UNIVERSITÄT ZU KÖLN

Counting Words

VL Sprachliche Informationsverarbeitung

Nils Reiter
nils.reiter@uni-koeln.de

November 2, 2023
Winter term 2023/24

INSTITUT FÜR DIGITAL HUMANITIES
UNIVERSITÄT ZU KÖLN

Section 1

Quantitatively Looking at Words

Word Counts

Count	Word
585	die
584	und
407	er
404	der
348	zu
311	sich
259	nicht
250	sie
243	in
243	den
233	war
218	Gregor
189	mit
178	das
176	auf
171	es
162	dem
155	hatte
137	ein
136	aber
133	daß
123	als
110	auch
107	Schwester
	...

Word Counts

Count	Word
585	die
584	und
407	er
404	der
348	zu
311	sich
259	nicht
250	sie
243	in
243	den
233	war
218	Gregor
189	mit
178	das
176	auf
171	es
162	dem
155	hatte
137	ein
136	aber
133	daß
123	als
110	auch
107	Schwester
	...

- Number of words in a text
- Most frequent words (MFW) are function words
- ‘Content words’ that appear often indicate text content

Zipf's Law

Manning/Schütze, 1999, 23 ff.

- George Kingsley Zipf (1902–1950): American Linguist
- Basic property of human language
  - Frequency distribution of words (in a corpus) is stable
  - Word frequency is inversely proportional to its position in the ranking

$$f \propto \frac{1}{r}$$

(there is a constant k, such that $f \times r = k$)

Zipf's Law

Manning/Schütze, 1999, 23 ff.

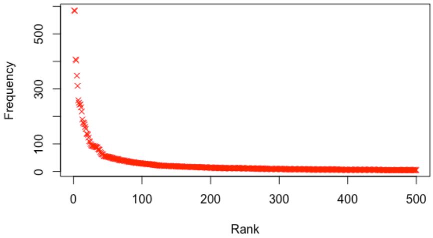

Figure: Words sorted after their frequency (red). Text: Kafka's "Die Verwandlung".

Zipf's Law

Manning/Schütze, 1999, 23 ff.

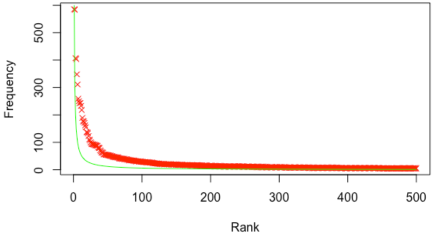

Figure: Words sorted after their frequency (red). Zipf distribution: $y = 600\frac{1}{x}$ (green). Text: Kafka's "Die Verwandlung".

Zipf's Law

Manning/Schütze, 1999, 23 ff.

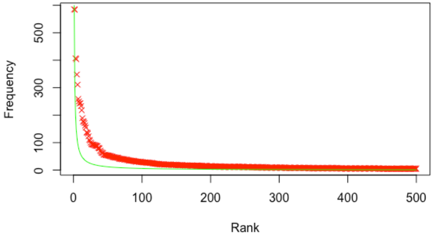

Figure: Words sorted after their frequency (red). Zipf distribution: $y = 600\frac{1}{x}$ (green). Text: Kafka's "Die Verwandlung".

Consequences

- Very few words appear with very high frequency
- The vast majority of words appear only once
  - It's difficult to learn something about these words!

Counting Words

- Absolute numbers are not that interesting
- Insights are only generated through comparison

Counting Words

- Absolute numbers are not that interesting
- Insights are only generated through comparison

Abs. number	Word form
20	women
67	woman
31	men
79	family
82	sister
83	friend
99	bath
117	father
133	man
144	sir

Table: Jane Austens's *Persuasion* (nouns)

Counting Words

- Absolute numbers are not that interesting
- Insights are only generated through comparison

Abs. number	Word form
20	women
67	woman
31	men
79	family
82	sister
83	friend
99	bath
117	father
133	man
144	sir

Table: Jane Austens's *Persuasion* (nouns)

Abs. number	Word form
0	friend
2	bath
11	women
23	men
30	father
68	woman
83	family
113	sir
121	man
282	sister

Table: Jane Austens's *Sense and Sensibility* (nouns)

Absolute Numbers

Word	Persuasion	Sense
woman	67	68
women	20	11
man	133	121
men	31	23
sister	82	282

…does it make sense to compare absolute numbers? No.

Absolute Numbers

Word	Persuasion	Sense
woman	67	68
women	20	11
man	133	121
men	31	23
sister	82	282

…does it make sense to compare absolute numbers? No.

- The texts/corpora do not have the same size
- Scaling using their length: Division by the total number of words

Absolute Numbers

Word	Persuasion		Sense	
woman	67	0.000 79 %	68	0.000 55 %
women	20	0.000 24 %	11	0.000 09 %
man	133	0.001 58 %	121	0.001 00 %
men	31	0.000 37 %	23	0.000 19 %
sister	82	0.000 97 %	282	0.002 33 %

...does it make sense to compare absolute numbers? No.

- The texts/corpora do not have the same size
- Scaling using their length: Division by the total number of words
- Visible changes: Proportion of “sister”: $3.4 \rightarrow 2.4$

Scaling

- Number of words: Result of a measurement
- If measuring in different scenarios, it's important to scale the results
  - "In a text that is much shorter, there are much less chances for a certain word to be used."

Scaling

- Number of words: Result of a measurement
- If measuring in different scenarios, it's important to scale the results
    - "In a text that is much shorter, there are much less chances for a certain word to be used."

Recipe

- Divide the result of the measurement by the theoretical maximum
- How many chances are there for "sister" to be used?
    - As many as there are words in the text
- Thus, we divide by the total number of words

Scaling

- Number of words: Result of a measurement
- If measuring in different scenarios, it's important to scale the results
  - "In a text that is much shorter, there are much less chances for a certain word to be used."

Recipe

- Divide the result of the measurement by the theoretical maximum
- How many chances are there for "sister" to be used?
  - As many as there are words in the text
- Thus, we divide by the total number of words

- It's not always obvious how to scaled
- When reading research: Was it scaled, and how?

Types and Tokens

Manning/Schütze, 1999, 21 f.

- If a text has been tokenized, we can access individual units: Tokens
- Not all tokens are words: Punctuation, detached prefixes, ...

Types and Tokens

Manning/Schütze, 1999, 21 f.

- If a text has been tokenized, we can access individual units: Tokens
- Not all tokens are words: Punctuation, detached prefixes, ...
- We are often also interested in different tokens: Types

Types and Tokens

Manning/Schütze, 1999, 21 f.

- If a text has been tokenized, we can access individual units: Tokens
- Not all tokens are words: Punctuation, detached prefixes, ...
- We are often also interested in different tokens: Types

Example

the cat chases the mouse

Types and Tokens

Manning/Schütze, 1999, 21 f.

- If a text has been tokenized, we can access individual units: Tokens
- Not all tokens are words: Punctuation, detached prefixes, ...
- We are often also interested in different tokens: Types

Example

the cat chases the mouse

- Tokens: the, cat, chases, the, mouse
- Types: the, cat, chases, mouse

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!
  - “the dog barks loudly .”

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!
  - "the dog barks loudly ."
- Construct a sentence with 5 tokens and 4 types!

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!
  - “the dog barks loudly .”
- Construct a sentence with 5 tokens and 4 types!
  - “the cat loves the mouse”

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!
  - “the dog barks loudly .”
- Construct a sentence with 5 tokens and 4 types!
  - “the cat loves the mouse”
- Construct a sentence with 5 tokens and 1 type!

Type-Token-Ratio (TTR)

- What is the relation between number of tokens and number of types?
- Construct a sentence with 5 tokens and 5 types!
  - "the dog barks loudly ."
- Construct a sentence with 5 tokens and 4 types!
  - "the cat loves the mouse"
- Construct a sentence with 5 tokens and 1 type!
  - "dog dog dog dog dog" (not really a sentence …)
  - It's not possible to create a 'proper' sentence with 1 type

Type-Token-Ratio (TTR)

- Measure for ‘lexical variability’

$$TTR = \frac{\text{number of types}}{\text{number of tokens}}$$

- Max value: 1

Type-Token-Ratio (TTR)

- Measure for 'lexical variability'

$$TTR = \frac{\text{number of types}}{\text{number of tokens}}$$

- Max value: 1 (there cannot be more types than tokens)
- Min value: $\epsilon = \frac{1}{\text{very large number}}$

Type-Token-Ratio (TTR)

- Measure for ‘lexical variability’

$$TTR = \frac{\text{number of types}}{\text{number of tokens}}$$

- Max value: 1 (there cannot be more types than tokens)
- Min value: $\epsilon = \frac{1}{\text{very large number}}$
- Real (German) texts
  - $10\,000$ words (Wikipedia): $\frac{4021}{10\,000} = 0.4021$

TTR and Text Length

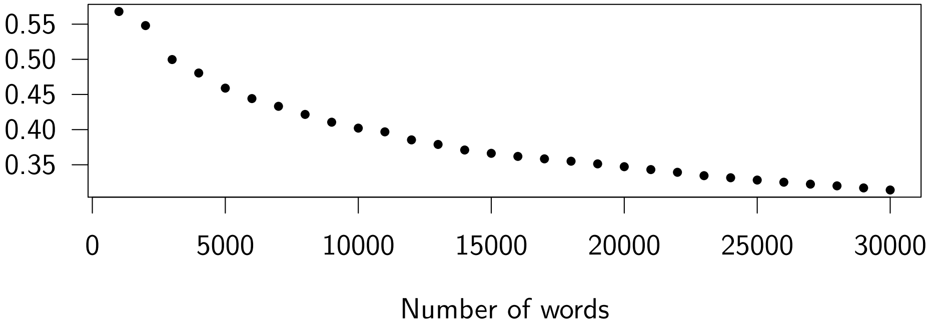

Figure: Type-Token-Ratio for increasing text lengths

TTR and Text Length

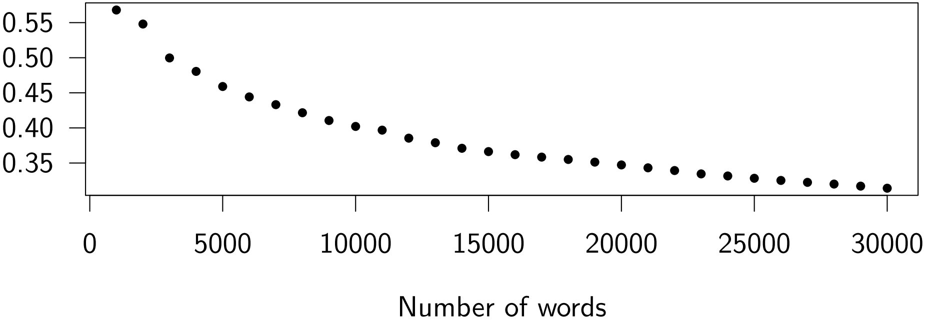

Figure: Type-Token-Ratio for increasing text lengths

- Increasing length → lower TTR!
- Why?

TTR and Text Length

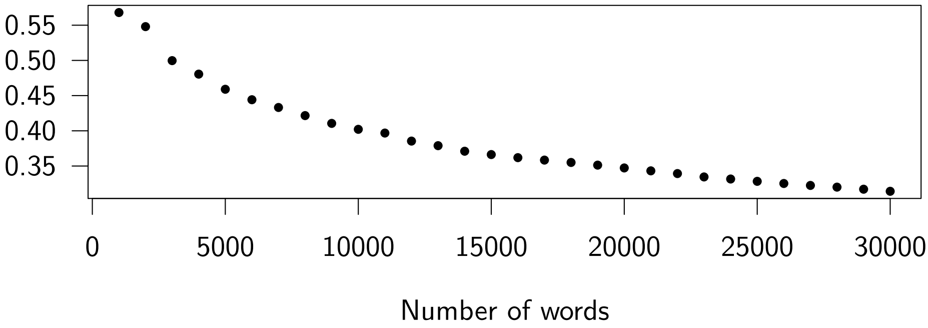

Figure: Type-Token-Ratio for increasing text lengths

- Increasing length → lower TTR!
- Why?– Zipf!

Standardized TTR (STTR)

- Calculate TTR over windows of fixed size (e.g., 1000 words)
- Calculate arithmetic mean over TTR values

Standardized TTR (STTR)

- Calculate TTR over windows of fixed size (e.g., 1000 words)
- Calculate arithmetic mean over TTR values

$$TTR_n = \frac{\text{number of types in } n\text{th window}}{\text{number of tokens in } n\text{th window}}$$

Standardized TTR (STTR)

- Calculate TTR over windows of fixed size (e.g., 1000 words)
- Calculate arithmetic mean over TTR values

$$
\begin{aligned}
TTR_n &= \frac{\text{number of types in } n\text{th window}}{\text{number of tokens in } n\text{th window}} \\
STTR &= \frac{1}{w}\sum_{i=0}^{w} TTR_i
\end{aligned}
$$

n-grams

- So far: Individual tokens
- But: Context is important for linguistic expressions

n-grams

- So far: Individual tokens
- But: Context is important for linguistic expressions
- n-gram: A list of n directly adjacent tokens
    - Popular choices for n: 2 to 4

n-grams

- So far: Individual tokens
- But: Context is important for linguistic expressions
- n-gram: A list of n directly adjacent tokens
    - Popular choices for n: 2 to 4

Example

The dog barks.

- 1-grams: “the”, “dog”, “barks”, “.”
- 2-grams (bigrams): “the dog”, “dog barks”, “barks .”
- 3-grams (trigrams): “the dog barks”, “dog barks .”

Counting Bigrams

- Simple idea: We count bigrams (i.e., pairs of subsequent tokens)

Counting Bigrams

- Simple idea: We count bigrams (i.e., pairs of subsequent tokens)
- Corpus: Wikipedia pages (first $10\,000$ sentences)

Bigram	Frequency
wurde er	630
in der	623
wurde die	501
an der	386
mit dem	363
in die	362
in den	329
mit der	312
wurde das	291
wurde der	291
für die	248
er in	193
war er	181
von der	174
wo er	169
bei den	168
bei der	166
und wurde	165
an die	161
und die	150
er die	143
er als	142
er mit	142
wurden die	142
auf dem	135
für den	133
wurde sie	127
er zum	123
auf der	122

Counting Bigrams

- Simple idea: We count bigrams (i.e., pairs of subsequent tokens)
- Corpus: Wikipedia pages (first $10\,000$ sentences)
- Again, there are a lot of function words. Why?

Bigram	Frequency
wurde er	630
in der	623
wurde die	501
an der	386
mit dem	363
in die	362
in den	329
mit der	312
wurde das	291
wurde der	291
für die	248
er in	193
war er	181
von der	174
wo er	169
bei den	168
bei der	166
und wurde	165
an die	161
und die	150
er die	143
er als	142
er mit	142
wurden die	142
auf dem	135
für den	133
wurde sie	127
er zum	123
auf der	122

Counting Bigrams

- Simple idea: We count bigrams (i.e., pairs of subsequent tokens)
- Corpus: Wikipedia pages (first $10\,000$ sentences)
- Again, there are a lot of function words. Why?
- Zipf's law: Two words that are highly frequent have much higher chance to co-occur with high frequency

Bigram	Frequency
wurde er	630
in der	623
wurde die	501
an der	386
mit dem	363
in die	362
in den	329
mit der	312
wurde das	291
wurde der	291
für die	248
er in	193
war er	181
von der	174
wo er	169
bei den	168
bei der	166
und wurde	165
an die	161
und die	150
er die	143
er als	142
er mit	142
wurden die	142
auf dem	135
für den	133
wurde sie	127
er zum	123
auf der	122

Section 2

Automatic Prediction of Linguistic Properties

Automatic Prediction of Linguistic Properties

- Linguistic understanding: Part of speech, lemma, syntactic structure, semantic representation, …
- Detection of content-related aspects: Named entities, sentiment, speech acts, …
- Applications: Machine translation, question answering, dialoge systems, …

Automatic Prediction of Linguistic Properties

- Linguistic understanding: Part of speech, lemma, syntactic structure, semantic representation, …
- Detection of content-related aspects: Named entities, sentiment, speech acts, …
- Applications: Machine translation, question answering, dialoge systems, …
- How to do that? Machine learning, nowadays

From Rules to Neural Networks

Rule-based part of speech tagging

```
# list of German determiners
determiners = ["der","die","ein",...]

for token in tokens:
  if token[0].islower() and
    token.endswith("en"):
    return "VERB"
  elif token[0].isupper():
    return "NAME"
  else:
     if token in determiners:
       return "DET"
...
```

From Rules to Neural Networks

Rule-based part of speech tagging

```
# list of German determiners
determiners = ["der","die","ein",...]

for token in tokens:
  if token[0].islower() and
    token.endswith("en"):
    return "VERB"
  elif token[0].isupper():
    return "NAME"
  else:
     if token in determiners:
       return "DET"
...
```

Which token properties are used here?

From Rules to Neural Networks

Rule-based part of speech tagging

```
# list of German determiners
determiners = ["der","die","ein",...]

for token in tokens:
  if token[0].islower() and
    token.endswith("en"):
    return "VERB"
  elif token[0].isupper():
    return "NAME"
  else:
     if token in determiners:
       return "DET"
...
```

Which token properties are used here?

- Casing (upper/lower)
- Suffix (en)
- Word list (Determiners)

From Rules to Neural Networks

Rule-based part of speech tagging

```python
# list of German determiners
determiners = ["der","die","ein",...]

for token in tokens:
  if token[0].islower() and
    token.endswith("en"):
    return "VERB"
  elif token[0].isupper():
    return "NAME"
  else:
     if token in determiners:
       return "DET"
...
```

Which token properties are used here?

- Casing (upper/lower)
- Suffix (en)
- Word list (Determiners)

Which properties are *not* used?

From Rules to Neural Networks

Rule-based part of speech tagging

```
# list of German determiners
determiners = ["der","die","ein",...]

for token in tokens:
  if token[0].islower() and
    token.endswith("en"):
    return "VERB"
  elif token[0].isupper():
    return "NAME"
  else:
     if token in determiners:
       return "DET"
...
```

Which token properties are used here?

- Casing (upper/lower)
- Suffix (en)
- Word list (Determiners)

Which properties are *not* used?

- Prefixes
- Token length
- Sequence: Previous tag

From Rules to Neural Networks

‘Classical’ machine learning

```
tokens = ["the", "dog", "barks"]
tags = ["DET", "NN", "VBZ"]

table = extract_features(tokens)

model = train(table, tags)
```

- Token properties → features
- Feature extraction / feature engineering
  - Finding useful features based on domain knowledge (e.g., linguistic knowledge)
  - ‘Playground’: What works well can really only be known after experiments

From Rules to Neural Networks

'Classical' machine learning

```
tokens = ["the", "dog", "barks"]
tags = ["DET", "NN", "VBZ"]

table = extract_features(tokens)

model = train(table, tags)
```

- Token properties $\rightarrow$ features
- Feature extraction / feature engineering
  - Finding useful features based on domain knowledge (e.g., linguistic knowledge)
  - 'Playground': What works well can really only be known after experiments
- Training: Estimate which features in which order allow best decisions
  - A large collection of algorithms has been developed: Decision trees, support vector machines, naive Bayes, ...
  - Training data needed!

From Rules to Neural Networks

‘Classical’ machine learning

- Annotated data
  - Used for training
  - Used for evaluation

From Rules to Neural Networks

'Classical' machine learning

- Annotated data
  - Used for training
  - Used for evaluation
- Three stages / contexts (and we need to know in which we are)
  - Training (train a model with annotated data)
  - Testing (test an existing model on annotated data)
  - Application (use an existing model)

From Rules to Neural Networks

‘Classical’ machine learning

- Annotated data
  - Used for training
  - Used for evaluation
- Three stages / contexts (and we need to know in which we are)
  - Training (train a model with annotated data)
  - Testing (test an existing model on annotated data)
  - Application (use an existing model)
- This still applies in the deep learning realm

From Rules to Neural Networks

Deep learning

- No more feature engineering
  - Let the computer figure out what it needs to know
- More computing (and more data)
- Black box
  - Intermediate states not interpretable for us humans
  - Only input and output can be understood

Machine Learning

- Collection of techniques for automatic
  - decision making
  - pattern detection
  - data analysis
- Machine learning vs. rule-based systems
  - Rule-based: Decision rules are hand-coded
    - if/then/else, ...
  - Machine learning: Decision rules are 'learned' from data
  - Data is used to estimate weights and criteria

Understanding Machine Learning

- Levels of understanding
  - Intuition
  - Formalization (math)
  - Implementation (code)

Understanding Machine Learning

- Levels of understanding
  - Intuition
  - Formalization (math)
  - Implementation (code)
- Areas to distinguish
  - Learning algorithm
  - Prediction model
  - Data preparation
    - Feature extraction (classical ML)
    - Shape of input data

Section 3

Types of Tasks

Task types

- Many ML/DL/NLP tasks are structurally similar
- Structurally similar: The same system can be used, all differences can be encoded in the training data

Task types

- Many ML/DL/NLP tasks are structurally similar
- Structurally similar: The same system can be used, all differences can be encoded in the training data

Example

- Part of speech tagging: Each token gets a label
  - Labels: NN, VBZ, DET, ADJA, ADJD, …
- Named entity recognition: Each token gets a label
  - O, B-PER, I-PER, B-LOC, I-LOC, …

Task types

- Many ML/DL/NLP tasks are structurally similar
- Structurally similar: The same system can be used, all differences can be encoded in the training data

Example

- Part of speech tagging: Each token gets a label
  - Labels: NN, VBZ, DET, ADJA, ADJD, …
- Named entity recognition: Each token gets a label
  - O, B-PER, I-PER, B-LOC, I-LOC, …

- Two important task types for NLP
  - Text classification: An entire text is classified (e.g., genre, sentiment, …)
  - Sequence labeling: Each individual word is classified (e.g., pos-tagging, …)

Task types

Text classification

- Texts belong to a class of texts

Examples

- Customer reviews $\rightarrow$ sentiment
- Novel $\rightarrow$ genre (fiction, non-fiction, ...)
- Posting $\rightarrow$ $\pm$ hate speech
- E-mail $\rightarrow$ $\{$spam, not spam, really important$\}$

Task types
Sequence labeling

- Words (or sequences of words) belong to classes
  - Sequence labeling: Classification + sequential dependency between classes

Examples

- Words $\rightarrow$ part of speech (noun, verb, adjective, ...)
- Words $\rightarrow$ proper noun
- Paragraphs $\rightarrow$ $\pm$ narrative scene
- **?** Collected works by Shakespeare $\rightarrow$ $\{$comedy, tragedy$\}$

Task types

Sequence labeling

- Words (or sequences of words) belong to classes
  - Sequence labeling: Classification + sequential dependency between classes

Examples

- Words $\rightarrow$ part of speech (noun, verb, adjective, ...)
- Words $\rightarrow$ proper noun
- Paragraphs $\rightarrow$ $\pm$ narrative scene
- **?** Collected works by Shakespeare $\rightarrow$ {comedy, tragedy}
  - Sequence of works probably irrelevant

Section 4

Summary

Summary

- Quantitatively looking at Words
  - Most frequent words are not the most informative words
  - Zipf distribution
  - Type-token ratio as a measure of lexical diversity
  - n-grams: Look at multiple tokens at once
- Predicting linguistic properties
  - From rules to neural networks
- Task types
  - Text classification
  - Sequence labeling

Section 5

Exercise

Übung 1

Besorgen Sie sich auf `https://opendiscourse.de/` Reden von zwei verschiedenen Politiker:innen aus unterschiedlichen Parteien, so dass sie insgesamt pro Person mehr als 10000 Wörter haben. Schreiben Sie dann in einer Programmiersprache Ihrer Wahl ein Programm, das die type-token-ratio für beide berechnet. Abgabe in Ilias bis zum 08.11.