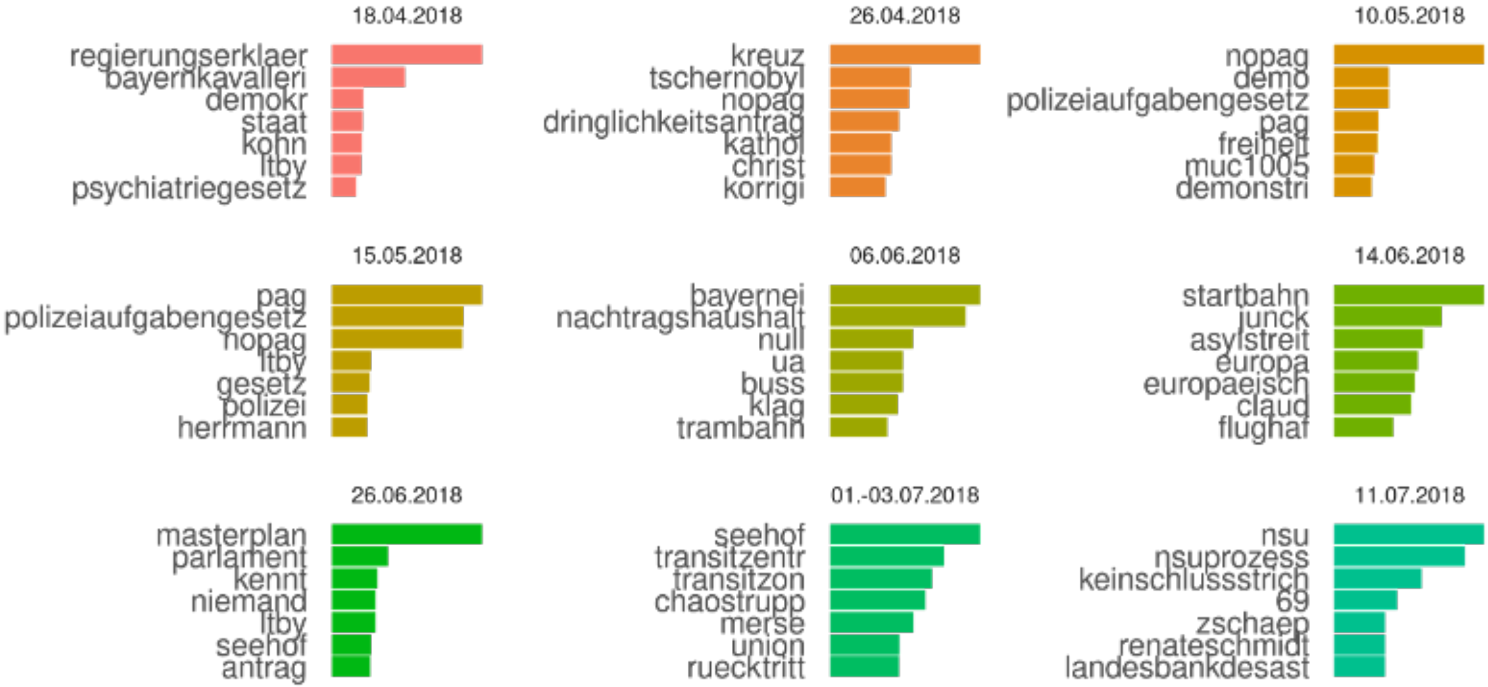
$$\text{TF}(t, d) = \frac{\text{frequency of term } t \text{ in document } d}{\text{total number of terms in document } d}$$

$$\text{IDF}(t) = \log\left(\frac{\text{total number of documents}}{\text{number of documents containing term } t}\right)$$

LENNERT (2023): ANALYZING THE TWITTER DISCOURSE OF BAVARIAN POLITICIANS



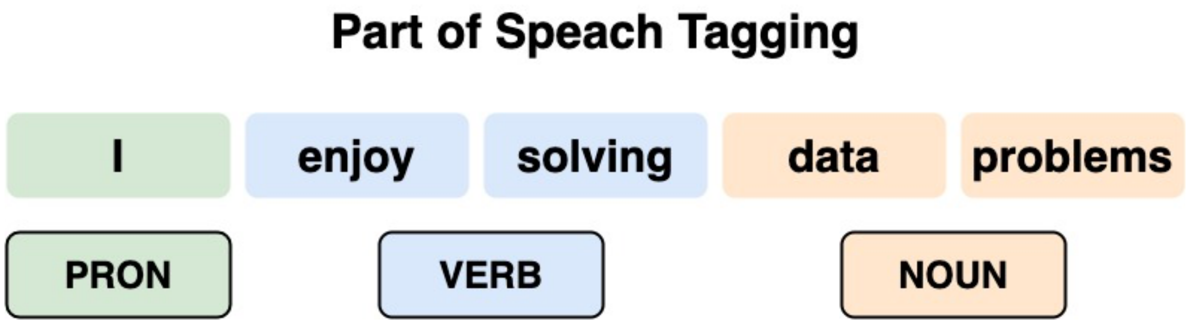
LENNERT (2023): ANALYZING THE TWITTER DISCOURSE OF BAVARIAN POLITICIANS



POS-TAGGING

- In language, certain kinds of terms have certain functions
  - noun, verb, pronoun, preposition, adverb, conjunction, participle, and article
  - For extensive descriptions of particular functions, read Jurafsky & Martin (forthcoming), chapter 8
- These terms are different **parts-of-speech (POS)**

POS-TAGGING


Part of Speach Tagging
I
enjoy
solving
data
problems
PRON
VERB
NOUN

POS-TAGGING

- Is performed model-based (for description, see Jurafsky & Martin (forthcoming), chapter 8)

Why is it good for us?

- Language is far too complex
- Knowing terms' POS-label allows us to filter unnecessary noise
- Example: Bail (2016) only focuses on nouns
  ⇒ assumption: nouns capture the substantial things that are talked about (e.g., people, issues, etc.)
- Decision has to be theoretically motivated

Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coordinating conjunction	*and, but, or*	PDT	predeterminer	*all, both*	VBP	verb non-3sg present	*eat*
CD	cardinal number	*one, two*	POS	possessive ending	*'s*	VBZ	verb 3sg pres	*eats*
DT	determiner	*a, the*	PRP	personal pronoun	*I, you, he*	WDT	wh-determ.	*which, that*
EX	existential 'there'	*there*	PRP$	possess. pronoun	*your, one's*	WP	wh-pronoun	*what, who*
FW	foreign word	*mea culpa*	RB	adverb	*quickly*	WP$	wh-possess.	*whose*
IN	preposition/ subordin-conj	*of, in, by*	RBR	comparative adverb	*faster*	WRB	wh-adverb	*how, where*
JJ	adjective	*yellow*	RBS	superlatv. adverb	*fastest*	$	dollar sign	$
JJR	comparative adj	*bigger*	RP	particle	*up, off*	#	pound sign	#
JJS	superlative adj	*wildest*	SYM	symbol	*+,%, &*	“	left quote	‘ or “
LS	list item marker	*1, 2, One*	TO	“to”	*to*	”	right quote	’ or ”
MD	modal	*can, should*	UH	interjection	*ah, oops*	(	left paren	[, (, {, <
NN	sing or mass noun	*llama*	VB	verb base form	*eat*	)	right paren	],), }, >
NNS	noun, plural	*llamas*	VBD	verb past tense	*ate*	,	comma	,
NNP	proper noun, sing.	*IBM*	VBG	verb gerund	*eating*	.	sent-end punc	. ! ?
NNPS	proper noun, plu.	*Carolinas*	VBN	verb past part.	*eaten*	:	sent-mid punc	: ; ... – -

Figure 8.1 Penn Treebank part-of-speech tags (including punctuation).

NAMED ENTITY RECOGNITION

- Named Entity Recognition (NER): identifying and classifying named entities
  ⟹ names of persons, organizations, locations, dates, etc.
- NER can be used to automatically extract structured information from unstructured text data

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces, counties
Facility	FAC	Bridges, buildings, airports
Vehicles	VEH	Planes, trains and automobiles

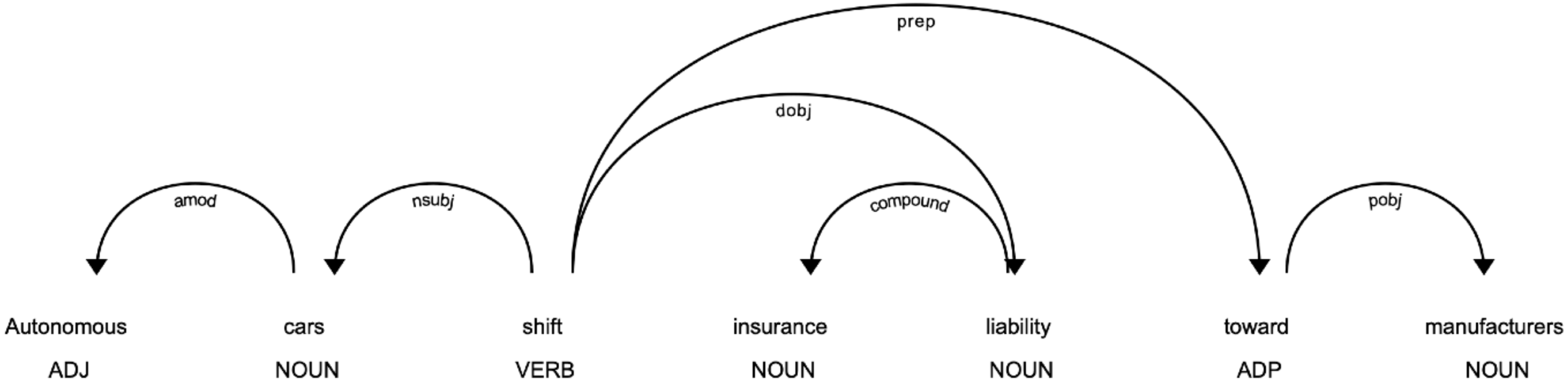
Figure 22.1 A list of generic named entity types with the kinds of entities they refer to.

Type	Example
People	*Turing* is often considered to be the father of modern computer science.
Organization	The *IPCC* said it is likely that future tropical cyclones will become more intense.
Location	The *Mt. Sanitas* loop hike begins at the base of *Sunshine Canyon*.
Geo-Political Entity	*Palo Alto* is looking at raising the fees for parking in the University Avenue district
Facility	Drivers were advised to consider either the *Tappan Zee Bridge* or the *Lincoln Tunnel*.
Vehicles	The updated *Mini Cooper* retains its charm and agility.

Figure 22.2 Named entity types with examples.

DEPENDENCY PARSING

- What's the relationship between different words/actors in sentences



DEPENDENCY PARSING

- Dependency parsing uncovers the relationships of entities
- Can help with
  - Sentiment analysis (who is described as what – also: by whom)
    $\Longrightarrow$ this approach may arguably bear more validity than topic models or word embeddings which are rather based on co-occurrence
  - interactions: “who does what to whom”

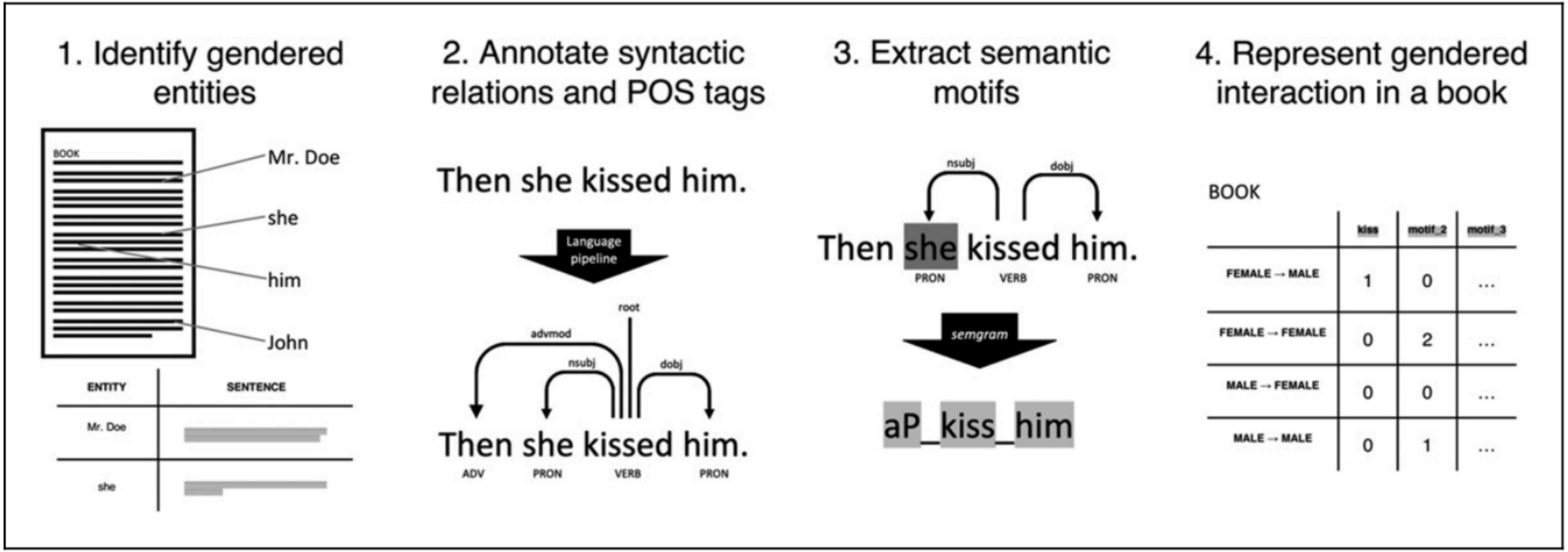
STUHLER 2022 – WHO DOES WHAT TO WHOM

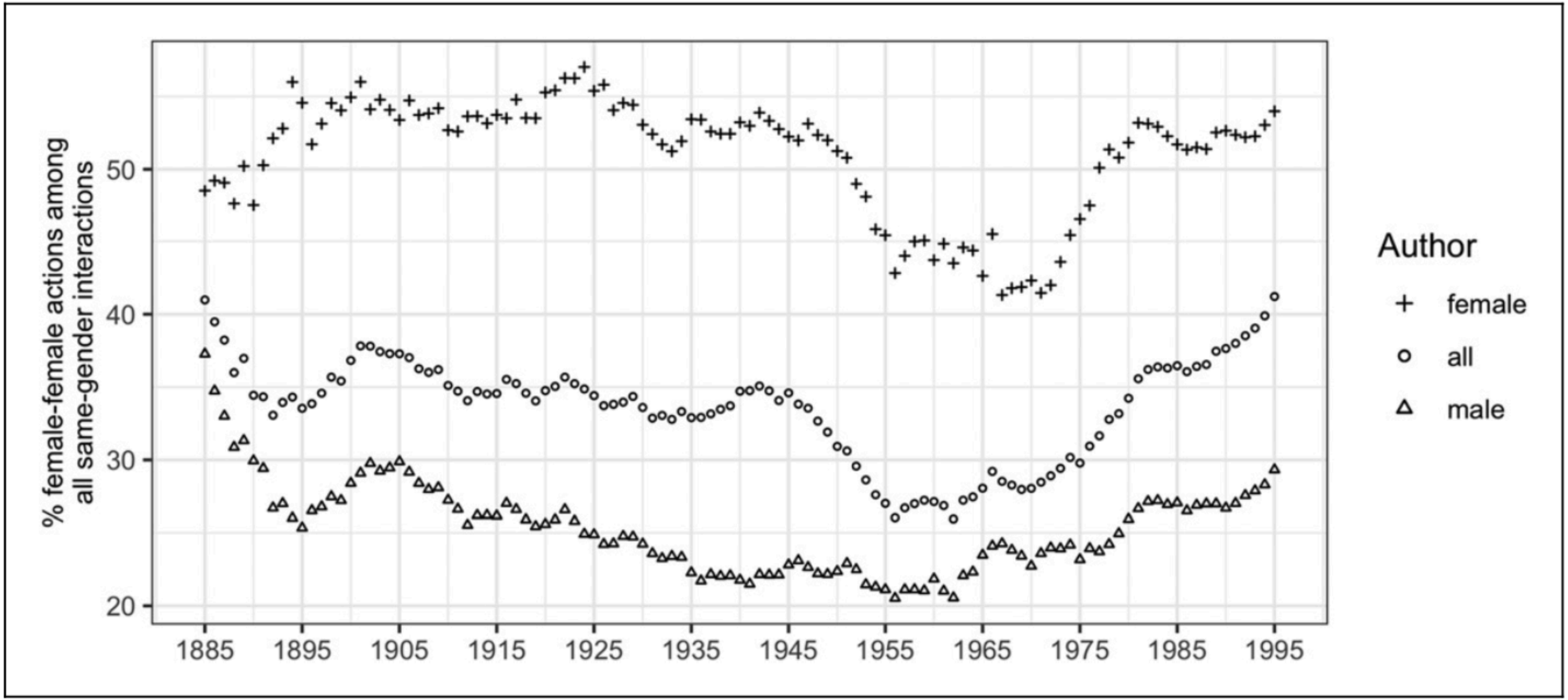
- Dependency parsing as valuable but underused tool for sociologists
- Provides framework to use it:
  - entity-centered – he has at least one entity of interest
  - components: “actions of an entity, treatments of an entity, agents acting upon an entity, patients acted upon by an entity, characterizations of an entity, and possessions of an entity” (p. 15)
- Goal: systematic extraction of relevant terms that are readily interpretable (e.g., “what men do to women”)

STUHLER 2022 – WHO DOES WHAT TO WHOM

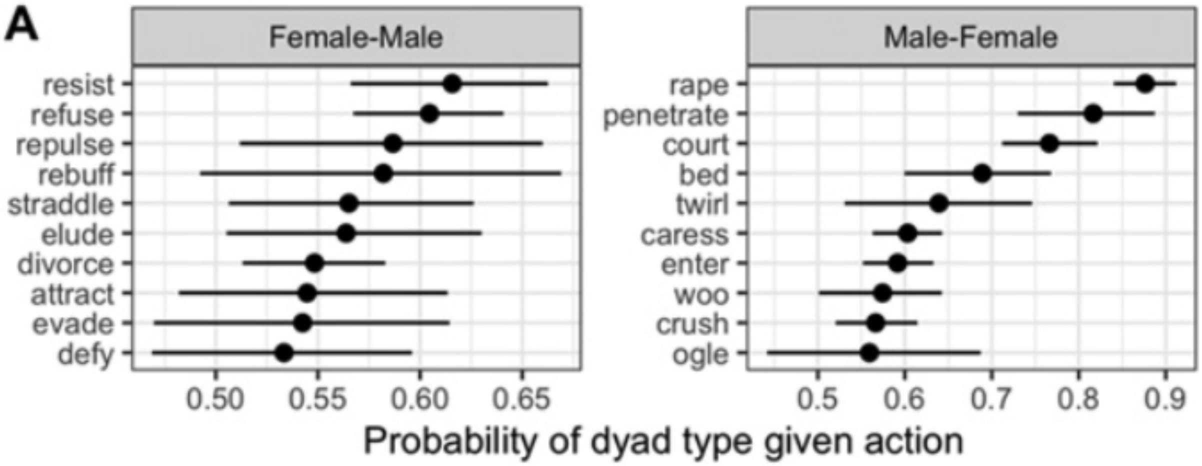
- Example: “what men do to women”
- Data: U.S. Novel Corpus (USNC); 9,088 American novels published between 1880 and 1990
- Identification of male and female agents based on first name and “Mr.,” “Mrs.,” “Miss,” and “Madame” and the pronouns “he,” “him,” “his,” “she,” and “her”
- Determines instances where a male/female person acted upon another male/female person

STUHLER 2022 – WHO DOES WHAT TO WHOM




% female-female actions among
all same-gender interactions
50
40
30
20
1885
1895
1905
1915
1925
1935
1945
1955
1965
1975
1985
1995
Author
female
all
male

STUHLER 2022 – WHO DOES WHAT TO WHOM



STUHLER 2022 – WHO DOES WHAT TO WHOM

- Significant effect of author’s gender on female-female interactions
- Men are described as “actionable” when it comes to sexual actions, women rather defensive
- However, over time acting agents’ gender given a particular action become less predictable – independent of author’s gender

MERCI

Felix Lennert
Institut für Soziologie

felix.lennert@uni-leipzig.de
www.uni-leipzig.de