Topic Modelling

Implementation des LDA Modells (Gensim) auf ein Korpus von Interviewfragen des NDR Corona Podcast

Inhalt

- Einführung in Topic Modelling
- LDA-Modell von Gensim
- Anwendung auf Corona Podcast
  - Implementation in Python
  - Ergebnisse
- Fazit/Ausblick

1. Topic Modelling

Topic Modelling

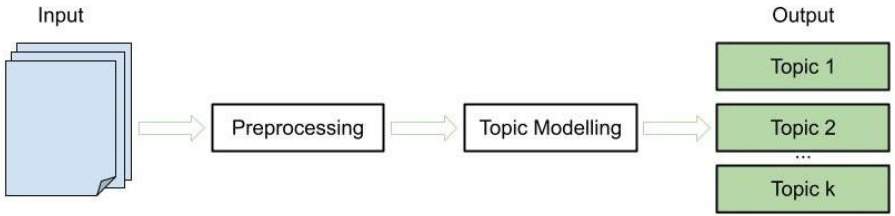


Abb. 1: vereinfachter Ablauf Topic Modelling

- Verfahren zur automatisierten Inhaltsanalyse
- Gebiet des unüberwachten maschinellen Lernens zuzuordnen
- Beschreibung bzw. Zusammenfassung der latenten Struktur großer Datensammlungen
- Berechnung von Themen aus Worthäufigkeiten in Dokumenten
- bekannteste Verfahren: LDA, CTM, STM

Topic Modelling bietet Methoden zum automatischen Organisieren, Verstehen, Suchen und Zusammenfassen großer Datenmengen an.

- Aufdecken latenter Themen in Sammlungen
- Klassifizierung der Elemente einer Sammlung
- Optimierung von Suchprozessen

Topic Modelling

Annahme:

Ein Korpus besteht aus *D* Dokumenten (einzelne Dokumente als d_1, d_2, ...) und *V* Wörtern bzw. Termen (= alle Wörter, die im gesamten Korpus vorkommen, einzelne Wörter als v_1, v_2, ...).

Es kommen latente Themen *K* zu unterschiedlichen Anteilen in den Dokumenten *D* vor, *K* wird vor Beginn festgelegt.

Alle Wörter *V* gehören zu unterschiedlichen Wahrscheinlichkeiten zu den Themen *K*.

(Vgl. Unkel 2020)

Topic Modelling

Ziel:

Berechnung der Matrix *D x K*: für jedes Dokument *d* und jedes Thema *k* wird die Wahrscheinlichkeit berechnet, dass das Thema im Dokument vorkommt

Berechnung der Matrix *V x K*: für jedes Wort *w* und jedes Thema *k* wird die Wahrscheinlichkeit berechnet, dass das Wort im Thema vorkommt

➜ Beschreibung und Interpretation der Themen

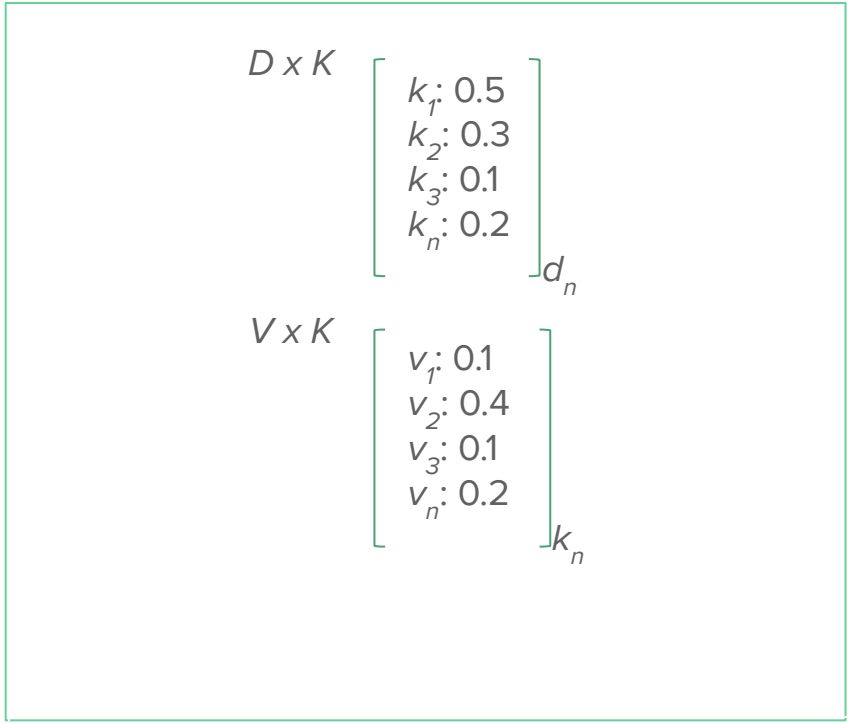


Abb. 2: Beispielmatrixen

2. Latent Dirichlet Allocation (LDA)

LDA

- generatives probabilistisches Modell für Datensammlungen (z.B. Textkorpora)
  - Technik um Dimension von Datensammlungen zu reduzieren
  - Anwendung auf Big Data
- Ziel: Finden von kurzen Beschreibungen zu den Elementen einer Datensammlung zur effizienten Verarbeitung großer Datenmengen
- Unterschied zu einem einfachen Dirichlet-multinomial Clustermodell: Dokumente werden mehrere Themen zugeordnet

(Vgl. Blei et al. 2003)

LDA

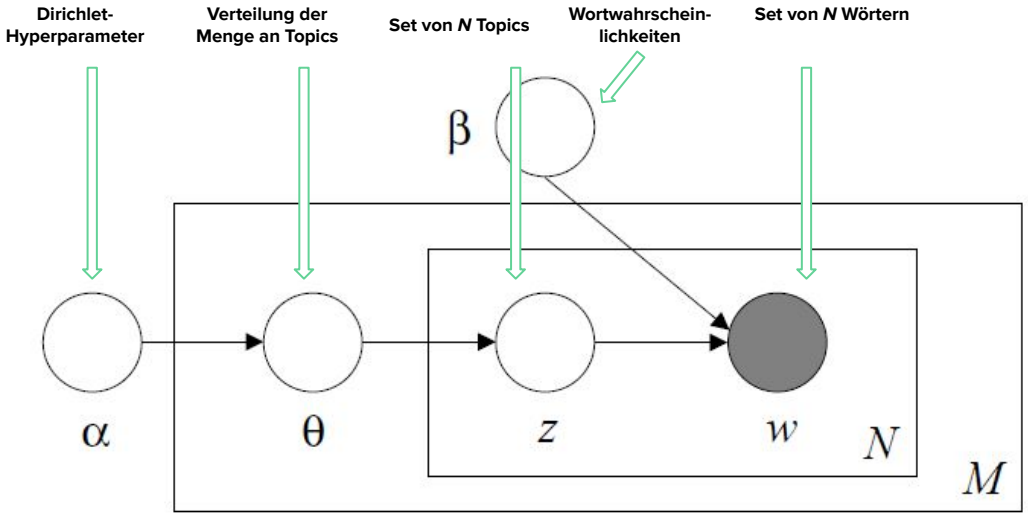
Annahmen:

- Jedes Dokument ist eine Sammlung von Wörtern bzw. ‘Bag-of-Words’.
  - Reihenfolge von Wörtern + grammatische Rollen werden nicht berücksichtigt
  - “assumption of exchangeability” (Blei et al. 2003: 994)
- Anzahl der Themen *k* ist im Voraus festgelegt
- Alle Themenzuordnungen mit Ausnahme des aktuellen Wortes sind korrekt
  - Zuordnung eines Themas zum aktuellen Wort wird anhand des generativen Modells der Dokumente durchgeführt

	the	red	dog	cat	eats	food
the red dog	1	1	1	0	0	0
cat eats dog	0	0	1	1	1	0
dog eats food	0	0	1	0	1	1
red cat eats	0	1	0	1	1	0

Abb. 3: Beispiel Bag-of-Words

LDA



M = Repräsentation von Dokumenten

N = Repräsentation von wiederholter Entscheidung von Topics und Wörtern innerhalb eines Dokuments

Abb. 4: Graphische Repräsentation des LDA-Modells (Blei et al. 2003: 997)

LDA

Ablauf - Berechnung der Wortwahrscheinlichkeiten (*V x K*):

1. Jedem Wort *w* eines Dokuments *d* wird eins von *n* Themen *K* zufällig zugeordnet. *n* wurde im Voraus festgelegt.
2. Für jedes Dokument *d* wird für jedes Wort *w* folgendes berechnet:
   a. ***p(k | d)***: Anteil der Wörter im Dokument *d*, die dem Thema *k* zugeordnet sind, das aktuelle Wort wird nicht betrachtet
   b. ***p(w | k)***: Anteil der Zuordnungen zum Thema k an allen Dokumenten, die von diesem Wort w stammen
3. Berechnung der Wahrscheinlichkeit für aktuelles Wort mit zugewiesenen Thema:

p(w mit k) = p(k | d) * p(w | k) (Vgl. Kulshrestha 2019)

3. LDA-Modell von Gensim

Was ist Gensim?

URL:
https://radimrehurek.com/gensim/_images/gensim_logo_positive_complete_tb.png

- frei verfügbare Open-Source Python-Bibliothek
- aufrufbar unter: https://radimrehurek.com/gensim/
- Bereitstellung unüberwachter maschineller Lernalgorithmen zur Verarbeitung unstrukturierter digitaler Texte
  - Darstellung dieser als semantische Vektoren
  - benötigt keine manuelle Annotation, nur ein “plain-text”-Korpus
  - z.B. Word2Vec, FastText, LSI, LDA
- Aufdecken semantischer Strukturen von Dokumenten durch Untersuchung statistischer “co-occurrence patterns” innerhalb eines Trainingskorpus

Gensim

Kernkonzepte:

- Dokumente: Textsequenzen (in Python: str)
- Korpora: Sammlungen von Dokumenten
  - Funktionen: Trainingskorpus (Suche nach gemeinsamen Themen, Initialisierung der internen Modellparameter), Testkorpus (Extraktion von Themen, Ähnlichkeitsabfragen, z.B. semantische Ähnlichkeit, Clustering)
- Vektoren: Darstellung der Dokumente als Vektor von Merkmalen (z.B. Anzahl von Wörtern, Absätzen, Schriftarten, etc.) zur mathematischen Verarbeitung
  - erlaubt Darstellung der Merkmale nur als einzelne Fließkommazahl
  - Nutzen eines Bag-of-Words-Vektors: enthält Häufigkeit jedes Wortes im Wörterbuch, Länge des Vektors entspricht der Anzahl der Einträge im Wörterbuch
  - erlaubt Vergleichbarkeit der Ähnlichkeit von Dokumenten eines Korpus anhand der Ähnlichkeit der Vektorrepräsentation
- Modelle: Algorithmus zur Umwandlung von Vektoren von einer Darstellung in eine andere
  - Transformation zwischen zwei Vektorräumen
  - Erlernen der Transformation während des Trainings

LDA-Modell als eine Umwandlung von Bag-of-Words-Zählungen in einen Themenraum mit geringerer Dimensionalität

- Themen von LDA als Wahrscheinlichkeits-verteilungen über Wörter
- Ableitungen der Verteilungen aus dem Trainingskorpus
- Dokumente als Mischung dieser Themen

LDA-Modell von Gensim

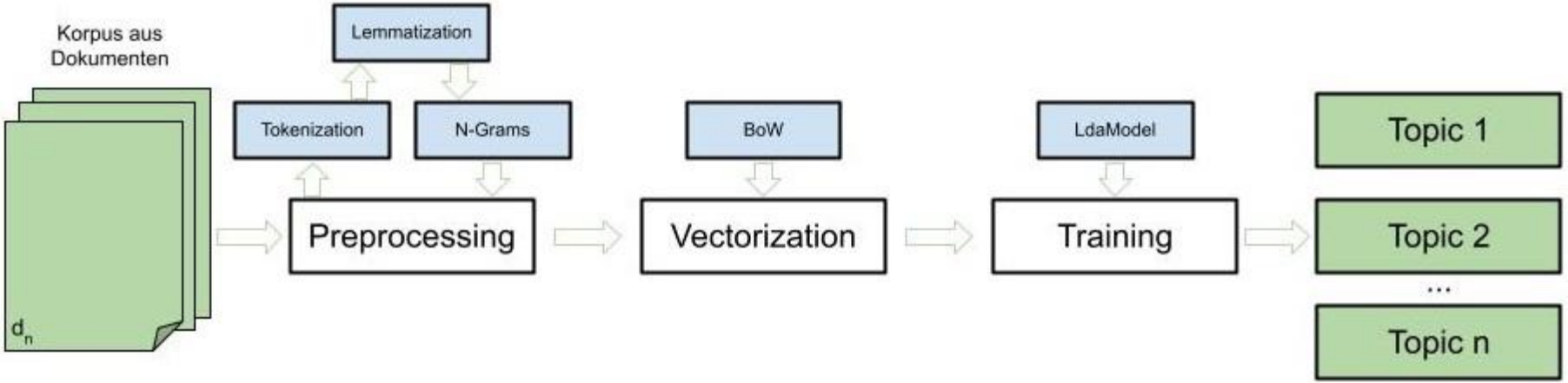


Abb.5: Ablauf LDA Gensim

4. Implementation LDA Corona Podcasts

Implementation LDA-Modell auf Corona Podcasts

- Implementation in Python 3.9
- Code verfügbar unter: https://github.com/ChristineSchaefer/topicmodel_coronapodcast.git
- Korpus: an die Wissenschaftlerin Sandra Ciesek und an den Wissenschaftler Christian Drosten gestellte **Fragen** im NDR Corona Podcast (Folge 1 - Folge 68)
  - Fragen extrahiert aus verfügbaren PDFs und abgespeichert in csv-Datei
- verwendete Bibliotheken: SpaCy, NLTK, Gensim
  - Training des LDA-Modells mit Lemmata (Unigram) anstelle von N-Grammen

Datengrundlage

	Sandra Ciesek	Christian Drosten
Anzahl der Interviews	8	57
Fragen/Dokumente	285	1135
Unique Tokens (insg.)	288	810
Unique Tokens (filter)	17	67
Dokumente (filter)	219	837

Tab. 1: Anzahl der verfügbaren Interviews, Fragen und Tokens als Input für das LDA-Modell

Implementation

Trainingsparameter:

- *num_topics*: Anzahl der Themen abhängig von Daten und Anwendung
  - relativ kleines Korpus ➔ Anzahl der Themen gering
- *chunksize*: Anzahl der Dokumente, die gleichzeitig verarbeitet werden sollen
- *passes*: Anzahl des Trainings auf das gesamte Korpus
- *iterations*: Anzahl der Wiederholung einer Schleife über jedes Dokument

```
# Set training parameters.
num_topics = 5      # number of topics
chunksize = 2000    # number of processed documents at a time
passes = 20         # how often the model will be trained
iterations = 400    # how often a particular loop will be repeated over each document
eval_every = 1      # evaluate each iteration

# make a index to word dictionary.
temp = dictionary[0]    # "load" the dictionary.
id2word = dictionary.id2token

# set model parameters
# Initialisierung (Training) des Transformationsmodells (LdaModel)
model = LdaModel(
    corpus=corpus,
    id2word=id2word,
    chunksize=chunksize,
    alpha='auto',
    eta='auto',
    iterations=iterations,
    num_topics=num_topics,
    passes=passes,
    eval_every=eval_every
)
```

Abb. 6: Code-Ausschnitt Training LDA

Evaluation

- Vergleich der beiden Modelle durch Berechnung der *Perplexity:*

$$perplexity(D_{\text{test}}) = \exp\left\{-\frac{\sum_{d=1}^{M}\log p(\mathbf{w}_d)}{\sum_{d=1}^{M} N_d}\right\}.$$

Abb. 7: Formale Darstellung der Perplexity (Brim et al. 2003: 1008)

- Evaluation nach jeder Runde des Trainings, nach 20 Runden:

	Sandra Ciesek	Christian Drosten
Perplexity	7.3	20.5
Perplexity pro Word	-2.870	-4.360
Topic Difference	0.005292	0.016556

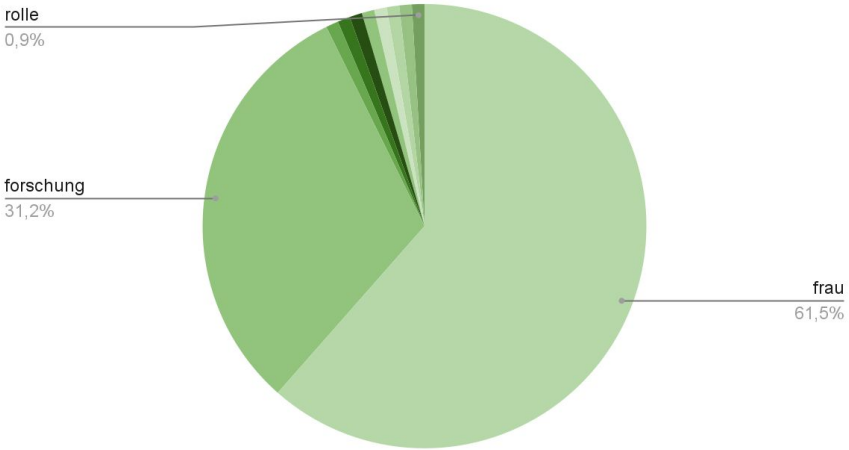
Tab. 2: Evalulationsergebnisse LDA

Sind die ermittelten Themen **verständlich**?
Sind die Themen **kohärent**?
Dient das Themenmodell dem **Zweck**, für den es verwendet wird?

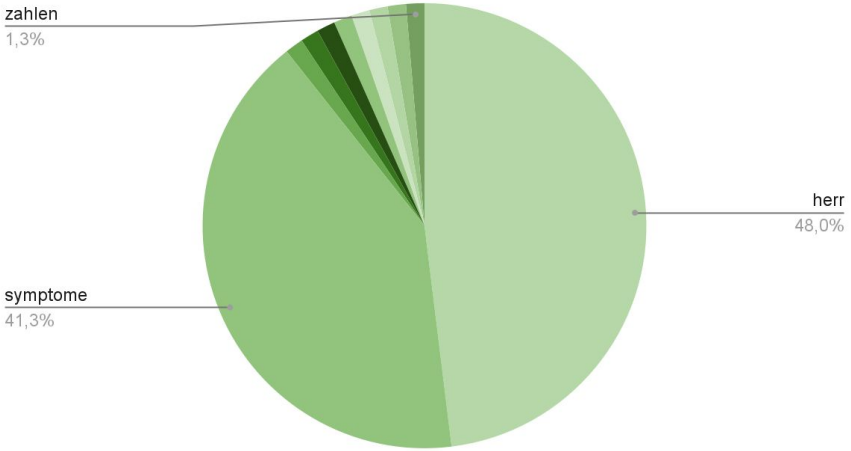
Ergebnisse

Topics Sandra Ciesek

Topic 1



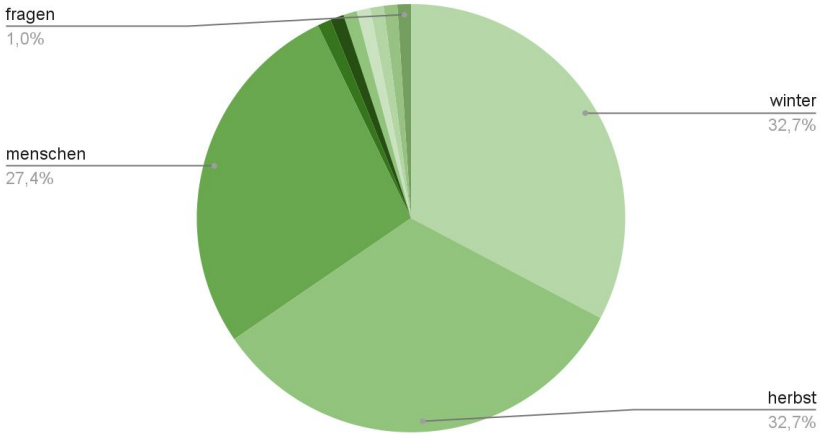
Topic 2



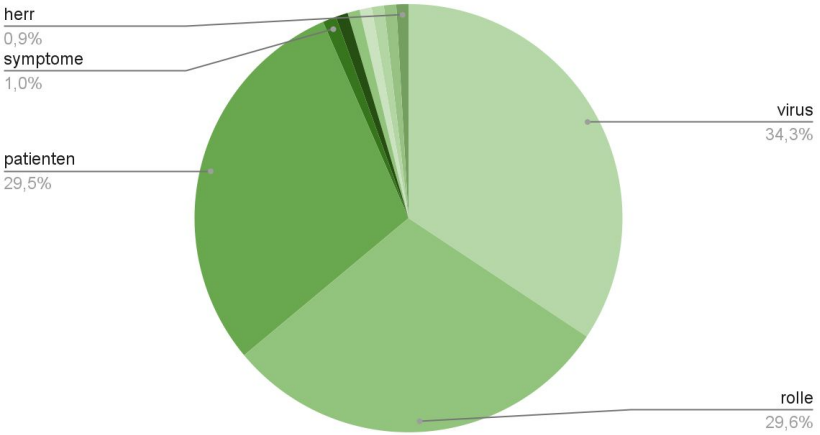
Ergebnisse

Topics Sandra Ciesek

Topic 3

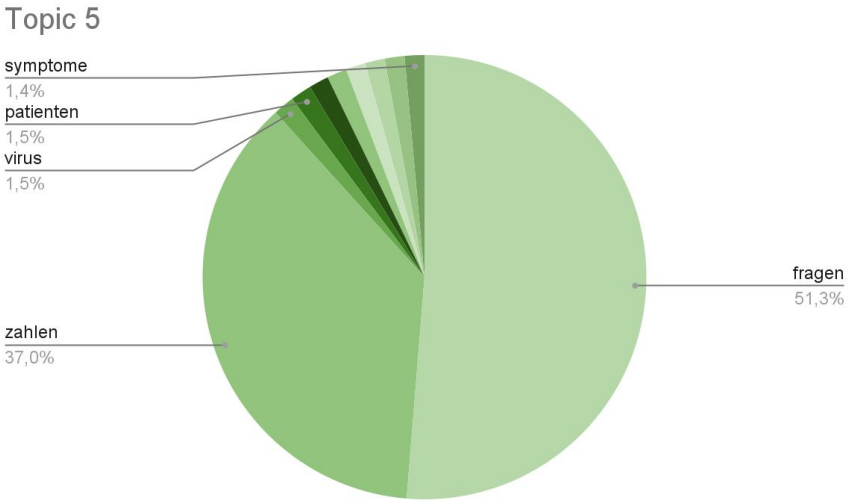


Topic 4



Ergebnisse

Topics Sandra Ciesek



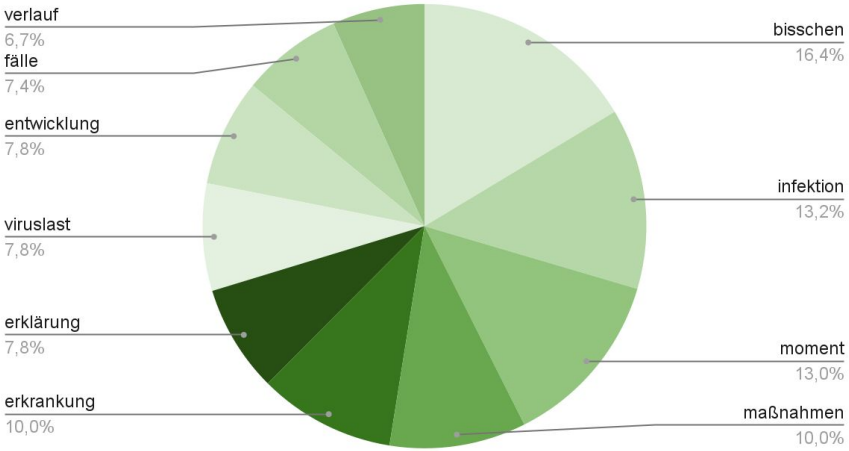
Topic 1	0.223
Topic 2	0.188
Topic 3	0.184
Topic 4	0.209
Topic 5	0.188

Tab. 3: Wahrscheinlichkeiten der Themen SC

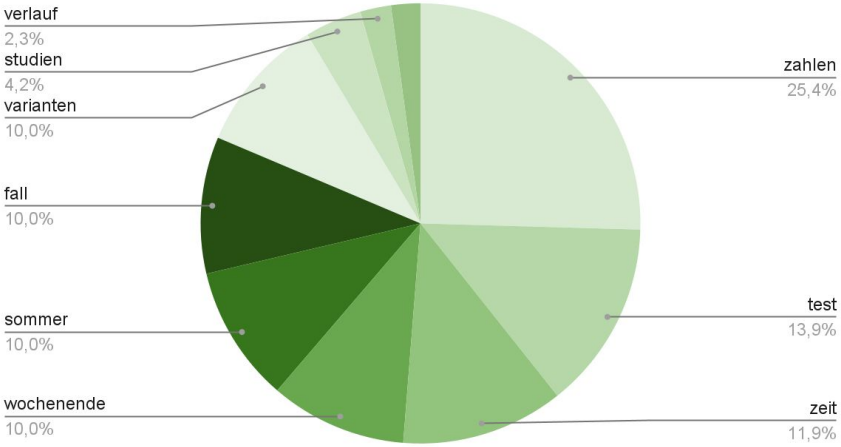
Ergebnisse

Topics Christian Drosten

Topic 1

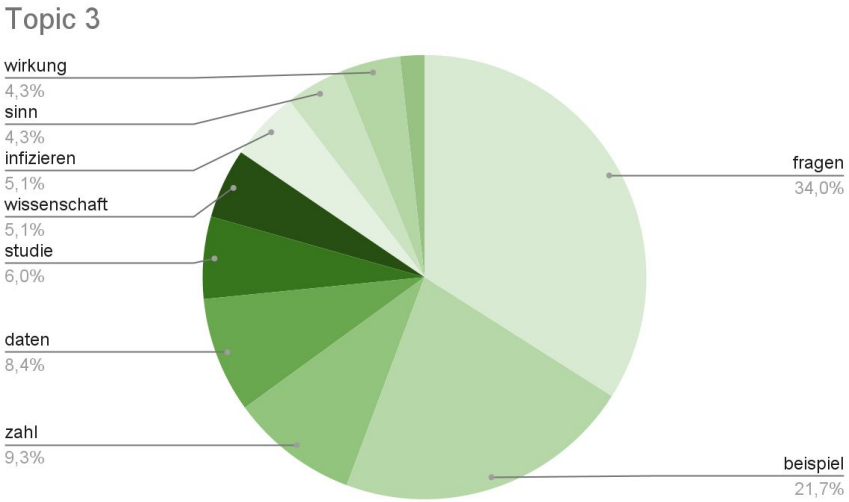


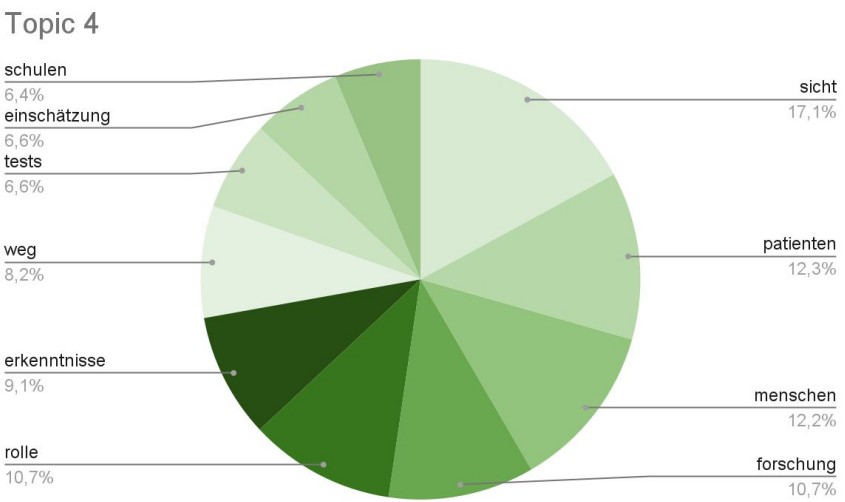
Topic 2



Ergebnisse

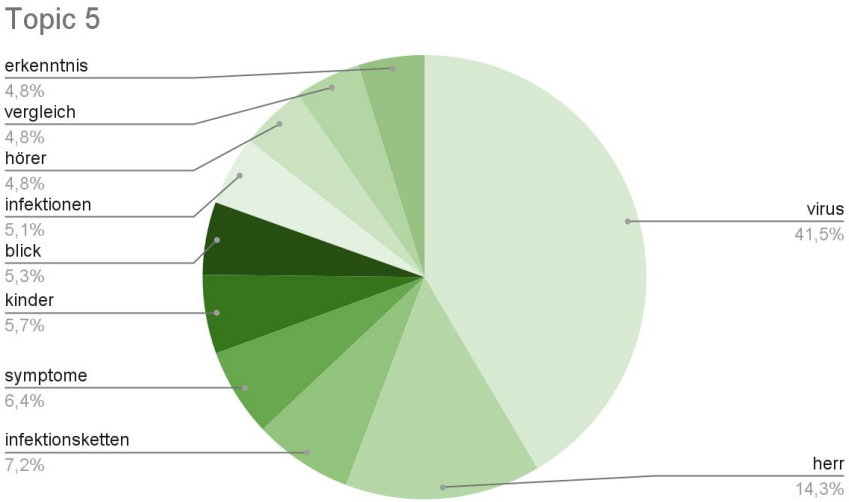
Topics Christian Drosten





Ergebnisse

Topics Christian Drosten



Topic 1	0.210
Topic 2	0.148
Topic 3	0.211
Topic 4	0.256
Topic 5	0.231

Tab. 4: Wahrscheinlichkeiten der Themen CD

5. Fazit und Ausblick

Fazit und Ausblick

- Anwendung des LDA Modells von Gensim auf ein Korpus von Fragen des NDR Corona Podcast als gute Möglichkeit, latente Themen innerhalb dieser sichtbar zu machen
  - weniger Arbeit als manuelle Einsicht
  - Interpretation der Themen (und deren Zusammenhänge) als möglich erachtet
- Aber:
  - zu kleines Korpus, um Ergebnisse als “wahr” anzuerkennen
  - Einbezug von Wörtern, die nicht bedeutungstragend für das Thema sind, aber trotzdem gewichtet wurden (z.B. Frau oder Herr)

➜ Anwendung auf ein größeres Korpus zum Trainieren und Testen

Fragen?

Literatur

Blei, David M. et al. (2003): *Latent dirichlet allocation*. J. Mach. Learn. Res. 3, null (3/1/2003). S. 993–1022.

Hoffman, Matthew D. et al. (2010): *Online Learning for Latent Dirichlet Allocation*. Advances in Neural Information Processing Systems. 23. S. 856-864.

Kulshrestha, Ria (2019): *A Beginner's Guide to Latent Dirichlet Allocation(LDA)*. A statistical model for discovering the abstract topics aka topic modeling. Abrufbar unter: https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2 (zuletzt aufgerufen: 18.10.2021)

Unkel, Julian (2020): *Topic Modeling*. In: Computational Methods in der politischen Kommunikationsforschung. Methodische Vertiefung: Computational Methods mit R und RStudio. Abrufbar unter: https://bookdown.org/joone/ComputationalMethods/topicmodeling.html (zuletzt aufgerufen: 18.10.2021)