Freie Universität Berlin

Similarity by Lyrics

Seminar Music Information Retrieval WS 16/17

David Bohn, Luca Keidel,
Fabian Reimeier und Friedrich Ueberreiter

Motivation

These:

Musik verschiedener Genre ist nicht nur durch musikalische Merkmale sondern auch durch die Lyrik vergleichbar.

Zentrale Frage:

„Wie kann ich die Ähnlichkeit von Musikstücken anhand ihrer Texte bestimmen?“

Naiver Ansatz:

Die Texte mit dem höchsten Anteil gemeinsamer Wörter sind die ähnlichsten.

Probleme

Polyseme: Überschätzung der Ähnlichkeit zweier Dokumenten durch Zählen gemeinsamer Ausdrücke mit unterschiedlicher Bedeutung
(Beispiel: Bank)

Synonyme: Unterschätzung der Ähnlichkeit von Dokumenten
(Beispiel: PKW und Auto)

Zero-frequency problem: Bei Verwendung nur sehr weniger Wörter des Vokabulars (Abstracts etc.)

Auch thematische eng verwandte Dokumente verwenden in der Regel nicht die exakt selben Wörter

Gliederung

1. Ansatz
2. Maschinelles Lernen
3. Latent Semantic Analysis *(LSA)*
4. Probabilistic Latent Semantic Analysis *(PLSA)*
5. Implementierung
6. Demo
7. Ausblick
8. Fazit

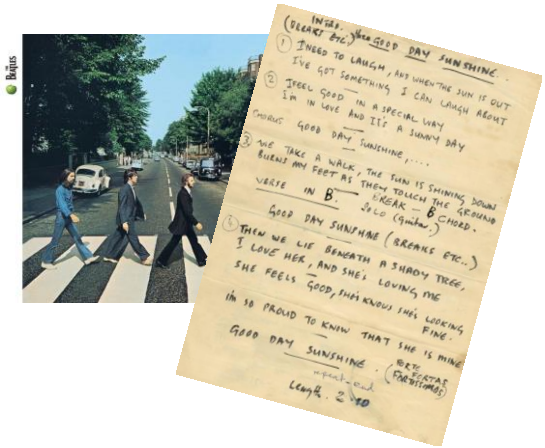
Ansatz

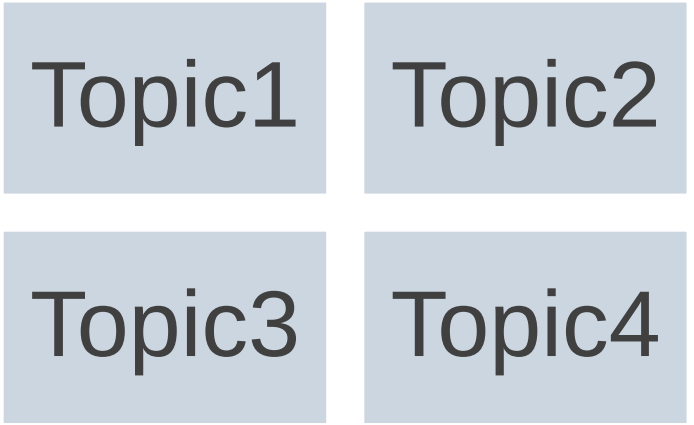
Zentrale Frage:

„Wie kann ich die Ähnlichkeit von Musikstücken anhand ihrer Texte bestimmen?“

Grundlegender Ansatz:

Klassifizierung der Liedtexte nach Inhalten/Topics.





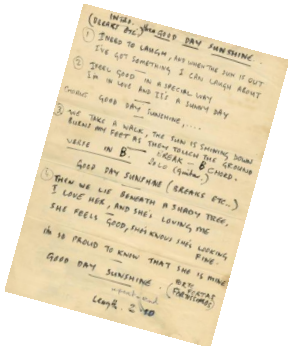
Maschinelles Lernen

- Überwachtes Lernen
- Teilüberwachtes Lernen
- **Unüberwachtes Lernen**

- Zwei Phasen:
  - **Lernphase**: Algorithmus wird mit Korpus (bspw. Liedtexte) angelernt
  - **Klassifikationsphase**: Bestimmung der Ähnlichkeit zweier Liedtexte anhand des Gelernten

LATENT SEMANTIC ANALYSIS

Latent Semantic Analysis (LSA)



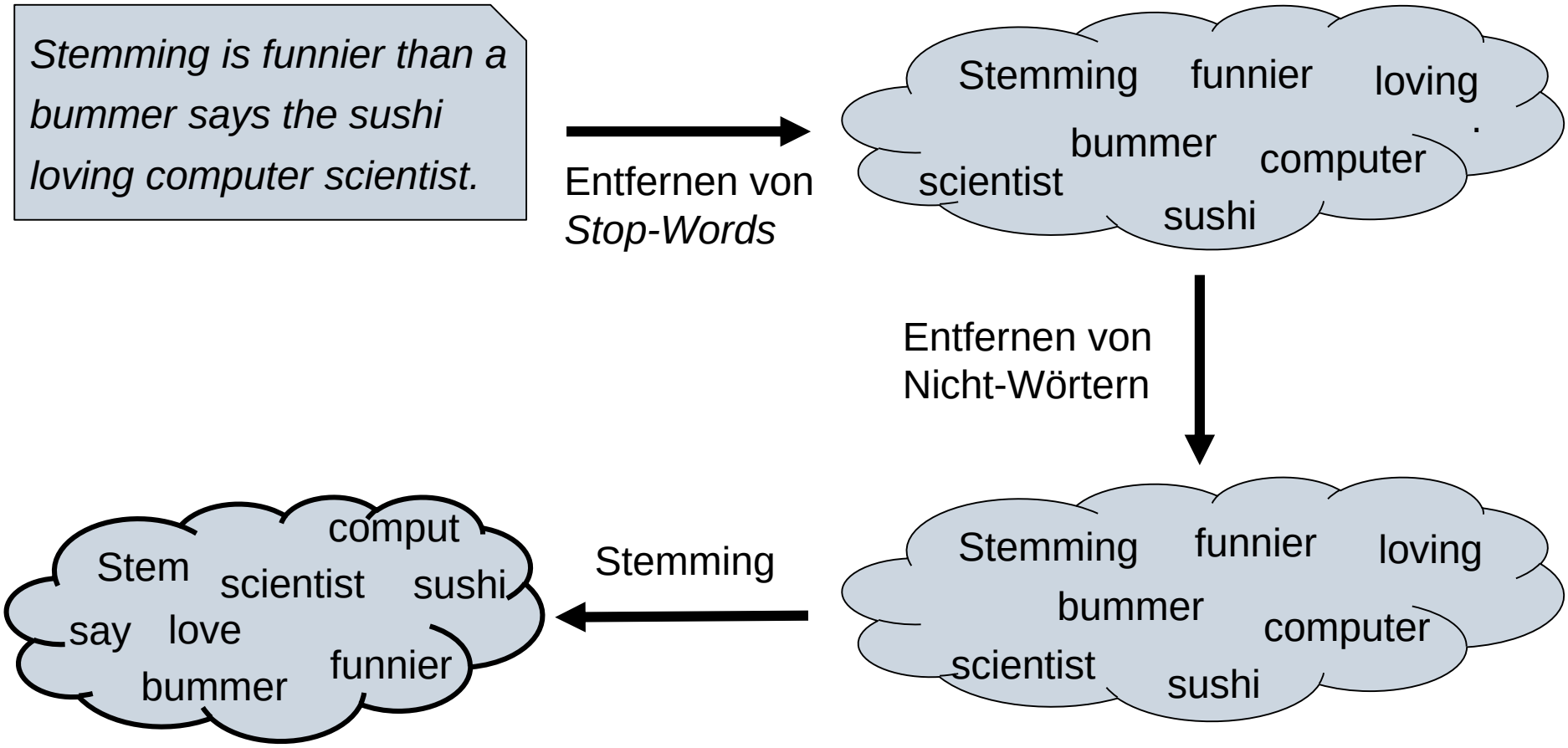
Ausgangssituation: Dokumente (Lyrics)

Grundidee:

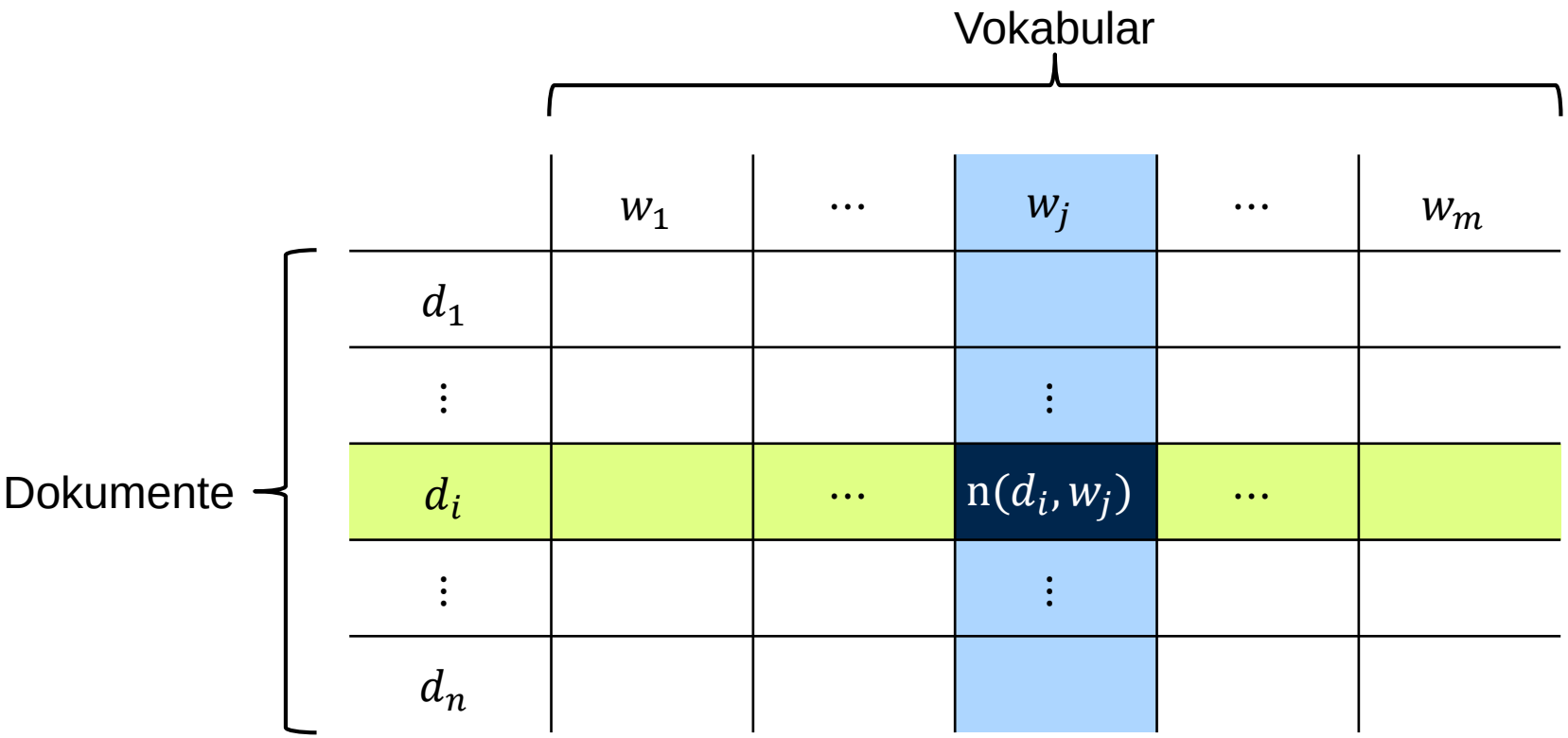
- Abbilden der Dokumente in einen Vektorraum → *latent semantic space*
- Mapping der Dokumente in den *latent semantic space* durch SVD
  - Optimal: Wörter mit einer gemeinsamen Bedeutung werden ungefähr in gleiche Richtung abgebildet
- Ähnlichkeitsmetrik: Winkel zwischen Vektoren oder Skalarprodukt

Vorverarbeitung der Dokumente

Umwandlung der Dokumente zu einem *Bag of Words*:



Document-Term Matrix



Singular Value Decomposition (SVD)

$A^{\{m,n\}}$

Matrix mit r Eigenwerten

$U^{T\{r,n\}}$

Matrix, die Zusammenhang zwischen Spaltenvektoren und Singulärwert darstellt

$$\boldsymbol{A} = \boldsymbol{S} \cdot \sum \cdot \boldsymbol{U}^T$$

$S^{\{m,r\}}$

Matrix, die Zusammenhang zwischen Zeilenvektoren und Singulärwerten darstellt

$\sum^{\{r,r\}}$

Diagonalmatrix mit r Singulärwerten auf der Diagonalen (absteigend)

Singulärwerte entsprechen den Wurzeln der Eigenvektoren von $A^T A$.

LSA mit SVD

Idee: Finde k besondere Merkmale in der Doc-Term Matrix (m Terms, n Docs).

Mathematisch: Reduzierung des Ranges der Matrix, dadurch Konzentrierung auf hervorstechende Merkmale.

- Führe **SVD auf Document-Term Matrix** aus.
- **Behalte die größten k Singulärwerte** der Matrix $\sum$
- $S \cdot \sum$**:** Zeilenvektoren um jedes Wort zu charakterisieren.
- $\sum \cdot U^T$**:** Spaltenvektoren um das Dokument zu charakterisieren.
- **Klassifiziere über Vektorabstände dieser Vektoren**.

Querying / Classification

Durch LSA werden Vektoren gewonnen, die Wörter und Dokumente klassifizieren.

⇒ Wie kann ich die Ähnlichkeit dieser Vektoren bestimmen/ausdrücken?

- Winkel θ zwischen zwei Vektoren u, v: (Kosinusähnlichkeit)

$$\theta = \frac{< u, v >}{\|u\| \cdot \|v\|}$$

Zusammenfassung

- **LSA kann das Problem lösen**
- Basiert auf **Singular Value Decomposition** (SVD) von **Document-Term Matrix**
- Relativ **simpel anzuwenden, aber hoher Rechenaufwand** ($O(n^2k^3)$)
- Das **Problem der Polyseme** (Mehrdeutigkeit) wird **nur teilweise gelöst**
- **Dimensionsproblem**: Wie soll k gewählt werden?

PROBABLISTIC LSA (PLSA)

Probabilistic Latent Semantic Analysis (PLSA)

- Ziel: **Identifizieren und Unterscheiden von verschiedenen Kontexten** von auftretenden Wörtern **ohne Thesaurus oder Wörterbuch**
  - Löst das Problem der Polyseme und Synonyme

- Im Gegensatz zu LSA liegt ein statistisches Modell zugrunde → *Aspect Model*
  - *latent variable model* für *co-occurence data (hier:* (w_j, d_i))
  - Weißt jeder Beobachtung eine **unbeobachtete Variable** $z_k \in \{z_1, \dots, z_K\}$ zu

- → **Ein Dokument kann mehrere Topics beinhalten und jedes Topic hat eigene charakteristische Wortverteilung**

Thomas Hofmann

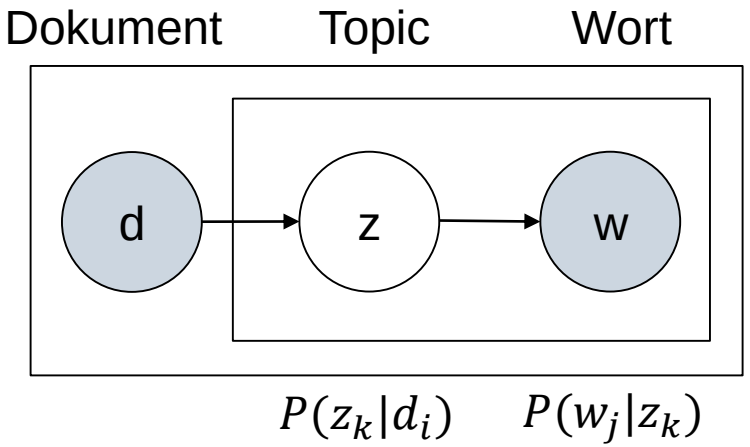
Exkurs: Grundbegriffe der Wahrscheinlichkeit

Seien a, b Zufallsvariablen, dann gelten folgende Eigenschaften der Wahrscheinlichkeitsfunktion p:

- **Bedingte Wahrscheinlichkeit:** $p(a|b) = \frac{p(a,b)}{p(b)}$
- **Satz von Bayes:** $p(a|b) = \frac{p(b|a)p(a)}{p(b)}$

PLSA

Basis bildet generatives Modell:



bekannt

$$P(d_i, w_j) = P(d_i)P(w_j|d_i)$$

$$P(w_j|d_i) = \sum_{k=1}^{K} P(w_j|z_k)P(z_k|d_i)$$

unbekannt

Likelihood

- **Ausgangssituation**: Ereignis ist eingetreten → Rückschlüsse ziehen auf zugrunde liegende Parameter
  - hier: $P(w_j|z_k)$ und $P(z_k|d_i)$

- **Ziel**: Approximation von Werten für Parameter, die am wahrscheinlichsten zum Eintreten des Ereignisses geführt haben

- *Likelihood function*:

$$L = \prod_{i=1}^{N} \prod_{j=0}^{M} P(d_i, w_j)^{n(d_i, w_j)}$$

bzw. als $\mathcal{L} = \log L$:

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{j=1}^{M} n(d_i, w_j) \log P(d_i, w_j)$$

EM-Algorithmus

- Maximiere die Likelihood Funktion $\mathcal{L}$.
- Finde eine Zuordnung der Parameter für bestmögliche Wahrscheinlichkeitsverteilung (model fitting)
- Problem: Topics z_i sind latent/versteckt ⇒ es fehlen Daten!
- Daher versuche durch wiederholte Ausführung von zwei Schritten zu schätzen:
  - Expectation Step
  - Maximization Step
- Gehe dabei von einer zufällig gewählten Verteilung aus und verbessere diese

Expectation Step

- Bestimme aus den derzeitig geschätzten Parametern die erwartete Likelihood unter Berücksichtigung der latenten Variablen:

$$P(z_k|d_i, w_j) = \frac{P((d_i, w_j)|z_k) \cdot P(z_k)}{P(w_j, d_i)}$$

Satz von Bayes

$$= \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_{l=1}^{K} P(w_j|z_l)P(z_l|d_i)}$$

Maximization Step

- Passe nun die Parameter so an, dass der Erwartungswert von $\mathcal{L}$ maximiert wird:

$$P(w_i|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j) P(z_k|d_i, w_j)}{\sum_{m=1}^{N} \sum_{i=1}^{N} n(d_i, w_m) P(z_k|d_i, w_m)}$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j) P(z_k|d_i, w_j)}{n(d_i)}$$

Was bekommen wir raus?

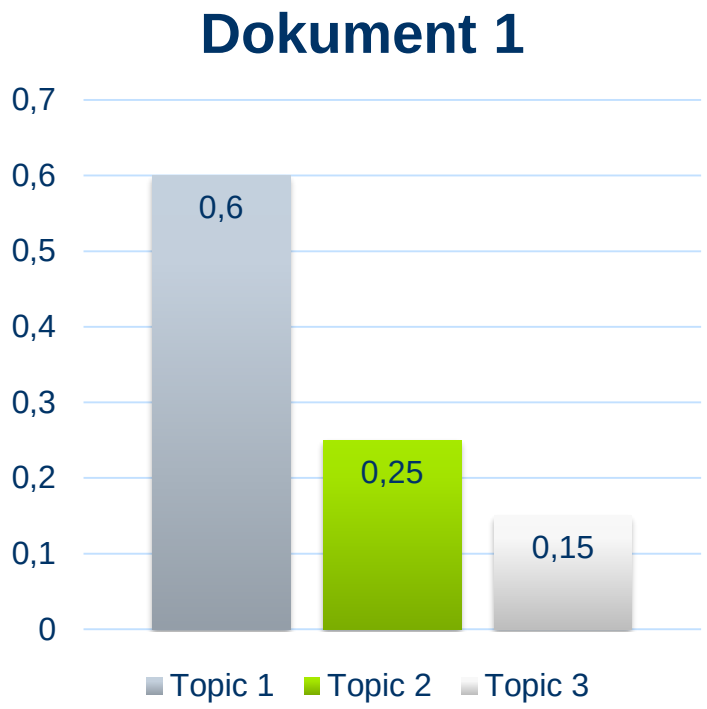
- Ergebnis von PLSA: Matrizen $P(w_j|z_k)$ und $P(z_k|d_i)$

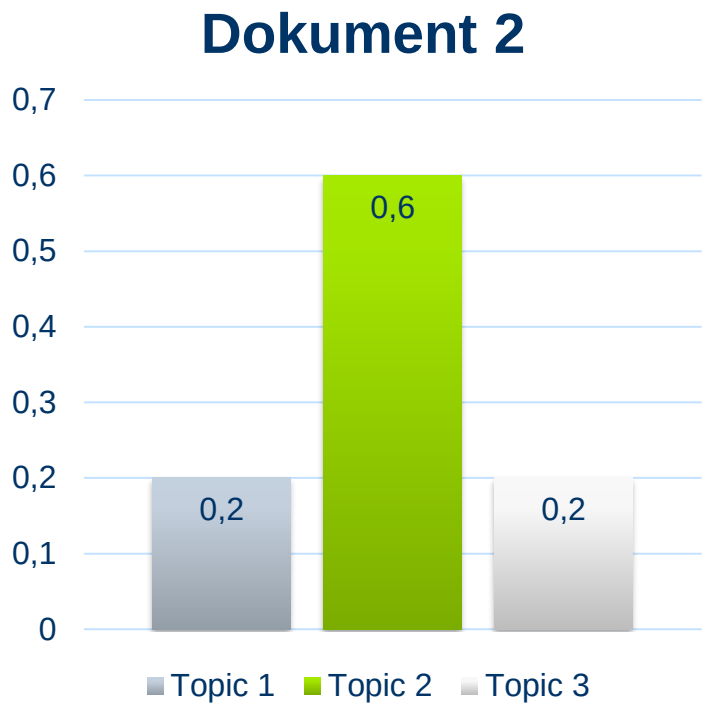
- $\boldsymbol{P(w_j|z_k)}$**: Zuordnung von Wörtern zu Topics**

Aspect 1	Aspect 2	Aspect 3	Aspect 4
plane	space	home	film
airport	shuttle	family	movie
crash	mission	like	music
flight	astronauts	love	new
safety	launch	kids	best
aircraft	station	mother	hollywood
air	crew	life	love
passenger	nasa	happy	actor
board	satellite	friends	entertainment
airline	earth	cnn	star

Was bekommen wir raus?

- $P(z_k|d_i)$: **Zuordnung von Topics zu Dokumenten**

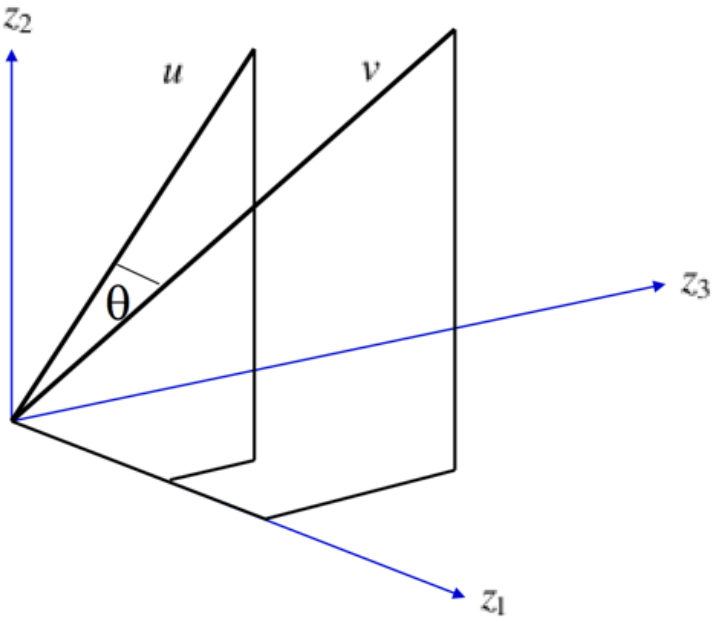




Classification

Welche Strategien gibt es um Dokumente zu Klassifizieren?

- **Interpretation** von den gewonnenen Werten $P(z_i|d)$ **als Vektor** (VSM)
  - v= $[P(z_1|d_1), P(z_2|d_1), P(z_3|d_1)]$
  - u= $[P(z_1|d_2), P(z_2|d_2), P(z_3|d_2)]$



- **Ähnlichkeit der Vektoren** bestimmen:
  - SSD: $\sum_{i=0}^{k}(u_i - v_i)^2$
  - Kosinusähnlichkeit: $\theta = \frac{<u,v>}{\|u\| \cdot \|v\|}$
  - Kullback-Leibler-Divergenz: $\sum_{i=0}^{k} u_i \cdot log \frac{u_i}{v_i}$

Finden von ähnlichen Dokumenten

- Begreifen der Topics als Vektorraum

- Gegeben ein gewünschtes Dokument, zu dem ähnliche Dokumente gesucht werden sollen

  1. Vergleichen der Topic-Vektoren der Dokumenten durch gewähltes Ähnlichkeitsmaß (→ Score)

  2. Rückgabe der n Lieder mit dem höchsten Score

Zusammenfassung

- Nutzt **Document-Term Matrix** als Informationsbasis
- Ist **robust gegenüber Synonymen und Polysemen**
- Basiert auf **maschinellem Lernen** und **Wahrscheinlichkeitsmodellen**
- Ist ein **rechenaufwendiges Verfahren**
- **Ergebnis sind zwei Matrizen**, $P(w_j|z_k)$ und $P(z_k|d_i)$
- **Klassifikation z.B. durch Ähnlichkeit von Vektoren**

Ist PLSA denn überhaupt anwendbar?

Reggae	Country	Newage	Rap	Rock
girl	love	adis	I‘m	I‘m
lover	I‘m	go	like	love
know	just	say	get	don‘t
love	don‘t	day	got	know
I‘m	know	night	don‘t	just
let‘s	like	love	n****	like
mi	got	sky	know	got
shout	time	says	s***	you‘re
Like	heart	ergo	ain‘t	time
gal	go	heart	yo	oh

Meiste Wörter pro Genre (ausgezählt)

Aspect 1	Aspect 2	Aspect 3	Aspect 4	Aspect 5
don‘t	love	blue	n****	que
feel	heart	beauti	s***	var
hate	feel	sun	ya	de
insid	god	oh	f***	la
god	sky	love	yo	y

Am häufigsten vorkommende Wörter in typischen Topics

IMPLEMENTIERUNG

Crawling

- Problem: **Aufbau des Korpus** (Beschaffung der Lyrics)
  - B. LOGAN et al. (2004): Crawling von *azlyrics.com*

- Extrahieren der Lyrics und Crawlen der Seite heute noch recht einfach
  - Problem: **Seite blockiert** schon nach wenigen Abrufen
  - Bei anderen frei verfügbaren Anbietern ähnliche Situation

- Gründe: Starke Konkurrenz zwischen Anbietern, lizenzrechtlich schwierig

- Alternative: kostenpflichtige Dienste (*musixmatch* etc.)
  - Wenige bieten kostenlose API an
  - Wenn dann nur stark limitiert

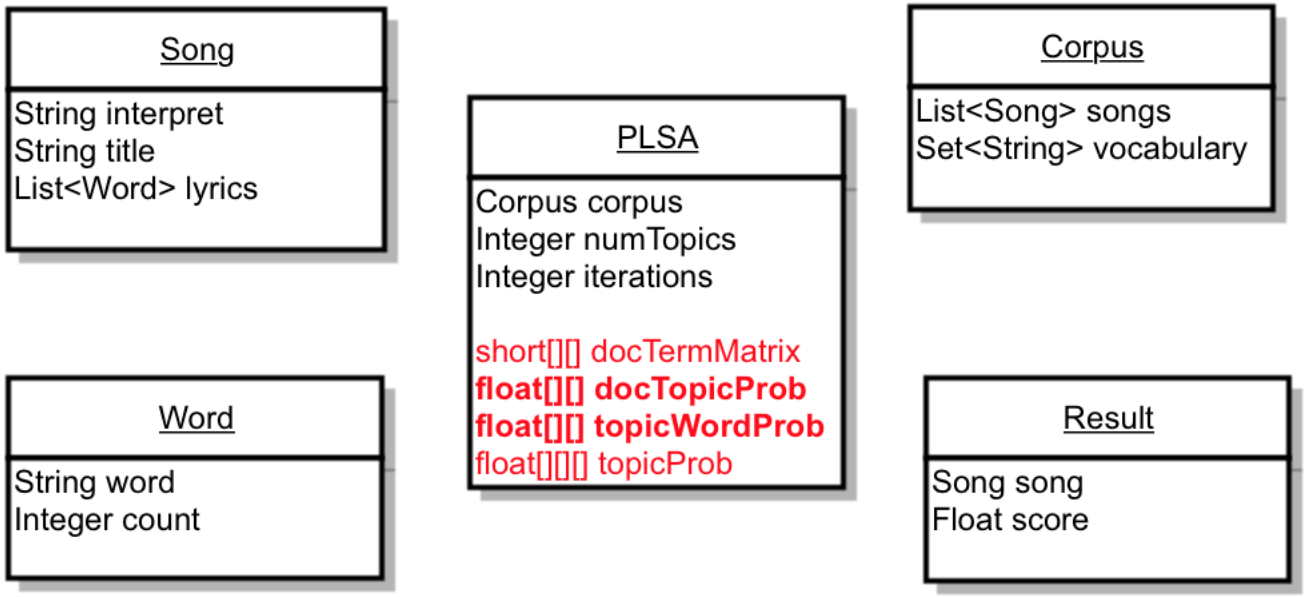
Million Songs / musixmatch Dataset

- 2011: Zusammenstellung des **Million Songs Datasets**
  - frei erhältliche Sammlung von Audio Features und Metadaten populärer Songs
- Abgleich des MSD mit musixmatch → *musixmatch Dataset*
- Enthält *Bag-of-Words* der Lieder bestehend aus den Top 5000 Wörtern insgesamt
  - Nachträgliche Entfernung von ca. 300 *Stop-Words* aus Wortliste nötig
- Dataset enthält **237.662** Songs

---

musiXmatch dataset, the official lyrics collection for the Million Song Dataset, available at: http://labrosa.ee.columbia.edu/millionsong/musixmatch

Datenstrukturen und resultierende Probleme



Parameter des PLSA Algorithmus

- **Anzahl der Dokumente im Korpus**
  - in der Implementierung: **max. 10000**
- **Anzahl der Topics**
  - In der Implementierung: **max. 20**
- **Stopkriterium für den EM-Algorithmus**
  - Konvergenz?
  - **Iterationen** → wenn ja, wie viele? Overfitting vermeiden!
    - Hier: **ca. 15**

DEMO

Ergebnisse

PLSA funktioniert

- Vorteil
  - Liefert **ähnliche Lieder (Dokumente)** basierend auf **Inhalt**
  - Unterschiedliche Sprachen werden zuverlässig erkannt

- Nachteil
  - Sehr **rechenaufwändig**
  - Viele falsche Ergebnisse

Weitere Ergebnisse

- Metriken: ARR (**Average Response Rank**) und FPA (**First Place Agreement**)

Technik	Topics	Stemming	ARR/FPA
PLSA	8 16 32	X X X	5,05/0,15 4,99/0,15 4,94/0,16
PLSA	8 16 32	✓ ✓ ✓	4,98/0,15 5,02/0,14 4,91/0,17
PLSA (iterated)	8 16 32	✓ ✓ ✓	4,82/0,18 4,89/0,16 4,95/0,17
Optimal	-	-	2,13/0,53
Zufällig	-	-	5,50/0,11

Adaptiert aus B. LOGAN et al. (2004)

Vergleich mit anderem Verfahren

- Verfahren von B. LOGAN und A. SALOMON:
  1. Audiodatei in **25ms Frames** aufteilen
  2. Für jeden Frame: **20 Mel-frequency cepstral features** berechnen
  3. Clustering mit K-Means Algorithmus → **Signatur für jeden Künstler**
  4. Ähnlichkeit wird über die *Earth Movers Distance* zwischen Clustern bestimmt

Genre	Antworten	Beide falsch	Lyrics richtig Acoustic falsch	Lyrics falsch Acoustic richtig
Rock	9513	6174 (65%)	1220 (13%)	1627 (17%)
Rap	316	176 (57%)	48 (15%)	51 (16%)
Reggae	227	176 (**78%**)	9 (4%)	39 (17%)
Country	225	127 (56%)	32 (14%)	44 (20%)
Latin	201	74 (37%)	48 (**24%**)	20 (**10%**)
Electronica	311	224 (**72%**)	9 (3%)	72 (**23**%)

Adaptiert aus B. LOGAN et al. (2004)

Ausblick: Latent Dirichlet Allocation (LDA)

- PLSA hat ebenfalls Nachteile:
  - Kein generatives Modell für Topiczuweisungen für Dokumente
    → **Klassifizierung schwierig für neue, unbekannte Dokumente**
  - Anzahl der Parameter wächst linear mit der Anzahl von Trainingsdokumenten
- Besser: **Latent Dirichlet Allocation (LDA)**
  - Erweiterung von PLSA
  - Anderes zugrundeliegendes generatives Model
  - Wesentlich komplexer als PLSA

Fazit

- Es gibt **mehrere Ansätze** das Problem zu lösen
- Alle gezeigten Algorithmen sind **sehr rechenaufwendig**
- Qualität der Ergebnisse: **über 50% Falschbestimmungen**
- Es **wird noch viel geforscht**, besonders in Hinblick auf
  - die Verbesserung der Wahrscheinlichkeitsmodelle (siehe LDA)
  - das maschinelle Lernen an sich
- Das Thema ist **hochgradig relevant und aktuell**

Literatur

T. Hofmann, „Probabilistic Latent Semantic Analysis“, *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, pp. 289–296, 1999.

B. Logan, A. Kositsky and P. Moreno, “Semantic Analysis of Song Lyrics”, *Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on, Vol. 2*, 2004. DOI: 10.1109/ICME.2004.1394328

T. Hofmann, “Unsupervised Learning by Probabilistic Latent Semantic Analysis”, *Machine Learning*, Vol. 42, pp. 177–196, 2001.

A. P. Dempster, N. M. Laird and D. B. Rubin, „Maximum Likelihood from Incomplete Data via the EM Algorithm“, *Journal of the Royal Statistical Society. Series B (Methodological)*, Vol. 39, pp. 1-38, 1977.

R. Lienhart, „Probabilistic Latent Semantic Analysis (pLSA)“, http://www.multimedia-computing.de/mediawiki/images/3/3d/SS08_BN-Lec11-pLSA.pdf, 2008. Last visited: 26.10.2016

Literatur

Mausam et al., „Document Similarity in Information Retrieval“, https://courses.cs.washington.edu/courses/cse573/12sp/lectures/17-ir.pdf, 2012. Last visited: 26.10.2016

T. Hofmann, „Latent Semantic Variable Models“, http://videolectures.net/slsfs05_hofmann_lsvm/, 2005. Last visited: 26.10.2016

T. Bertin-Mahieux, D. P.W. Ellis, B. Whitman and P. Lamere, „The Million Song Dataset“, *Proceedings of the 12th International Conference on Music Information*, 2011.

D. Blei, A. Ng and M. Jordan, „Latent Dirichlet Allocation“, *Journal of Machine Learning Research, Vol. 3*, pp. 993-1022, 2003.

Bildnachweise

- http://www.abeautifulhell.com/wp-content/uploads/2015/10/lyrics-1024x576.jpg
- https://www.irt.uni-hannover.de/fileadmin/_migrated/pics/ml.png
- http://mlss.tuebingen.mpg.de/2013/portrait_hofmann.jpeg
- http://cdn2.spiegel.de/images/image-8355-galleryV9-meml-8355.jpg
- http://cbsnews2.cbsistatic.com/hub/i/r/2014/11/29/3f587472-0211-4fe9-bd2f-54915c8d5618/resize/620x465/de19c981926774c5aadbd076b88b69a4/beatles-lyrics-good-day-sunshine-465.jpg