UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

# Betrachtung eines statischen Topic-Modells zur Repräsentation wechselnder Dokumente

*Examination of a Static Topic Model Representing Varying Documents*

**Bachelorarbeit**

im Rahmen des Studiengangs
**Informatik**
der Universität zu Lübeck

vorgelegt von
**Magnus Bender**

ausgegeben und betreut von
**Prof. Dr. Ralf Möller**

mit Unterstützung von
Felix Kuhr, Tanya Braun und Prof. Dr. Karsten Keller

Lübeck, den 1. Oktober 2019

**Kurzfassung** In der heutigen Zeit existieren große Mengen an Textdokumenten, die zu einem Korpus zusammengefasst automatisch analysiert werden sollen. Verfahren zur Dimensionsreduktion bieten eine Möglichkeit, die Textdokumente zu vergleichen und zu gruppieren. Wir nehmen an, dass stetig neue Dokumente dazukommen und somit ein zu analysierender Korpus nie vollständig ist.

Ein bekanntes Verfahren zur Topic-Modellierung ist Latent Dirichlet Allocation (LDA) [BNJ03]. LDA führt eine Dimensionsreduktion eines Korpus auf Topics, bestehend aus Topicverteilungen, durch. LDA lernt eine Repräsentation, d. h. ein Modell, für einen statischen Korpus. Um auch neu dazukommende Dokumente analysieren zu können, ist es in der Regel ratsam das gelernte Modell zu erneuern. Die Neuberechnung eines Modells ist jedoch sehr rechenintensiv. Gerade bei nur wenigen neuen Dokumenten kann es sinnvoll sein, das vorhandene Topic Modell nur an die neuen Dokumente anzupassen.

In dieser Bachelorarbeit evaluieren wir zwei unterschiedliche Verfahren, die beide zum Ziel haben, Modelle an neue Dokumente anzupassen. Das erste Verfahren ist Fold In Gibbs Sampling (FIGS) [GG84]. Mit FIGS können wir aus den gelernten Topicverteilungen eines initial gelernten Modells die Topicverteilung von neuen, bisher ungesehenen Dokumenten approximieren.
Das zweite Verfahren ist Online LDA [HBB10]. Online LDA integriert die bisher ungesehenen Dokumente in das vorhandene Modell, im Vergleich zu LDA jedoch ohne über den gesamten Korpus zu iterieren. Online LDA erzeugt somit einen deutlich geringeren Rechenaufwand.

Wir lernen zu Anfang für einen Korpus ein Modell mittels LDA. Anschließend erweitern wir den Korpus um neue Dokumente und bestimmen mit FIGS und Online LDA die Topicverteilungen für den erweiterten Korpus. Wir vergleichen die erweiterten Modelle mit einem durch LDA für alle Dokumente zusammen gelernten Modell hinsichtlich der Klassifikationsleistung, der Genauigkeit und der Perplexität.

Von den hier untersuchten Verfahren stellt FIGS eine schnelle und real nutzbare Alternative zur Neuberechnung eines Modells dar. Soll ein Korpus um eine größere Menge an Dokumenten erweitert werden, ist Online LDA ein geeignetes Verfahren.

**Abstract** Today, there are a large number of text documents. Combined into one corpus, these text documents are supposed to be analyzed automatically. Techniques for dimension reduction provide the possibility to compare and group text documents. We assume that new documents are constantly being added and therefore a corpus which has to be analyzed is never complete.

Latent Dirichlet Allocation (LDA) [BNJ03] is a well-known method used for topic modelling. LDA performs a dimension reduction of a corpus into topics, represented by topic distributions. However, LDA learns a representation, i.e. a model, for a static corpus. To analyze newly added documents, it is usually preferable to renew the learned model. But recalculating a model is computationally very intensive. Especially if adding only a few new documents, it can be useful to adapt the existing model to the corpus containing new documents.

In this thesis, we evaluate two different methods adapting topic models to new documents. The first method, Fold In Gibbs Sampling (FIGS) [GG84], approximates the topic distribution of a new, previously unseen document using the topic distributions of an initially learned model.
The second method, Online LDA [HBB10], integrates the previously unseen documents into an existing model, but without iterating over the entire corpus as LDA would need to. Thus, Online LDA leads to a significantly lower computing effort.

At the beginning, we learn a model for a corpus using LDA. Then, we extend the corpus with new documents and determine the topic distributions for the extended corpus using FIGS and Online LDA. We compare the performance of the adapted models by examining the classification performance, the accuracy, and the perplexity of each model using a model learned by LDA as a baseline.

Among the methods examined here, FIGS represents a fast and practicable alternative to the recalculation of a model. If a corpus has to be extended by a larger number of documents, Online LDA is a suitable procedure.

# Liste der verwendeten Variablen und Notationen

Diese Liste dient dazu, einen Überblick über die in der Arbeit genutzten Variablen und Notationen zu verschaffen. Alle Variablen werden im Verlauf der Arbeit eingeführt und erklärt.

- Kapitel 2

$d$: Textdokument bestehend aus $N$ Worten $\{w_1, ..., w_N\}$

$\mathcal{D}$: Korpus bestehend aus $|\mathcal{D}|$ Textdokumenten

$\mathcal{M}$: Topic-Modell eines Korpus $\mathcal{D}$

$K$: Anzahl der Topics eines Topic-Modells $\mathcal{M}$

$\theta_d$: Dokument-Topic-Verteilung für Dokument $d$

$\varphi_k$: Topic-Wort-Verteilung für Topic $k$

$\alpha$: Hyperparameter für LDA (wenige Topics pro Dokument)

$\beta$: Hyperparameter für LDA (wenige Worte pro Topic)

$z_{d,j}$: Topic des $j$-ten Wortes in Dokument $d$

$\text{LDA}(\mathcal{D})$: Topic-Modell mittels LDA für den Korpus $\mathcal{D}$ gelernt

$\text{OLDA}(\mathcal{M}, \mathcal{D}')$: Topic-Modell $\mathcal{M}$ erweitert mit OLDA und dem Korpus $\mathcal{D}'$

- Kapitel 3

$\text{HD}(t, t')$: Hellinger-Distanz zwischen zwei diskreten Wahrscheinlichkeitsverteilungen $t, t'$

$m_k = k'$: Mapping der Topic $k$ auf die Topic $k'$ eines anderen Topic-Modells

$\overline{m}_{k'} = k$: Umkehr des Mappings $m_k$

$\Delta_m(\mathcal{M}, \mathcal{M}')$: Distanz zwischen zwei Topic-Modellen $\mathcal{M}, \mathcal{M}'$ bei Verwendung des Mappings $m$

$m_{\min}(\mathcal{M}, \mathcal{M}')$: Optimales oder minimales Mapping zwischen Topic-Modellen $\mathcal{M}, \mathcal{M}'$

$\mathcal{K}(\mathcal{M})$: Klassifikationsleistung eines Topic-Modells $\mathcal{M}$

$\mathcal{I}_0 \cup \mathcal{E} = \mathcal{D}$: Initialer Teil $\mathcal{I}_0$ vereinigt mit Held-Out-Teil $\mathcal{E}$ zum Korpus $\mathcal{D}$

$L$: Anzahl der Anpassungen/Erweiterungen des Korpus eines initial gelernten Topic-Modells

$e_1 \cup ... \cup e_L = \mathcal{E}$: Hinzuzufügende Korpora $e_i$ für jede Anpassung $i = 1, ..., L$

$\mathcal{M}_{i,\text{LDA}}$: Topic-Modell nach der $i$-ten Anpassung mit LDA für Korpus $\mathcal{I}_i$

$\mathcal{M}_{i,\text{OLDA}}$: Topic-Modell nach der $i$-ten Anpassung mit OLDA für Korpus $\mathcal{I}_i$

$\mathcal{M}_{i,\text{FIGS}}$: Topic-Modell nach der $i$-ten Anpassung mit FIGS für Korpus $\mathcal{I}_i$

- Kapitel 4

$(\mathcal{D}, K, \alpha, \beta)$: Konfiguration der vor einem Lernvorgang festzulegenden Werte

$i$: Laufende Nummer der Anpassung $i = 1, ..., L$

## Erklärung

Ich versichere an Eides statt, die vorliegende Arbeit selbstständig und nur unter Benutzung der angegebenen Quellen und Hilfsmittel angefertigt zu haben.

(Magnus Bender)
Lübeck, den 1. Oktober 2019

# Inhaltsverzeichnis

# 1. Einleitung

In der heutigen Zeit kommen täglich große Mengen an Textdokumenten auf Plattformen wie Twitter, Facebook und Instagram zusammen. Twittert jeder täglich aktive Benutzer pro Tag einen Tweet von 140 Zeichen Länge, so entstehen dabei 17.64 Gigabyte[1] Textdokumente pro Tag. Aber nicht nur auf Plattformen wie Twitter fallen täglich mehrere Gigabyte Textdokumente an. Auch in Unternehmen können zum Beispiel durch den E-Mail-Verkehr mehrere Gigabyte Text pro Tag entstehen. Nehmen wir an, die Menge der E-Mails stellt einen Korpus von Textdokumenten dar, der nützliche Zusammenhänge und Erkenntnisse für das Unternehmen enthält. Durch die Gruppierung nach Themen können die eingehenden E-Mails zum Beispiel thematisch sortiert direkt den passenden Sachbearbeitern zugeordnet werden. Auch können einem Twitter-Benutzer basierend auf seinen Tweets und Likes andere Nutzer mit ähnlichen Interessen vorgeschlagen werden.

Mit Hilfe von Verfahren zur Topic-Modellierung ist es möglich, Korpora auf verschiedene Arten zu analysieren. Im ersten Schritt findet eine Dimensionsreduktion der komplexen natürlichsprachlichen Texte auf die Zugehörigkeit zu Topics statt. So entsteht eine vereinfachte Repräsentation der Dokumente des Korpus – das Topic-Modell. Mit dem Topic-Modell ist es möglich, Dokumente anhand ihrer Themen zu gruppieren sowie Dokumente und Gruppierungen zu vergleichen.

Weit verbreitet zur Topic-Modellierung ist das Verfahren Latent Dirichlet Allocation (LDA) [BNJ03], da es ein einfaches und grundlegendes Verfahren für die Topic-Modellierung ist. LDA berechnet für einen statischen Korpus eine Topicverteilung. In der Realität sind die Korpora jedoch nicht statisch. Den ganzen Tag über kommen im Unternehmen neue E-Mails an und die Benutzer schreiben neue Tweets. Jedes der neuen Dokumente kann zusammen mit den bereits vorhandenen Dokumenten nützliche Erkenntnisse liefern und muss für die Analyse in das Topic-Modell integriert werden.

In dieser Arbeit geht es um die Fragestellung, wie ein bestehendes Topic-Modell genutzt werden kann, um die Topicverteilung für hinzukommende Dokumente zu bestimmen. Es gibt unterschiedliche Möglichkeiten diese Frage zu beantworten. Einerseits kann das Modell aus dem um die neuen Dokumente erweiterten Korpus neu

[1] 126 Millionen aktive Benutzer pro Tag · 140 ASCII-Zeichen: 17 640 000 000 Bytes; `https://www.statista.com/topics/737/twitter/`

gelernt werden. Diese Neuberechnung eines Modells ist jedoch sehr rechenintensiv. Andererseits kann das vorhandene Modell ohne Neuberechnung an den um die neuen Dokumente erweiterten Korpus angepasst werden, d. h. das Topic-Modell wird um die neuen Dokumente erweitert. Gerade bei nur wenigen neuen Dokumenten kann es sinnvoll sein, nur eine solche Anpassung an die neuen Dokumente vorzunehmen. Es gibt mehrere Verfahren, die eine Anpassung des Topic-Modells an veränderte Korpora erlauben. Zwei Verfahren werden wir in dieser Arbeit evaluieren. Neben der einzusparenden Rechenleistung wird die Qualität der resultierende Modelle anhand der Perplexität und der Klassifikationsleistung auf Dokumente untersucht. Die Perplexität ist ein Maß für die Verwirrtheit eines Modells. Die Klassifikationsleistung der erweiterten Modelle wird mit einem vollständig neu gelernten Modell verglichen.

Unsere Untersuchungen wollen wir mit einem einfachen und grundlegenden Verfahren durchführen, daher wählen wir LDA zur Berechnung unserer Topic-Modelle. LDA berechnet aus dem Korpus zwei diskrete Wahrscheinlichkeitsverteilungen, die zusammen das Topic-Modell darstellen. Die Dokument-Topic-Verteilung gibt für jedes Dokument die Wahrscheinlichkeit der Zugehörigkeit zu den Topics an. Die Topic-Wort-Verteilung charakterisiert die Topics durch ihre häufigen Worte.

Zur Erweiterung der mittels LDA gelernten Modelle untersuchen wir die Verfahren Online LDA (OLDA) [HBB10] und Fold In Gibbs Sampling (FIGS) [GG84]. FIGS nutzt nur die Topicverteilungen des initial gelernten Modells und versucht, auf Basis dieser Verteilungen die Topicverteilungen neuer Dokumente zu approximieren. FIGS benötigt keine weiteren Lernvorgänge, ändert aber die Topicverteilungen für neue Dokumente nicht. OLDA hingegen führt eine Änderung der Verteilungen des initial gelernten Topic-Modells durch. Es werden jedoch nur die neuen Dokumente betrachtet und die Topicverteilungen nur angepasst und nicht neu gelernt. OLDA ist daher rechenintensiver als FIGS, jedoch schneller als die Neuberechnung durch LDA.

Wir unterteilen diese Arbeit in die folgenden drei Kapitel:

**Grundlagen** Das erste Kapitel der Arbeit beginnt mit der Festlegung einiger Notationen. Anschließend führen wir Topic-Modelle ein und beschreiben das Verfahren LDA. Zum Schluss gehen wir auf die beiden betrachteten Verfahren OLDA und FIGS ein.

**Vergleich von Modellen** Im zweiten Teil dieser Arbeit werden Verfahren zur Simulation und Evaluation der veränderlichen Korpora und Modelle behandelt. Zuerst wird Bezug auf die Auswahl der LDA-Implementierung genommen. Außerdem beschreiben wir verschiedene Ansätze, Topic-Modelle zu vergleichen.

Die Schwierigkeit bei Vergleichen von Topic-Modellen besteht darin, dass die Berechnung eines Topic-Modells nicht deterministisch ist und somit zwei auf dem gleichen Korpus gelernte Modelle verschieden sind.

**Auswertung** In diesem Kapitel untersuchen wir die Perplexitäten sowie die Klassifikationsleistungen der resultierenden Modelle der verschiedenen Verfahren. Dabei setzen wir die Möglichkeiten zum Vergleich von Topic-Modellen und unser Wissen über LDA ein. Zusätzlich vergleichen wir die Laufzeiten der Verfahren. Die Modelle werden auf verschiedenen Korpora bestehend aus Teilmengen des 20 Newsgroups[2] Datensatzes gelernt. Der 20 Newsgroups Datensatz setzt sich aus dem E-Mail-Verkehr verschiedener öffentlicher Newsgruppen zusammen.

Es zeigt sich, dass OLDA eine bessere Klassifikationsleistung als FIGS erreicht, die Perplexitäten der mit OLDA angepassten Modelle jedoch schlechter werden. Gerade für wenige hinzugefügte Dokumente steht FIGS OLDA in der Klassifikationsleistung kaum nach und führt zu deutlich besseren Perplexitäten. Bei großen Mengen neuer Dokumente liefert OLDA neben einer guten Klassifikationsleistung auch eine gute Perplexität.

[2] `http://qwone.com/~jason/20Newsgroups/`

# 2. Grundlagen

In diesem Kapitel führen wir die für den weiteren Verlauf notwendigen Notationen ein und geben einen Überblick über Topic-Modelle, insbesondere LDA. Weiterhin betrachten wir Verfahren zur Anpassung eines mit LDA berechneten Topic-Modells.

## 2.1. Notationen

Wir definieren die folgenden Variablen und Funktionen, um Eingaben und Konfigurationen der betrachteten Modelle und Korpora zu formalisieren.

- Ein Textdokument $d$ ist eine Menge von $N$ Worten $\{w_1, ..., w_N\}$.
- Ein Korpus $\mathcal{D}$ ist eine Menge bestehend aus $|\mathcal{D}|$ Textdokumenten.
- Ein Topic-Modell $\mathcal{M}$ ist eine Repräsentation eines Korpus $\mathcal{D}$.
- $K$ ist die Anzahl der Topics eines Topic-Modells $\mathcal{M}$.
- Mit einem Apostroph gekennzeichnete Variablen beschreiben ein anderes Objekt des gleichen Types, z. B. sind $\mathcal{D}$ und $\mathcal{D}'$ zwei Korpora bestehend aus unterschiedlichen Textdokumenten.
- Für einen Korpus $\mathcal{D}$ beschreibt LDA($\mathcal{D}$) ein mit LDA gelerntes Topic-Modell.
- Für einen Korpus $\mathcal{D}'$ und das initiale Topic-Modell $\mathcal{M}$ beschreibt OLDA($\mathcal{M}, \mathcal{D}'$) ein mittels OLDA gelerntes Topic-Modell für $\mathcal{D} \cup \mathcal{D}'$.

## 2.2. Topic-Modelle

Topic-Modelle sind eine abstrakte Repräsentation der Themengebiete innerhalb eines Korpus. Die Dokumente des Korpus werden in mehrere Themen, sogenannte Topics, unterteilt. Dabei beschreibt eine Topic ein bestimmtes Themengebiet innerhalb des Korpus. Eine Topic ist eine Verteilung aller vorkommenden Worte im Korpus, belegt mit der Wahrscheinlichkeit für jedes Wort.

Während des Lernens eines Topic-Modells findet eine Dimensionsreduktion der Dokumente des Korpus statt, um die Komplexität der natürlichsprachlichen Texte zu reduzieren. Eine Folge der Dimensionsreduktion ist, dass die Texte leichter zu analysieren sind. Mit Hilfe der gelernten Topicverteilungen eines Topic-Modells können Dokumente bezüglich ihres Inhaltes verglichen werden. Für den Vergleich kommen Distanzmaße wie die Hellinger-Distanz [Hel09] oder die Kullback-Leibler-Divergenz [KL51] zum Einsatz. So kann z. B. der thematische Abstand zwischen Dokumenten bestimmt werden und es können ähnliche Themengebiete erkannt werden.

Topic-Modelle können dem unüberwachten Lernen zugeordnet werden. Neben der Vorverarbeitung der Dokumente, der Wahl eines geeigneten Wertes für die Anzahl der zu bestimmenden Topics $K$ und der Festlegung von Hyperparametern müssen keine weiteren Vorgaben gemacht werden.

Es gibt viele erweiterte Formen von klassischen Topic-Modellen. Dynamische Topic-Modelle (DTM) [BL06] sind eine thematisch zu dieser Arbeit passende erweiterte Form. DTMs können die Entwicklung von Topics über mehrere in Intervallen hinzugefügte Korpora darstellen. Meist werden DTMs zur Untersuchung von Korpora eingesetzt, die über einen zeitlichen Verlauf in mehreren Intervallen entstanden sind. Im ersten Intervall wird ein Topic-Modell gelernt. Kommen im nächsten Intervall nun weitere Dokumente hinzu, so werden sie nicht in das vorhandene Modell integriert. Es wird ein neues Modell gelernt, wobei die Beziehungen zwischen den Topics erhalten bleiben. Mit DTMs lassen sich einzelne Topics und deren Verlauf über die Zeit hinweg untersuchen. In dieser Arbeit beschränken wir uns auf einfache Formen von Topic-Modellen ohne Erweiterungen.

## 2.3. Latent Dirichlet Allocation

Ein bekanntes Verfahren zur Topic-Modellierung von Dokumenten ist LDA [BNJ03]. LDA nutzt einen Korpus bestehend aus Textdokumenten. Jedes Dokument wird als sogenannter Bag of Words interpretiert, d. h. jedes Dokument wird als eine Menge von Worten und deren Häufigkeiten dargestellt. Reihenfolgen und Zusammenhänge zwischen beieinander stehenden Worten werden nicht betrachtet, um die Komplexität zu reduzieren.

Ein LDA-Lernvorgang beginnt mit der Zusammenstellung der Dokumente, die den Korpus repräsentieren und einer sinnvollen Wahl für die Anzahl der Topics $K$. Ein gutes $K$ zu wählen, ist ohne Betrachtung aller Dokumente des Korpus schwierig. [ZCP+15] stellt verschiedene Strategien zu Bestimmung dar. Zu große $K$ führen zu sehr unscharfen und mit anderen Topics übereinanderliegenden Topics. Eine zu

geringe Anzahl an Topics führt zu sehr allgemeinen Topics mit geringer Trennschärfe zu anderen Topics. $K$ muss für jeden Anwendungsfall neu bestimmt werden.

Zusätzlich gibt es noch zwei Hyperparameter $\alpha, \beta \in \mathbb{R}^+$, deren Wahl die Gewichtung der folgenden zwei Ziele beeinflusst:

(i) Ordne die Worte eines jeden Dokumentes so wenigen Topics wie möglich zu ($\alpha$).

(ii) Wähle für jede Topic so wenige relevante Worte wie möglich ($\beta$).

Beide Ziele stehen in einer Austauschbeziehung. Ziel (i) kann erreicht werden, indem alle Worte eines Dokuments einer Topic zugeordnet werden. Dadurch wird jedoch Ziel (ii) verletzt. Sind hingegen in jeder Topic nur die relevanten Worte, so kann Ziel (i) für Dokumente mit vielen verschiedenen Worten nicht erreicht werden.

Sinnvolle Werte für die Hyperparameter sind $\alpha \in [0.01, 0.1]$ und $\beta \approx 0.01$ [Zha15]. Wir erreichen mit diesen Werten dünn besetzte Topicverteilungen. Wählen wir größere Werte, nähern sich die Topicverteilungen einer Gleichverteilung an und die Modellperplexität wird schlechter. Die vorgeschlagenen Werte sind durch Vergleich der Perplexitäten für verschiedene Werte der Hyperparameter bestimmt worden.

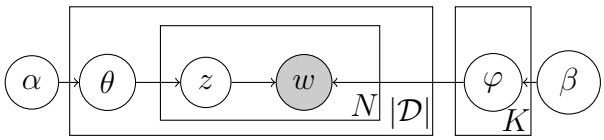


**Abbildung 2.1.:** Grafische Darstellung von LDA, nach [BNJ03]. Als einzige Variable kann $w_{d,j}$ beobachtet werden (grau dargestellt).

Ein mittels LDA gelerntes Topic-Modell besteht aus zwei gelernten diskreten Wahrscheinlichkeitsverteilungen. Beide Verteilungen werden aus den Dokumenten $d \in \mathcal{D}$ für die Topics $k \in \{1, ..., K\}$ gelernt.

- Die Dokument-Topic-Verteilung $\theta_d$ für jedes Dokument $d$ enthält die Wahrscheinlichkeit, mit der das Dokument $d$ zu jeder Topic $k$ gehört.
- Die Topic-Wort-Verteilung $\varphi_k$ für jede Topic $k$ gibt für jedes Wort aus $\mathcal{D}$ an, mit welcher Wahrscheinlichkeit das Wort zu Topic $k$ gehört.

### Algorithmus

Beschreibe Dir($\gamma$) eine Dirichletverteilung mit dem Parameter $\gamma$ und Multinom($\gamma'$) eine Multinomialverteilung über einen Versuch mit dem Parameter $\gamma'$. Dann folgt die Berechnung eines Topic-Modells, d. h. der beiden Wahrscheinlichkeitsverteilungen $\theta_d$ und $\varphi_k$, dem Schema:

(I) Wähle $\theta_d \sim \text{Dir}(\alpha)$

(II) Wähle $\varphi_k \sim \text{Dir}(\beta)$

(III) Für jede Wortposition $j = 1, ..., N$ im Dokument $d$:

(i) Wähle eine Topic $z_{d,j} \sim \text{Multinom}(\theta_d)$

(ii) Wähle ein Wort $w_{d,j} \sim \text{Multinom}(\varphi_{z_{d,j}})$

Die Schwierigkeit bei der Berechnung liegt in der Bestimmung der *a posteriori* Verteilung gegebener Dokumente für verborgene Variablen, also der Berechnung der folgenden Wahrscheinlichkeitsverteilung:

$$p(\theta, z \mid w, \alpha, \beta) = \frac{p(\theta, z, w \mid \alpha, \beta)}{p(w \mid \alpha, \beta)} \tag{2.1}$$

Eine exakte Berechnung der in Gleichung (2.1) dargestellten Verteilung ist nicht möglich. Allerdings eignen sich die folgenden Techniken zur approximativen Berechnung der Gleichung (2.1):

(i) Bayes-basierte Variationsrechnungen [BNJ03],

(ii) Gibbs Sampling [Gri02] sowie

(iii) Expectation-Propagation [ML02].

Im folgenden beschreibe $\mathcal{M} = \text{LDA}(\mathcal{D})$ ein mittels LDA auf dem Korpus $\mathcal{D}$ gelerntes Topic-Modell $\mathcal{M}$. Das Modell $\mathcal{M}$ besteht aus Matrizen mit den gelernten Werten für die Dokument-Topic-Verteilung $\theta$ und die Topic-Wort-Verteilung $\varphi$ sowie den Worten $w$ des Korpus und den zugehörigen Topics $z$ der Worte.

## 2.4. Korpuserweiterung

Es gibt verschiedene Ansätze, ein Topic-Modell an einen erweiterten Korpus anzupassen, ohne das Modell neu lernen zu müssen. OLDA passt das Modell an neue Dokumente durch eine geringe Anzahl Iterationen für jedes neue Dokument an. FIGS führt keine Iterationen oder Änderungen an den gelernten Verteilungen durch.

## Fold In Gibbs Sampling

FIGS [GG84] nutzt nur die gelernten Verteilungen des initialen Topic-Modells, ohne das initiale Modell durch die neuen Dokumente zu ändern. Somit bleibt die Topic-Wort-Verteilung unverändert. Die Topic-Wort-Verteilung wird für im initialen Korpus nicht vorhandene Worte auf null gesetzt. Die Dokument-Topic-Verteilung wird für die neuen Dokumente aus den vorhandenen Verteilungen des Topic-Modells approximiert.

FIGS ist somit ein einfacher Ansatz, um Approximationen der Dokument-Topic-Verteilung für neue Dokumente aus Topic-Modellen zu ziehen. Da keine Änderung der Modelle stattfindet, können die Approximationen für große Mengen neuer Dokumente jedoch sehr ungenau werden. Ein weiteres Problem liegt darin, dass die Aufteilung der Topics fix ist. Liegt ein neues Dokument in einem anderen Themengebiet als die Dokumente des anfänglichen Korpus des Modells, so wird keine sinnvolle Zuordnung der neuen Themen zu den bekannten Topics möglich sein.

## Online LDA

OLDA [HBB10] ist eine Weiterentwicklung des klassischen LDA. Es wird wie mit LDA ein Topic-Modell für einen Korpus errechnet. Jedoch kommt das Verfahren *Online variational Bayes for LDA* [HBB10] zur Berechnung von $p(\theta, z \mid w, \alpha, \beta)$ [Gleichung (2.1)] zum Einsatz.

*Online variational Bayes for LDA* zeichnet sich durch konstanten Speicherbedarf und eine schnelle garantierte Konvergenz aus, dabei werden ähnliche Perplexitäten wie mit klassischem LDA erreicht. Die Iteration über den hinzuzufügenden Korpus erfolgt in Form eines Datenstroms. Jedes Dokument muss nur einmal betrachtet werden. Die Änderungen am Topic-Modell folgen aus lokalen Optima für die gerade betrachteten Dokumente. Beim Erweitern eines Topic-Modells um Dokumente mittels OLDA muss somit nicht über den ganzen Korpus iteriert werden. Anders als bei FIGS finden jedoch Änderungen des initialen Topic-Modells statt.

OLDA kann auch zur Erweiterung eines leeren Topic-Modells genutzt und daher wie klassisches LDA eingesetzt werden. Aufgrund des konstanten Speicherbedarfs und der besseren Laufzeit implementieren viele Bibliotheken nur OLDA.

Sei $\mathcal{M} = \text{LDA}(\mathcal{D})$ ein Topic-Modell, dann beschreibe $\mathcal{M}' = \text{OLDA}(\mathcal{M}, \mathcal{D}')$ ein mittels OLDA um die Dokumente aus $\mathcal{D}'$ erweitertes Topic-Modell $\mathcal{M}'$.

Erweitern wir ein Topic-Modell in mehreren Iterationen um neue Dokumente, so gibt es zwei verschiedene Ansätze, die Anpassungen mit OLDA durchzuführen. Wir unterscheiden zwischen inkrementell gelernten Modellen und initial gelernten Modellen. Sei $\mathcal{M} = \mathrm{LDA}(\mathcal{D})$, bezeichne $\mathcal{D}', \mathcal{D}''$ die in der ersten und zweiten Iteration hinzuzufügenden Korpora sowie $\mathcal{M}', \mathcal{M}''$ die angepassten Modelle nach der ersten und zweiten Iteration.

**Inkrementelles OLDA** $\mathcal{M}' = \mathrm{OLDA}(\mathcal{M}, \mathcal{D}'), \quad \mathcal{M}'' = \mathrm{OLDA}(\mathcal{M}', \mathcal{D}'')$
Inkrementelles OLDA nutzt immer das zuletzt gelernte Modell, dadurch sind pro Iteration nur die neuen Dokumente hinzuzufügen.

**Initales OLDA** $\mathcal{M}' = \mathrm{OLDA}(\mathcal{M}, \mathcal{D}'), \quad \mathcal{M}'' = \mathrm{OLDA}(\mathcal{M}, \mathcal{D}' \cup \mathcal{D}'')$
Initiales OLDA erweitert immer ein bisher nicht erweitertes Topic-Modell. Somit wird die Menge der hinzuzufügenden Dokumente mit jeder Iteration größer.

# 3. Vergleich von Modellen

In diesem Kapitel beschreiben wir, wie Topic-Modelle gelernt und verglichen werden können. Dazu wählen wir im ersten Schritt eine LDA-Implementierung aus. Für die Evaluation definieren wir die Klassifikationsleistung eines Topic-Modells und entwickeln ein Verfahren zur Berechnung der Klassifikationsleistung. Weiterhin behandeln wir die Perplexität von Topic-Modellen.

## 3.1. Implementierungen

Wir wählen eine LDA-Implementierung aus, um die zu evaluierenden Topic-Modelle zu lernen und um weitere Dokumente erweitern zu können. Für die Wahl wurden folgende Kriterien festgelegt:

- Lernen eines Topic-Modells mittels LDA für einen Korpus unter Vorgabe von Hyperparametern und Anzahl der Topics
- Verwendung von OLDA auf einem initial gelernten Topic-Modell, sodass das Modell weitere Dokumente repräsentiert
- Approximieren der Dokument-Topic-Verteilung für ein unbekanntes Dokument unter Verwendung von FIGS
- Zugriff auf die Dokument-Topic- und die Topic-Wort-Verteilung eines Modells zur Berechnung der Klassifikationsleistung
- Bestimmung der Perplexität eines Modells, auch über Teile des Korpus
- Speicherung der gelernten Modelle in Dateien, damit mehrere Modelle gelernt und später verglichen werden können und damit die Möglichkeit besteht, Zwischenschritte während der Evaluation zu speichern
- Open Source und kostenfrei nutzbar
- Große Nutzerschaft, die viele Tutorials erstellt hat und die Möglichkeit bietet, bei Problemen Hilfestellungen zu finden

In die engere Wahl kamen die Implementierungen:

- *LDAPlusPlus*[1], C++
- *Gensim*[2], Python
- *Topic Models*[3], R

Wir haben alle drei Implementierungen getestet und wählen die *Gensim*-Implementierung von Radim Řehůřek [ŘS10]. *Gensim* erfüllt alle Kriterien und lässt sich als Paket in Python installieren sowie mit einfachen Python-Skripten konfigurieren und erweitern. Gegen *LDAPlusPlus* spricht, dass wir das Programm selbst kompilieren müssen und es keine offizielle Unterstützung für andere Betriebssysteme außer Linux gibt. Auf Python können wir auch nicht verzichten, denn *LDAPlusPlus* nutzt ein Dateiformat, welches mit Python weiterverarbeitet wird. Die Installation und Nutzung von *Gensim* gestaltet sich einfacher als die Nutzung von *Topic Models*, ohne dass *Topic Models* Vorteile gegenüber *Gensim* mitbringt.

## 3.2. Modellähnlichkeit

Für die Evaluation der verschiedenen Ansätze zur Anpassung von Topic-Modellen benötigen wir ein Verfahren, um die Ähnlichkeit zwischen zwei Modellen zu bestimmen. Wir legen fest, dass die Ähnlichkeit eines um Dokumente erweitertes Topic-Modells zu einem direkt mit allen Dokumenten gelernten Modell ein Maß für die Qualität des genutzten Ansatzes zur Anpassung ist. Weiterhin darf die Perplexität eines Topic-Models durch die Anpassung nicht schlechter werden.

### 3.2.1. Mapping

Wollen wir zwei Topic-Modelle vergleichen, müssen wir im ersten Schritt ein Mapping zwischen den Topics berechnen. Wir können nicht annehmen, dass Topic 1 in einem Modell die gleiche Topic wie Topic 1 in einem anderen Modell repräsentiert. Abbildung 3.1 zeigt den Unterschied der Dokument-Topic-Verteilung des gleichen Dokumentes aus zwei verschiedenen Modellen, die aus dem gleichen Korpus berechnet worden sind. Wir sehen Topic 1 im Modell 1 repräsentiert Topic 2 im Modell 2. Da LDA Techniken zur approximativen Berechnung nutzt, sind die Werte der Wahrscheinlichkeiten in zwei Modellen auch nicht gleich, sondern nur ähnlich. Ein gutes Mapping zeichnet sich durch geringe Abweichungen der Werte aus.

[1] `http://ldaplusplus.com/`
[2] `https://radimrehurek.com/gensim/`
[3] `https://cran.r-project.org/web/packages/topicmodels`

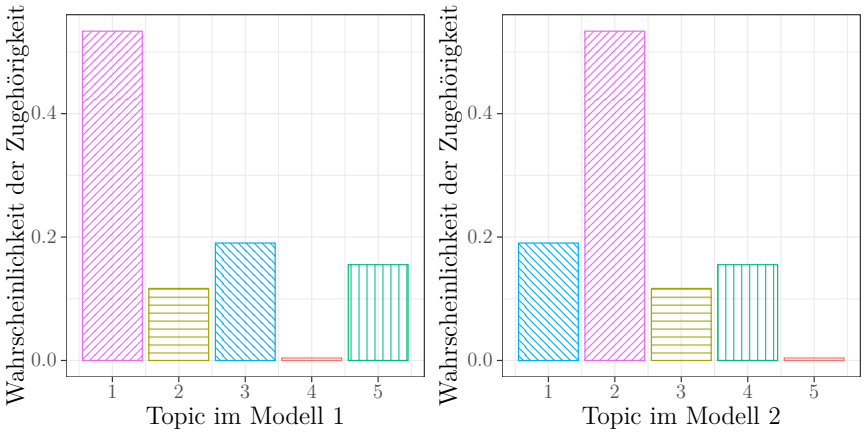


**Abbildung 3.1.:** Beispielhafte Darstellung der Dokument-Topic-Verteilung für das gleiche Dokument in zwei verschiedenen Topic-Modellen, die mit dem gleichen Korpus und gleichen Parametern gelernt wurden. Das optimale Mapping von Modell 1 auf Modell 2 ist hier $m = (2, 3, 1, 5, 4)$.

Für die Berechnung eines Mappings können nur die Differenz der Dokument-Topic-Verteilungen und der Topic-Wort-Verteilungen beider Modelle genutzt werden. Als Differenzfunktion bietet sich die Hellinger-Distanz (HD) [Hel09] an, da die HD oft im Kontext von Topicverteilungen genutzt wird. Außerdem bietet *Gensim* eine Schnittstelle, die HD zwischen zwei Modellen, genauer deren Topic-Wort-Verteilungen, als Hellinger-Distanzmatrix auszugeben. Eine Hellinger-Distanzmatrix gibt dabei für jede Topic des ersten Modells die Distanz zu jeder Topic des zweiten Modells an.

Im Folgenden gehen wir davon aus, dass unsere Modelle $K$ Topics haben.

**Definition 3.1.** Die Hellinger-Distanz (HD) ist für zwei diskrete Topicverteilungen $t = (t_1, \ldots, t_K)$ und $t' = (t'_1, \ldots, t'_K)$ definiert durch

$$\mathrm{HD}(t, t') = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{K} \left(\sqrt{t_i} - \sqrt{t'_i}\right)^2}. \tag{3.1}$$

**Definition 3.2.** Ein Mapping $m = (m_1, ..., m_K)$ ordnet jeder Topic $k \in \{1, ..., K\}$ eines Modells $\mathcal{M}$ eine Topic $k' \in \{1, ..., K\}$ eines anderen Modells $\mathcal{M}'$ zu.

$$m_k = k' \quad \text{und} \quad \overline{m}_{k'} = k \text{ ist seine Umkehr} \tag{3.2}$$

**Definition 3.3.** Gegeben sei ein Mapping $m$ und die Topic-Wort-Verteilungen $\varphi_k, \varphi'_k$, dann definieren wir die Distanz des Mappings von $\mathcal{M}$ zu $\mathcal{M}'$ als durchschnittliche Abweichung der Topics

$$\Delta_m(\mathcal{M}, \mathcal{M}') := \frac{\sum_{k=1}^{K} \mathrm{HD}(\varphi_k, \varphi'_{m_k})}{K}. \tag{3.3}$$

Um das optimale Mapping zu erhalten, müssen wir das nachfolgende Optimierungsproblem lösen.

**Definition 3.4.** Gegeben seien zwei Modelle $\mathcal{M}$ und $\mathcal{M}'$, dann ist das minimale Mapping definiert durch

$$m_{\min}(\mathcal{M}, \mathcal{M}') := \min \left\{ \Delta_m(\mathcal{M}, \mathcal{M}') \,\middle|\, \begin{array}{l} m \text{ ist ein Mapping} \\ \text{zwischen } \mathcal{M} \text{ und } \mathcal{M}' \end{array} \right\}. \tag{3.4}$$

Die minimale Distanz kennzeichnet das optimale Mapping.

### 3.2.2. Mapping bestimmen

Das Finden von optimalen Mappings ist nicht trivial und lässt sich unter realen Bedingungen nur näherungsweise durchführen. Folgende Ansätze werden betrachtet:

1. **Permutatives Mapping** Bei einer geringen Anzahl Topics ($K < 13$) ist es möglich, alle Permutationen für das Mapping zu durchlaufen und das Minimum nach Definition 3.4 zu bestimmen. Der Ansatz erreicht als einziger ein optimales Mapping, unter der Bedingung, dass wir von jeder und auf jede Topic nur einmal mappen.
   Für $K > 13$ ist dieses permutative Verfahren aufgrund der hohen Laufzeit $\mathcal{O}(K!)$ nicht mehr möglich. In Abbildung 3.3 zeigen wir rechts einen Boxplot der Werte für permutativ berechnete minimale Mappings. Links sehen wir zum Vergleich den Durchschnitt der Distanzen der Mappings über alle Permutationen und in der Mitte den dritten Ansatz.

2. **Minimale Hellinger-Distanz** Gegeben sei eine Hellinger-Distanzmatrix, die für jede Topic des ersten Modells die HD zu jeder Topic des zweiten Modells angibt. Jede Zeile stellt eine Topic aus $\mathcal{M}$ dar, jede Spalte eine Topic aus $\mathcal{M}'$. Wir mappen für jede Topic, also jede Zeile, auf die Spalte mit dem minimalen Wert in dieser Zeile. In Abbildung 3.2 sehen wir eine Hellinger-Distanzmatrix. Das Verfahren liefert sehr gute Ergebnisse bei einer Laufzeit von $\mathcal{O}(K^2)$, teilweise sind die Ergebnisse besser als mit einem permutativen Mapping (Ansatz 1), da wir auf eine Topic mehrfach mappen können. Es können jedoch qualitativ schlechte Mappings zwischen unähnlichen Topics entstehen.

   Ein qualitativ schlechtes Mapping zeigt sich, falls zu oft auf dieselbe Topic gemappt wird. Kommen in allen Dokumenten eines Korpus bestimmte Worte vor oder haben alle Topics eines Modells die gleichen häufigen Worte, so kann ein schlechtes Mapping entstehen. Wir sollten daher Mappings mit einem anderen Ansatz überprüfen. Das Mapping in Abbildung 3.2 ist ein schlechtes Mapping, da dreimal auf eine Topic gemappt wird.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.32 | 0.34 | 0.31 | 0.42 | 0.42 |
| 2 | 0.31 | 0.29 | 0.29 | 0.39 | 0.36 |
| 3 | 0.32 | 0.42 | 0.38 | 0.32 | 0.50 |
| 4 | 0.34 | 0.35 | 0.34 | 0.39 | 0.38 |
| 5 | 0.30 | 0.30 | 0.31 | 0.33 | 0.30 |

**Abbildung 3.2.:** Eine Hellinger-Distanzmatrix, mit der das Mapping über die minimale HD (Ansatz 2) bestimmt wird. Wir mappen für jede Zeile auf die Spalte mit dem kleinsten Wert. Das resultierende Mapping wäre hier $m = (3, 2, 1, 1, 1)$.

**Definition 3.5.** Gegeben seien zwei Mengen $A, B$, dann ist der Jaccard-Koeffizient [Jac01] definiert durch

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{3.5}$$

**3. Maximaler Jaccard-Koeffizient** Aus der Topic-Wort-Verteilungen lässt sich eine Menge der charakterisierenden Worte für jede Topic bestimmen. Die Menge kann zum Beispiel auf die wahrscheinlichsten 25 Worte für jede Topic beschränkt werden. Zur Bestimmung der Ähnlichkeit zweier Mengen bietet sich der Jaccard-Koeffizient an.
Ähnlich zur Hellinger-Distanzmatrix können wir nun eine Jaccard-Distanzmatrix bestimmen. Für gleiche Mengen ist der Jaccard-Koeffizient gleich 1, da Zähler und Nenner identisch sind. Adaptieren wir nun das Verfahren der minimalen HD (Ansatz 2), so wählen wir in jeder Zeile der Matrix die Spalte mit dem größten Wert.
Wie in dem zweiten Ansatz ist es möglich, doppelt auf eine Topic zu mappen. Außerdem ist das Mapping stark von der Anzahl der Worte in den zu vergleichenden Mengen abhängig. Da wir die Mengen aus den wahrscheinlichsten Worten für eine Topic erstellen, geht die Relevanz einzelner Worte verloren. Es wird immer eine feste Anzahl relevanter Worte gewählt, auch wenn weniger Worte eine Topic charakterisieren.

**4. Jaccard-Koeffizient der Dokument-Topic-Verteilung** In den ersten drei Ansätzen haben wir bei der Berechnung des Mappings nur die Topic-Wort-Verteilung genutzt, wir können jedoch auch die Dokument-Topic-Verteilung nutzen. Wie für den maximalen Jaccard-Koeffizienten (Ansatz 3) müssen wir zuerst die Dokument-Topic-Verteilung auf Mengen reduzieren, die wir mit dem Jaccard-Koeffizienten vergleichen können. Anders als in dem dritten Ansatz bestimmen wir hier für jede Topic die Menge der wahrscheinlichsten Dokumente und nicht die Menge der häufigsten Worte. Die Mengen für zwei Topic-Modelle sind in

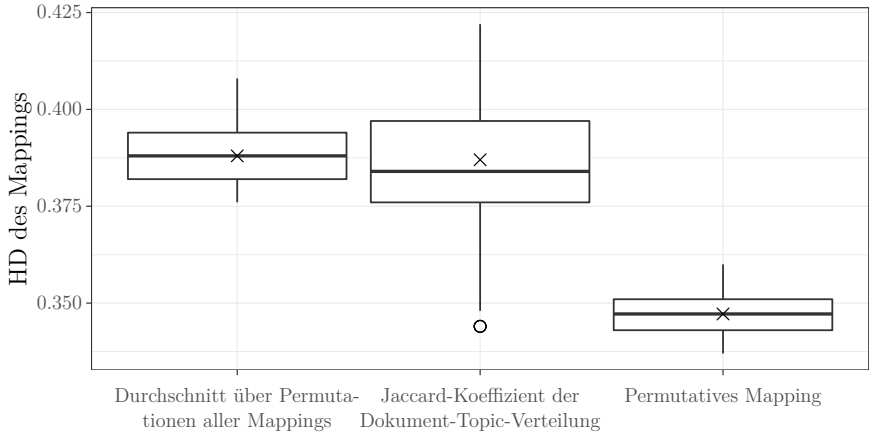


**Abbildung 3.3.:** Alle 45 Modelle wurden auf dem gleichen Korpus errechnet. Die Distanz des Mappings über den *Jaccard-Koeffizienten der Dokument-Topic-Verteilung* (Ansatz 4) streut stark und ist schlechter als das *permutative Mapping* (Ansatz 1). Links zum Vergleich der Durchschnitt der Werte über die Permutationen aller Mappings.

Abbildung 3.4 dargestellt. Wir können für die Menge jeder Topic aus Modell 1 auf die ähnlichste Menge aus Modell 2 mappen. Dabei bestimmen wir die Ähnlichkeit mit dem Jaccard-Koeffizienten.
Das Mapping können wir mit einem über den maximalen Jaccard-Koeffizienten bestimmten Mapping (Ansatz 3) kombinieren, indem wir jedes Mapping zweimal bestimmen. Zuerst wird das Mapping zwischen Model 1 und 2, danach umgekehrt zwischen Model 2 und 1 bestimmt. Anschließend führen wir einen Mehrheitsentscheid unter den 4 Mappings durch. In Abbildung 3.3 sehen wir in der Mitte, dass die Mappings aus dieser Kombination beider Ansätze stark streuen. Ein Grund dafür ist, dass die Relevanz einzelner Worte bei der Reduktion der Topicverteilungen auf Mengen verloren geht.

**5. Jaccard-Hellinger-Differenz** Nachdem wir die HD und den Jaccard-Koeffizienten genutzt haben, liegt die Idee nahe, beide Maße zu kombinieren. Da der Jaccard-Koeffizient eine 1 bei identischen Mengen und die HD eine 0 bei identisch Verteilungen annimmt, sollte eine große Differenz zwischen beiden Werten ein gutes Mapping kennzeichnen. Die so bestimmten Mappings sind aber deutlich schlechter als die permutativen Mappings (Ansatz 1) und die Mappings mit der minimalen HD (Ansatz 2).

$\mathcal{M}$ :

| $k$ | $d$ |
|---|---|
| 1 | $i, ii, vii$ |
| 2 | $iii, iv$ |
| 3 | $v, vi$ |

$\mathcal{M}'$ :

| $k$ | $d$ |
|---|---|
| 1 | $iii, iv$ |
| 2 | $i, ii, vii$ |
| 3 | $v, vi$ |

**Abbildung 3.4.:** Die Dokument-Topic-Verteilung von zwei Topic-Modellen $\mathcal{M}, \mathcal{M}'$ auf Mengen reduziert, um das Mapping über den Jaccard-Koeffizient der Dokument-Topic-Verteilung (Ansatz 4) zu bestimmen. Jedes der Dokumente $d \in \{i, ii, ..., vii\}$ wird der wahrscheinlichsten Topic $k \in \{1, ..., 3\}$ zugeordnet. Das Mapping wäre hier $m = (2, 1, 3)$.

Die verschiedenen Ansätze können auch zusammen genutzt werden. Falls während der Berechnung eines Mappings Zuordnungen nicht eindeutig sind, kann ein weiteres Mapping mit einem anderen Verfahren bestimmt werden. Auch können immer mehrere Verfahren genutzt werden. Z. B. ist die Bestimmung der Mengen für die Ansätze mit dem Jaccard-Koeffizienten nicht symmetrisch, daher sollte immer das Mapping zwischen Model 1 und 2 sowie umgekehrt zwischen Model 2 und 1 berechnet werden. Aus den resultierenden Mappings für jede Topic kann dann mittels eines Mehrheitsentscheides das endgültige Mapping bestimmt werden.

### 3.2.3. Gewähltes Mappingverfahren

In dieser Arbeit kommt als Verfahren zur Berechnung von Mappings ein Verbund aus mehreren Ansätzen zum Einsatz. Neben dem permutativen Mapping (Ansatz 1) wird das Mapping über die minimale HD (Ansatz 2) und über den maximalen Jaccard-Koeffizienten (Ansatz 3) berechnet.

Wir beginnen in Algorithmus 1 mit der Berechnung des Mappings über die Hellinger- und Jaccard-Distanzmatrizen nach Ansatz 2 und 3. Dabei berechnen wir jedes Mapping einmal von Modell 1 zu Modell 2 und einmal umgekehrt. Wir erhalten so 4 Mappings, innerhalb derer wir einen Mehrheitsentscheid durchführen. Für das aus dem Mehrheitsentscheid erhaltene Mapping bestimmen wir anschließend die Distanz.

Weiterhin berechnen wir solange $K < 13$ ist das permutative Mapping (Ansatz 1). Obwohl die Berechnung einen hohen Aufwand bedeutet, ist die Laufzeit für maximal 12 Topics auf unseren Servern ausreichend effizient und wir erhalten zuverlässig ein Mapping mit geringer Distanz und guter Qualität. Auch für das permutative Mapping bestimmen wir die Distanz.

Zum Schluss wählen wir von beiden berechneten Mappings das Mapping mit der geringeren Distanz als optimales Mapping. Bezeichne $m_{\min}(\mathcal{M}, \mathcal{M}')$ im Folgenden das in hier geschilderte Verfahren zur näherungsweisen Bestimmung von optimalen Mappings zwischen $\mathcal{M}$ und $\mathcal{M}$'.

**Algorithmus 1 :** Approximative Bestimmung eines optimalen Mappings.

**Daten** : Hellinger-Distanzmatrix $H_{k,k'}$ und Jaccard-Distanzmatrix $J_{k,k'}$ zwischen $\mathcal{M}$ und $\mathcal{M}'$ mit $k, k' = \{1, ..., K\}$ und $K = K'$

**Ergebnis** : approximatives optimales Mapping $m = m_{\min}(\mathcal{M}, \mathcal{M}')$

**Beginn**

$A = (a_{k,i})_{K,K}$

$\forall k, \forall i : a_{k,i} = 0$

**für** $k \in \{1, ..., K\}$ **tue**

$i = \texttt{minindex}_{j \in \{1,...,K\}}(H_{k,j})$ ▷ Mappe auf Spalte mit min. HD

$a_{k,i} = a_{k,i} + 1$

$i = \texttt{maxindex}_{j \in \{1,...,K\}}(J_{k,j})$ ▷ Mappe auf Spalte mit max. JK

$a_{k,i} = a_{k,i} + 1$

$i = \texttt{minindex}_{j \in \{1,...,K\}}(H_{j,k})$ ▷ Mappe auf Zeile mit min. HD

$a_{i,k} = a_{i,k} + 1$

$i = \texttt{maxindex}_{j \in \{1,...,K\}}(J_{j,k})$ ▷ Mappe auf Zeile mit max. JK

$a_{i,k} = a_{i,k} + 1$

**für** $k \in \{1, ..., K\}$ **tue**

$m_k = \texttt{maxindex}_{j \in \{1,...,K\}}(a_{k,j})$ ▷ Mehrheitsentscheid

$m' =$ berechnePermutativesMapping$(H, J)$

**wenn** $\Delta_m(\mathcal{M}, \mathcal{M}') > \Delta_{m'}(\mathcal{M}, \mathcal{M}')$ **dann**

$m = m'$ ▷ Besseres Mapping wählen

**Funktion** `maxindex(`$v$`)`
Gibt den ersten Index des größten Wertes des Vektors $v$ zurück.

**Funktion** `minindex(`$v$`)`
Gibt den ersten Index des kleinsten Wertes des Vektors $v$ zurück.

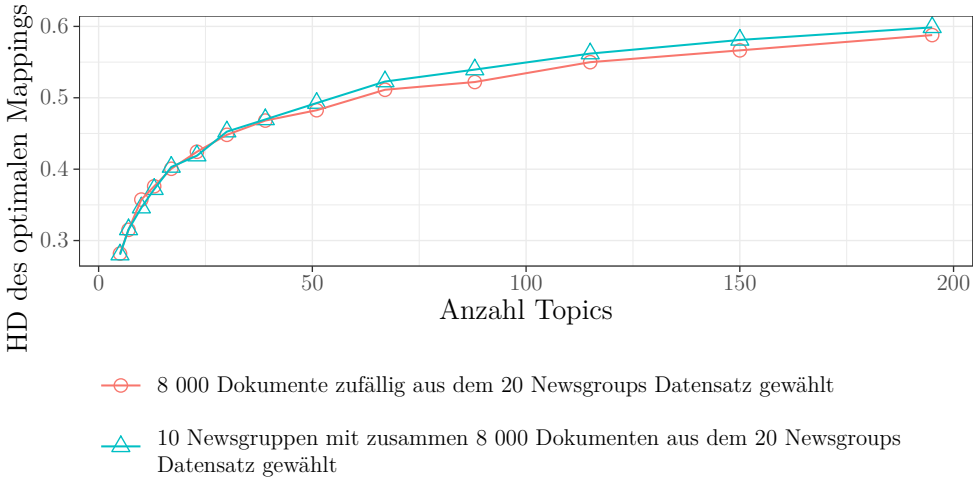


**Abbildung 3.5.:** Mit der Anzahl der Topics steigt auch der durchschnittliche Wert eines approximativ bestimmten optimalen Mappings für Topic-Modelle gleicher Korpora.

Abbildung 3.5 zeigt den Verlauf der Werte für ein optimales Mapping bei steigender Anzahl der Topics $K$. Die untersuchten Modelle wurden für dem gleichen Korpus gelernt. Wir stellen fest, dass die Distanz der Mappings mit der Anzahl der Topics logarithmisch wächst. Für die in der Auswertung betrachteten Topic-Modelle mit $K = 11$ und Hyperparametern der Größenordnung $\alpha \approx 0.1$ und $\beta \approx 0.1$ ergibt sich durchschnittlich ein optimales Mapping von $m_{\min} \approx 0.35$.

### 3.2.4. Bewertung von Modellen

Für die Bewertung der Klassifikationsleistung eines Modells nutzen wir wieder die HD. Ein Modell $\mathcal{M}$, welches aus den Dokumenten des Korpus $\mathcal{D}$ gelernt wurde, wird für die Bewertung immer mit einem Modell $\mathcal{M}' = \text{LDA}(\mathcal{D})$ verglichen. Dabei stellt $\mathcal{M}'$ das optimale mittels LDA errechnete Topic-Modell für $\mathcal{D}$ dar.

Beschreibe $\theta_d, \theta'_d$ die Dokument-Topic-Verteilung für jedes Dokument $d \in \mathcal{D}$ der beiden zu vergleichenden Topic-Modelle. Dabei sei das Mapping zwischen den Modellen bereits durchgeführt, d. h. wählen wir aus $\theta_d$ und $\theta'_d$ die Wahrscheinlichkeit für Topic 1, dann beschreibt Topic 1 in beiden Modellen inhaltlich die gleiche Topic.

Die Klassifikationsleistung $\mathcal{K}$ des Modells $\mathcal{M}$ ergibt sich mittels

$$\mathcal{K}(\mathcal{M}) := \begin{cases} \frac{\sum_{d \in \mathcal{D}} \text{HD}(\theta_d, \theta'_d)}{|\mathcal{D}|} - m_{\min}(\mathcal{M}, \mathcal{M}') & \text{falls } \frac{\sum_{d \in \mathcal{D}} \text{HD}(\theta_d, \theta'_d)}{|\mathcal{D}|} > m_{\min}(\mathcal{M}, \mathcal{M}') \\ 0 & \text{sonst.} \end{cases} \tag{3.6}$$

Die Klassifikationsleistung basiert auf den HD der Dokument-Topic-Verteilungen beider Modelle. Die Distanzen für alle Dokumente werden mit dem arithmetischen Mittel zu einem Wert zusammengefasst. Der Wert des Mappings $\Delta_{m_{\min}(\mathcal{M},\mathcal{M}')}$ wird abgezogen, um $\mathcal{K}$ für verschiedene Modelle vergleichbar zu machen und Verzerrungen durch schlechte Mappings zu verhindern. Da die Klassifikationsleistung eine HD ist, stellen kleine Werte gute Klassifikationsleistungen dar.

### Perplexität

Als weiteres Maß zur Bewertung eines Modell wird die Perplexität [Ple13] genutzt. Die Perplexität bewertet das gelernte Topic-Modell auf Basis der Vorhersagbarkeit. Ein Modell mit einer hohen Perplexität ist wenig aussagekräftig und kann keine eindeutigen Aussagen treffen. Somit kennzeichnen geringe Werte gute Modelle.

Die Perplexität wird auf Basis der Log-Likelihood [Vie97] bestimmt und ist für das Modell des Korpus $\mathcal{D}$ wie folgt definiert

$$\text{perp}(\mathcal{D}) = \exp\left\{-\frac{\log p(\mathcal{D}|\varphi_k, \alpha)}{\text{Anzahl Wörter}}\right\}, \text{ mit } \exp\{x\} := 2^x. \tag{3.7}$$

## 3.3. Anpassung eines Modells

Abbildung 3.6 stellt die Schritte zur Simulation der veränderlichen Korpora und der damit zusammenhängenden Lernvorgänge dar. Dabei unterteilen wir in zwei verschiedene Schritte. Im ersten Schritt findet die Vorbereitung statt. Anschließend folgt die Anpassung der Modelle. Die Anpassung kann mehrfach hintereinander durchgeführt werden. Damit simulieren wir, dass über die Zeit laufend neue Dokumente hinzukommen.

### Vorbereitung

Um das Hinzufügen von Dokumenten zu simulieren, müssen wir zuerst unseren Korpus $\mathcal{D}$ in zwei Teile unterteilen, einen initialen Teil $\mathcal{I}_0$ und einen Held-Out-Teil $\mathcal{E}$. Letzterer stellt die Dokumente dar, die wir schrittweise hinzufügen wollen. Die einzelnen Schritte ergeben sich aus der Unterteilung von $\mathcal{E}$ in $L$ Teilmengen $e_1, ..., e_L$.

Jedes Dokument durchläuft anschließend eine Vorverarbeitung, dabei werden Satzzeichen, Wortendungen, Ziffern und nicht alphanumerische Zeichenketten entfernt.

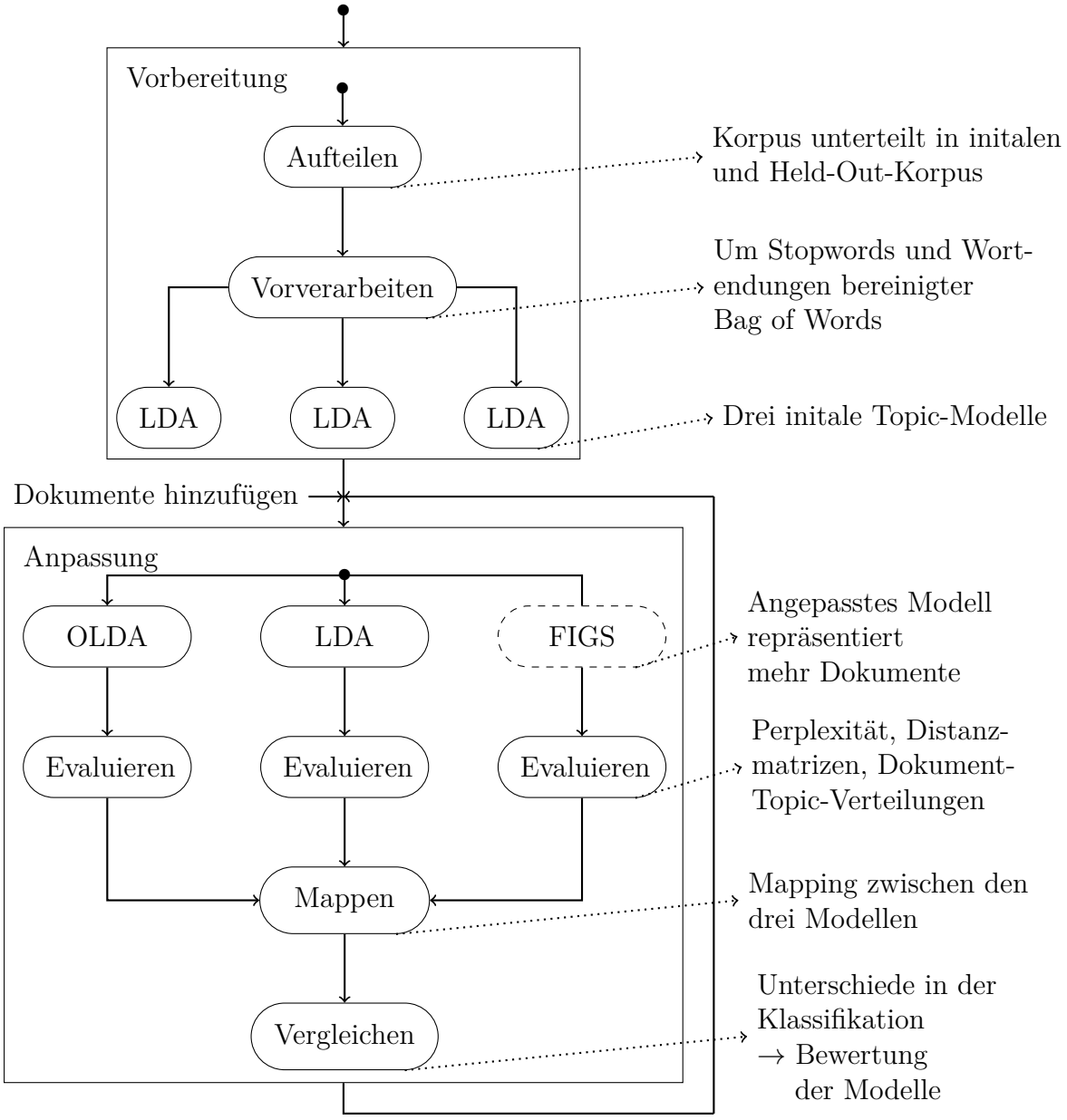


**Abbildung 3.6.:** Darstellung der Schritte zur Simulation der veränderlichen Korpora und der damit zusammenhängenden Lernvorgänge.

Außerdem werden 337 Stopwords [SDK11] entfernt. Nach der Vorverarbeitung lernen wir drei Topic-Modelle mit LDA für den initalen Korpus, je eines für OLDA, LDA und FIGS.

Für unseren Korpus $\mathcal{D}$ existiert nun eine Aufteilung in zwei Korpora $\mathcal{I}_0$ und $\mathcal{E}$, wobei $\mathcal{E}$ in weitere $L$ Teile aufgeteilt ist. Es gilt: $\mathcal{D} = \mathcal{I}_0 \cup \mathcal{E} = \mathcal{I}_0 \cup (e_1 \cup e_2 \cup \cdots \cup e_L)$. Weiterhin existieren drei Topic-Modelle $\mathcal{M}_{0,\text{LDA}} = \text{LDA}(\mathcal{I}_0)$, $\mathcal{M}_{0,\text{OLDA}} = \text{LDA}(\mathcal{I}_0)$ und $\mathcal{M}_{0,\text{FIGS}} = \text{LDA}(\mathcal{I}_0)$.

## Anpassung

Nach der Vorbereitung werden $L$ Anpassungen durchgeführt, jede Anpassung folgt dabei dem gleichen Muster. Sei $i = 1, ..., L$ die laufende Nummer der Anpassung.

Zuerst muss der Korpus $\mathcal{I}_{i-1}$ erweitert werden, indem er mit der Menge $e_i$ vereinigt wird ($\mathcal{I}_i = \mathcal{I}_{i-1} \cup e_i$). Anschließend können die drei Modelle an den erweiterten Korpus angepasst werden. Jedes Modell wird mit einem der Verfahren LDA, OLDA oder FIGS erweitert. Wahlweise wird inkrementelles oder initiales OLDA genutzt.

| | | | |
|---|---|---|---|
| LDA | $\mathcal{M}_{i,\text{LDA}}$ | = | $\text{LDA}(\mathcal{I}_i)$ |
| Inkrementelles OLDA | $\mathcal{M}_{i,\text{OLDA}}$ | = | $\text{OLDA}(\mathcal{M}_{i-1,\text{OLDA}}, e_i)$ |
| Initiales OLDA | $\mathcal{M}_{i,\text{OLDA}}$ | = | $\text{OLDA}(\mathcal{M}_{0,\text{OLDA}}, e_1 \cup ... \cup e_i)$ |
| FIGS | $\mathcal{M}_{i,\text{FIGS}}$ | = | $\mathcal{M}_{i-1,\text{FIGS}}$ |

Ein Modell $\mathcal{M}_i$ repräsentiert also den Korpus $\mathcal{I}_i = \mathcal{D} \setminus (e_{i+1} \cup ... \cup e_L)$. Um die Modelle vergleichen zu können, wird aus jedem Modell die Perplexität, die Hellinger- und Jaccard-Distanzmatrix sowie die Dokument-Topic-Verteilung extrahiert. Weiterhin werden die Topics der Modelle $\mathcal{M}_{i,\text{OLDA}}$ und $\mathcal{M}_{i,\text{OLDA}}$ auf die Topics des Modells $\mathcal{M}_{i,\text{LDA}}$ gemappt, siehe Unterabschnitt 3.2.3.

Bevor $i$ inkrementiert und zur nächsten Anpassung gewechselt werden kann, findet die Bewertung der Modelle statt. Die Bewertung besteht aus der Bestimmung der Perplexitäten und der Klassifikationsleistung nach Unterabschnitt 3.2.4.

# 4. Auswertung

Mit den erarbeiteten Verfahren zum Vergleich von Topic-Modellen können wir nun LDA, OLDA und FIGS untersuchen. Wir werden Modelle für verschiedene Konfigurationen lernen und deren Klassifikationsleistung und Modellperplexität vergleichen. Dabei werden wir die Größe der Korpora und die Themen der enthaltenen Textdokumente variieren. Außerdem werden wir die Laufzeit der drei Verfahren betrachten.

**Definition 4.1.** Eine Konfiguration $(\mathcal{D}, K, \alpha, \beta)$ beschreibt die vor den Lernvorgängen festzulegenden Werte. Sie besteht aus

- dem Korpus $\mathcal{D}$, unterteilt in $\mathcal{D} = \mathcal{I}_0 \cup \mathcal{E} = \mathcal{I}_0 \cup (e_1 \cup ... \cup e_L)$,
- der Anzahl der Topics $K$ sowie
- den Hyperparametern $\alpha$ und $\beta$.

Im allgemeinen Fall kann noch die maximale Anzahl der Iterationen während des Lernens eines Modells angegeben werden. Wir wollen die Ergebnisse nicht durch kleine Werte für die Iterationen verfälschen, daher limitieren wir immer mit einem ausreichend hohen Wert wie 10 000 bzw. warten auf die Konvergenz bei OLDA.

## 4.1. Dokumentenauswahl

Alle Korpora bestehen aus Textdokumenten in englischer Sprache. *Gensim* liefert die besten Ergebnisse in der Vorverarbeitung englischsprachiger Texte. Außerdem werden in der Literatur typischerweise Topic-Modelle für englischsprachige Korpora untersucht. Somit können die Ergebnisse dieser Arbeit mit anderen Ergebnissen verglichen werden und in fortführende Arbeiten einfließen.

Als Korpus nutzen wir 20 Newsgroups[1]. 20 Newsgroups ist ein bekannter Korpus bestehend aus dem E-Mail-Verkehr von 20 E-Mail-Newsgruppen. Thematisch können die 20 Gruppen in sechs gröbere Themengebiete unterteilt werden. Der gesamte Korpus besteht aus 18 828 Textdokumenten, die Dokumente haben dabei zwischen 1 und 39 682 Wörter mit einem Median von 160 Wörtern.

[1] `http://qwone.com/~jason/20Newsgroups/`

Wir nehmen die bereits vorverarbeitete Version von 20 Newsgroups, ohne Duplikate und nur mit Absender und Betreff. Weiterhin entfernen wir die Absender, da andernfalls in den Modellen die Topics durch Worte wie *.com* charakterisiert werden. Aus dem gleichen Grund löschen wir auch das Wort *Subject* aus den Betreffzeilen.

Die sechs groben Themengebiete sind:

- Computer, 5 Newsgruppen
- Sport, 4 Newsgruppen
- Wissenschaft, 4 Newsgruppen
- Zu verkaufen, 1 Newsgruppe
- Politik, 3 Newsgruppen
- Religion, 3 Newsgruppen

Wir können verschiedene Teilmengen des 20 Newsgroups Datensatzes für unsere Konfigurationen auswählen, dabei können wir aus gleichen oder verschiedenen Themengebieten wählen. Auch sind ausreichend viele Dokumente vorhanden, um repräsentative Korpora zu erstellen.

Nutzen wir den ganzen 20 Newsgroups Datensatz, so ist $K = 11$ eine gute Wahl für die Anzahl der Topics. Jedes Themengebiet kann durch zwei Topics repräsentiert werden, nur *Zu verkaufen* erhält nur eine Topic, da es aus nur einer Newsgruppe besteht. Abbildung A.1 im Anhang zeigt die Perplexitäten für andere $K$.

## 4.2. Quellcode und Rohdaten

Auf der CD im Anhang befinden sich die für die Berechnung der Modelle und Auswertung der Ergebnisse genutzten Skripte und Programme. Die Programme sind in den Programmiersprachen Bash, Python, PHP und R geschrieben und liegen teilweise in einem Git Repository vor. Die Datei *Datasets_BA.zip* beinhaltet Rohdaten der Ergebnisse sowie Skripte zur Erstellung der Abbildungen in dieser Arbeit. Die Datei *Weitere_Datasets.zip* enthält weitere Ergebnisse und Grafiken, die jedoch nicht für diese Arbeit genutzt wurden.

Die Datei *Gensim_Code_Alt.zip* beinhaltet die genutzten Programme zur Berechnung der Modelle. Die Programme liegen in einem Git Repository vor. Verschiedene Branches berechnen die Modelle mit verschiedenen Ansätzen. Allgemein hat jeder Branch ein Skript zum Starten der Lernvorgänge und eine JSON-Datei zu Konfiguration im obersten Verzeichnis. Die Ergebnisse werden je in einem neuen Unterordner unter *saves* gesichert. Es gibt weiterhin zwei überarbeitete Versionen

des alten Codes. Den neuen Versionen liegt eine Readme-Datei mit Erklärungen zur Funktionsweise bei. Es gibt eine Version zur Berechnung von Perplexität und Klassifikationsleistung sowie eine Version nur für die Berechnung der Perplexität.

## 4.3. Perplexität

Die erste Bewertung eines Modells geschieht auf Basis der Perplexität. Wir unterscheiden zwischen der Modellperplexität und der Held-Out-Perplexität. Die Modellperplexität wird über alle Dokumente $\mathcal{I}_i$, die das Modell nach Anpassung $i$ repräsentiert, berechnet. Die Held-Out-Perplexität wird nur über die neu hinzugefügten Dokumente $(\mathcal{I}_i \setminus \mathcal{I}_0)$ berechnet, also über die zu Anfang herausgehaltenen Dokumente. Anhand der Modellperplexität lassen sich Schlüsse für das ganze Modell ziehen, anhand der Held-Out-Perplexität erhalten wir Erkenntnisse über die Anpassung des Modells an die hinzugefügten Dokumente.
Um verlässliche Werte zu erhalten, wird jeder Vorgang zehn Mal ausgeführt und das arithmetische Mittel über die Ergebnisse aller zehn Durchläufe genutzt.

Die für Abbildung 4.1, Abbildung 4.2 und Abbildung 4.3 betrachten Modelle haben alle eine Konfiguration der Form $(\mathcal{D}, K = 11, \alpha = 0.1, \beta = 0.1)$, wobei wir nur $\mathcal{D}$ variieren. Variationen in den Hyperparametern führen zu allgemein schlechteren Modellperplexitäten. In Abbildung A.1 im Anhang sehen wir Variationen der Anzahl der Topics $K$. Der initiale Korpus besteht aus $|\mathcal{I}_0| \approx 8\ 000$ Dokumenten. Wir unterscheiden zwischen zwei Typen für die Aufteilung des Korpus $\mathcal{D}$:

**Typ 1** Die initial gelernten Dokumente $\mathcal{I}_0$ des Korpus stammen aus anderen Themengebieten als die hinzugefügten Dokumente $\mathcal{E}$.

**Typ 2** Die initial gelernten Dokumente $\mathcal{I}_0$ und die hinzugefügten Dokumente $\mathcal{E}$ sind so gewählt, dass jeweils alle Themengebiete vertreten sind.

Weiterhin untersuchen wir verschiedene Schrittgrößen für die hinzugefügten Dokumente $|e_i|$. Es gilt entweder $|e_1| = ... = |e_8| = 1\ 000$ oder $|e_1| = 1, |e_2| = 2, |e_3| = 3, |e_4| = 10,\ |e_5| = 100, |e_6| = 500, |e_7| = 1\ 000$.

Abbildung 4.1 und Abbildung 4.2 zeigen die Modellperplexitäten von Modellen, die mit OLDA, LDA und FIGS angepasst wurden.

In Abbildung 4.1 findet für LDA und FIGS keine Unterteilung in Typen statt, da sich beide Typen nicht sichtbar unterscheiden. Wir stellen fest, dass die Perplexität der mit LDA errechneten Modelle nahezu konstant bleibt. Auch bei FIGS steigt die Modellperplexität nur langsam an, wahrscheinlich dadurch bedingt, dass das initiale Topic-Modell ausschlaggebend für die Modellperplexität ist. Erst zum Ende wird

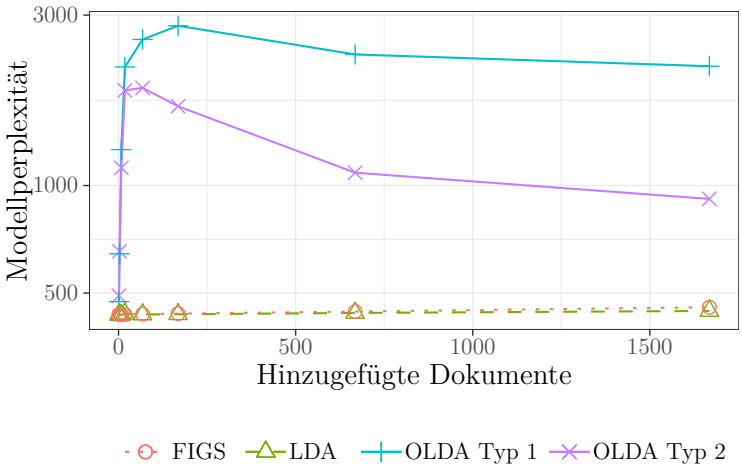


**Abbildung 4.1.:** Verlauf der Modellperplexitäten der Topic-Modelle berechnet mit LDA, inkrementellem OLDA und FIGS über mehrere Anpassungen mit kleinen Schrittgrößen.

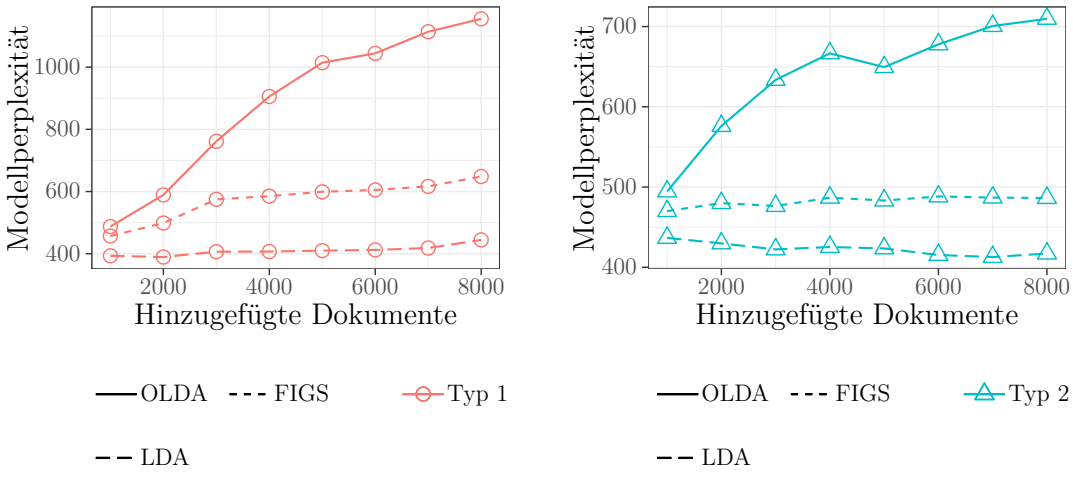


**Abbildung 4.2.:** Verlauf der Modellperplexitäten der Topic-Modelle berechnet mit LDA, inkrementellem OLDA und FIGS über mehrere Anpassungen mit großen Schrittgrößen.

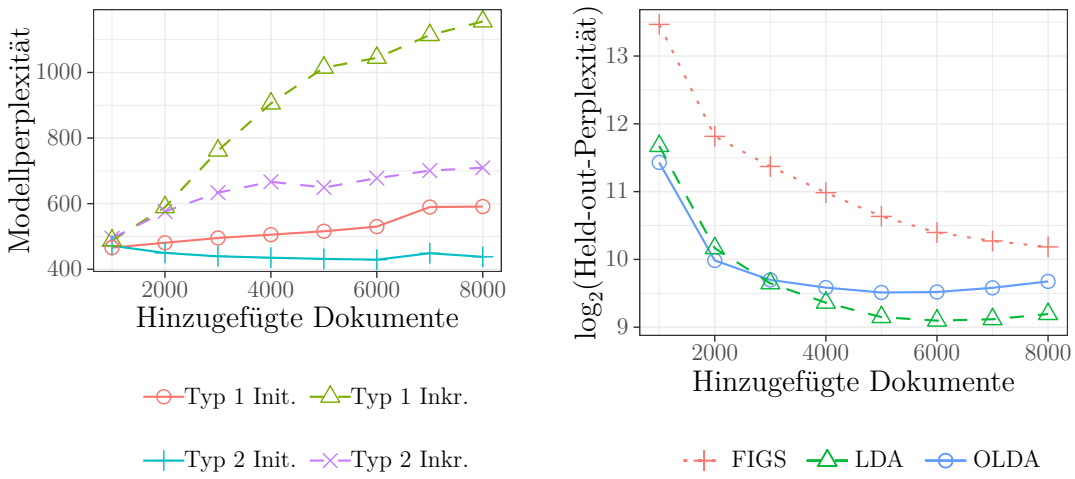


**Abbildung 4.3.:** (links) Verlauf der Modellperplexitäten der Topic-Modelle berechnet mit inkrementellem OLDA und initialem OLDA untereilt nach Typ 1 und Typ 2. (rechts) Held-Out Perplexität der Topic-Modelle berechnet mit LDA, inkrementellem OLDA und FIGS über mehrere Anpassungen. Arithmetisches Mittel über die Werte für Typ 1 und Typ 2.

die Modellperplexität auch über viele nicht im Modell berücksichtigte Dokumente errechnet und steigt damit bei FIGS an.

Interessant ist die hohe Modellperplexität der mit OLDA angepassten Modelle. Wenn nur eine kleine Anzahl Dokumente hinzugefügt wird, wird die Modellperplexität besonders hoch. Durch das spätere Hinzufügen weiterer Dokumente fällt die Modellperplexität wieder ab. Für kleine Anzahlen an Dokumenten reichen die von OLDA durchgeführten Iterationen scheinbar nicht aus, um das Modell ausreichend anzupassen. Das ganze Modell wird weniger aussagekräftig.

Beim Vergleich von Typ 1 und Typ 2 lässt sich in Abbildung 4.2 feststellen, dass die Modellperplexitäten von Modellen des Typs 2 besser sind. Typ 2 erreicht eine bessere Modellperplexität, da die Dokumente aus allen Themengebieten des Korpus gewählt wurden. So sind die hinzugefügten Dokumente thematisch dem Modell bereits bekannt. Bei Typ 1 sind die Themengebiete der hinzugefügten Dokumente dem Modell thematisch unbekannt und somit schwieriger in die bereits bestehenden Topics einzuordnen. Der Unterschied zwischen den Typen ist nur bei OLDA und FIGS deutlich sichtbar.

Abbildung 4.3 zeigt links die Modellperplexität der mit OLDA angepassten Modelle. Dabei vergleichen wir inkrementelles OLDA mit initialem OLDA und unterscheiden nach Typ 1 und Typ 2. Die Modellperplexität der mit initialem OLDA gelernten

Modelle ist besser als die Modellperplexität der mit inkrementellem OLDA gelernten Modelle. Eine Anpassung eines Modells mit OLDA an neue Dokumente scheint die Modellperplexität dauerhaft zu verschlechtern, sodass mehrfaches Anpassen eines Modells zu deutlich schlechteren Modellperplexitäten führt. Fügen wir die neuen Dokumente immer dem zu Anfang mit LDA gelernten Modell $\mathcal{M}_{0,\mathrm{OLDA}}$ hinzu (initiales OLDA), so erreichen wir deutlich bessere Modellperplexitäten. Weiterhin sehen wir, wie schon in Abbildung 4.1 und Abbildung 4.2, dass Typ 1 schlechtere Werte für die Modellperplexität erreicht als Typ 2.

Abbildung 4.3 zeigt rechts die Held-Out-Perplexitäten. Die Held-Out-Perplexität ist für kleine Anzahlen hinzugefügter Dokumente sehr hoch und fällt mit steigender Anzahl hinzugefügter Dokumente ab. FIGS erzeugt die Modelle mit den schlechtesten Werten. Die schlechten Werte für FIGS folgen aus der fehlenden Anpassung an die für die Held-Out-Perplexität untersuchten Dokumente. OLDA und LDA erreichen ähnliche Werte, ab 3 000 hinzugefügten Dokumenten wird LDA besser als OLDA. OLDA nähert sich bei 8 000 hinzugefügten Dokumenten FIGS an. Nach vielen hinzugefügten Dokumenten wird die Held-Out-Perplexität für OLDA wie schon die Modellperplexität schlecht im Vergleich zu LDA.

## 4.4. Klassifikationsleistung

Neben der Bewertung eines Modells durch die Perplexität ist die Klassifikationsleistung ein wichtiges Merkmal. Wir wollen die berechneten Topic-Modelle zur Unterteilung der Dokumente in Topics nutzen. Daher muss ein an weitere Dokumente angepasstes Modell die alten und die neuen Dokumente richtig klassifizieren. Wie in Unterabschnitt 3.2.4 festgelegt, nehmen wir ein mit LDA gelerntes Modell $\mathcal{M}_{i,\mathrm{LDA}}$ zum Vergleich.

Wir nutzen für die Untersuchung der Klassifikationsleistung die gleichen Konfigurationen, die wir bereits in Abschnitt 4.3 Perplexität genutzt haben. In den Abbildungen zur Klassifikationsleistung invertieren wir die y-Achse, damit weiter oben verlaufende Linien bessere Klassifikationsleistungen kennzeichnen.

In Abbildung 4.4 sehen wir die Klassifikationsleistungen für mit inkrementellem OLDA, initialem OLDA und FIGS angepasste Modelle. Der Vergleich der linken und rechten Abbildung zeigt, dass die Klassifikationsleistungen bei kleinen Anzahlen an hinzugefügten Dokumenten besser ist als bei größeren Anzahlen. Auch ist der Abstand zwischen initialem und inkrementellem OLDA rechts größer. Wir müssen jedoch beachten, dass bei der Berechnung der Klassifikationsleistung die HD des Mappings abgezogen wird. Abbildung A.2 zeigt, dass die Mappings für OLDA mit kleinen Anzahlen an hinzugefügten Dokumenten sehr groß werden. Somit täuschen

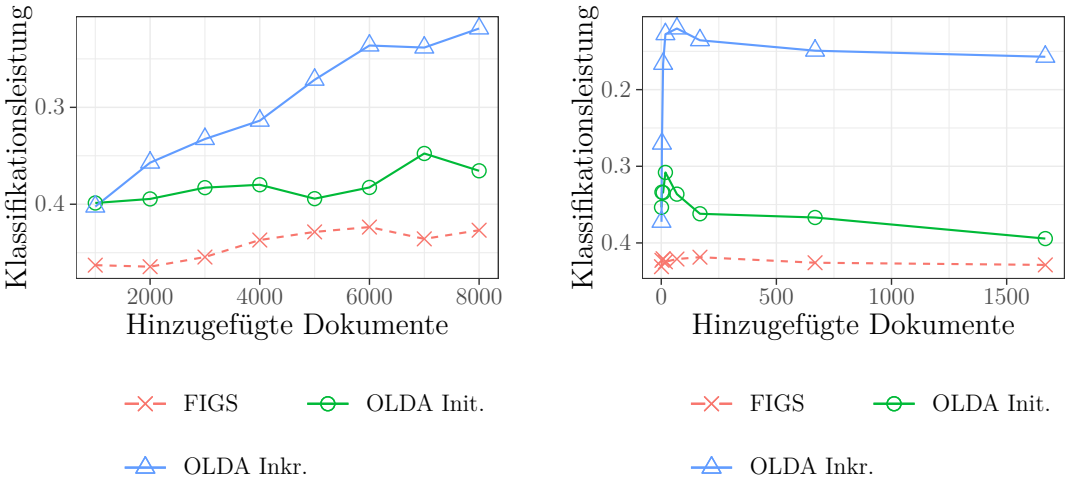


**Abbildung 4.4.:** Klassifikationsleistungen von mit FIGS, initalem OLDA und inkrementellem OLDA angepassten Topic-Modellen. Arithmetisches Mittel über die Werte für Typ 1 und Typ 2.

die zu Anfang extrem guten Klassifikationsleistungen von OLDA in der rechten Abbildung etwas.

Im Vergleich von FIGS und OLDA stellen wir fest, dass FIGS erwartungsgemäß die schlechteste Klassifikationsleistung erreicht. Inkrementelles OLDA erreicht die beste Klassifikationsleistung. Initiales OLDA liegt in der Mitte. Während die Werte für initiales OLDA und FIGS auch bei großen Mengen an hinzugefügten Dokumenten annähernd konstant bleiben, wird inkrementelles OLDA bei großen Mengen an hinzugefügten Dokumenten immer besser.

Abbildung 4.5 zeigt nur die Klassifikationsleistungen für OLDA. Die Linien von Typ 1 und Typ 2 verlaufen jeweils für inkrementelles und initiales OLDA nah beieinander. Typ 2 erreicht wie schon bei der Perplexität meist bessere Werte als Typ 1. Die Unterschiede zwischen initialem und inkrementellem OLDA passen zu unseren Erkenntnissen aus Abbildung 4.4, inkrementelles OLDA hat eine bessere Klassifikationsleistung als initiales OLDA.

Bei der Perplexität haben wir zwischen der Modellperplexität und der Held-Out-Perplexität unterschieden. Für die Klassifikationsleistung wollen wir nun analog die Klassifikationsleistung für den vollständigen Korpus $\mathcal{I}_i$ und Held-Out-Korpus $\mathcal{I}_i \setminus \mathcal{I}_0$ untersuchen. Abbildung 4.6 zeigt links die Klassifikationsleistung von OLDA und rechts von FIGS auf dem vollständigen Korpus im Vergleich zur Klassifikationsleistung auf dem Held-Out-Korpus.

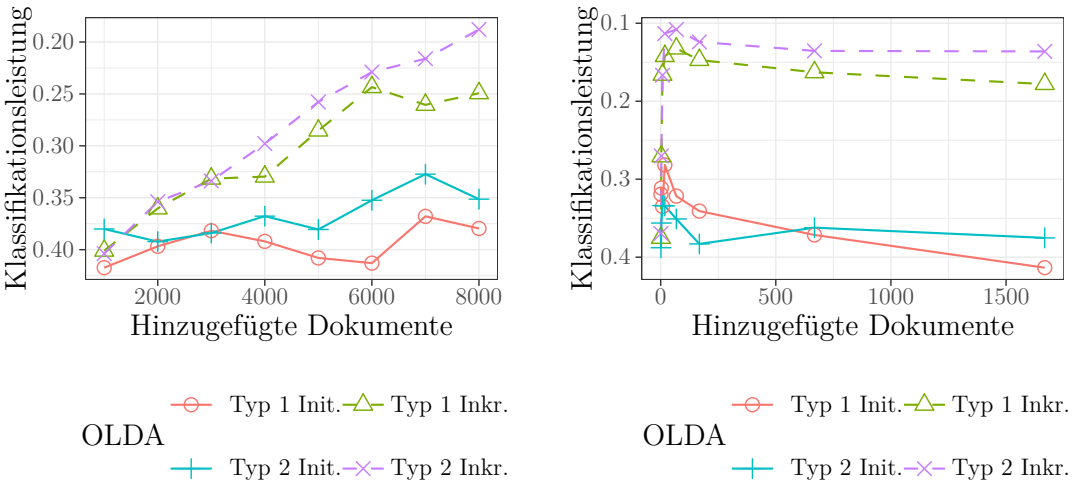


**Abbildung 4.5.:** Klassifikationsleistungen von Topic-Modellen, die mit inkrementellem OLDA und initialem OLDA angepasst wurden, unterteilt nach Typ 1 und Typ 2.

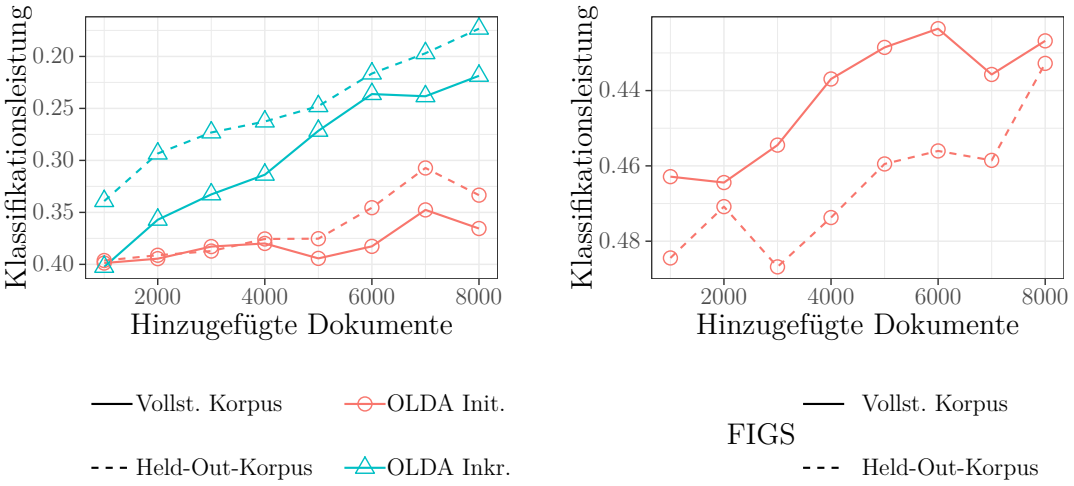


**Abbildung 4.6.:** Klassifikationsleistungen für den vollständig nach der Anpassung zu repräsentierenden Korpus sowie nur für den Held-Out-Korpus. Inkrementelles OLDA und initales OLDA sehen wir auf der linken Seite, FIGS auf der rechten. Arithmetisches Mittel über die Werte für Typ 1 und Typ 2.

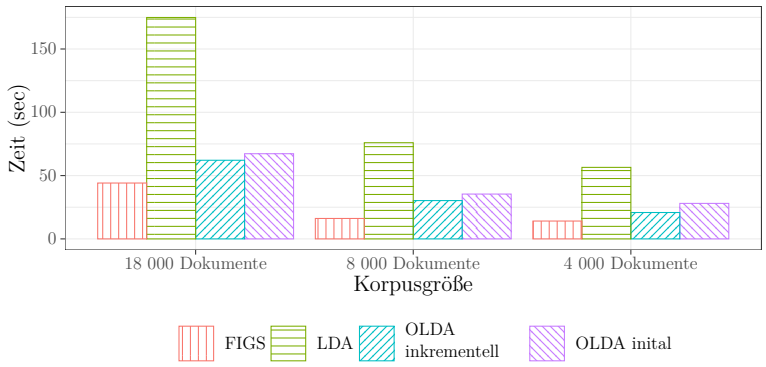


**Abbildung 4.7.:** Laufzeit für drei Anpassungen mit FIGS, LDA, inkrementellem und initialem OLDA für Korpora mit 18 000, 8 000 und 4 000 Dokumenten.

FIGS erreicht auf dem Held-Out-Korpus eine schlechtere Klassifikationsleistung als auf dem vollständigen Korpus. Obwohl FIGS die Dokumente des Held-Out-Korpus nicht lernen konnte, ist der Abstand zwischen den Klassifikationsleistungen auf dem vollständigen und Held-Out-Korpus gering. Mit wachsender Anzahl Dokumente im Held-Out-Korpus wird die Klassifikationsleistung besser, da mehr dem Modell bekannte Worte im Held-Out-Korpus vorhanden sind.

Initiales OLDA erreicht wie schon in Abbildung 4.5 für beide Korpora eine schlechtere Klassifikationsleistung als inkrementelles OLDA. Anders als bei FIGS ist die Klassifikationsleistung auf dem Held-Out-Korpus jedoch besser als auf dem vollständigen Korpus. Die von OLDA durchgeführten Anpassungen eines Modells führen zu einer besseren Repräsentation der hinzugefügten Dokumente. OLDA eignet sich daher auch gut zur Anpassung eines Topic-Modells, wenn die neuere Dokumente wichtiger als die älteren sind.

## 4.5. Laufzeit

Neben der Klassifikationsleistung und der Perplexität ist auch die Laufzeit der Verfahren interessant. Unsere Intention aus der Einleitung war es, Rechenleistung zu sparen. Daher ist für den Einsatz unter realen Bedingungen ein Vergleich der benötigen Laufzeiten für die Anpassung eines Modells wichtig.

Abbildung 4.7 zeigt die Laufzeiten für LDA, FIGS, initiales OLDA und inkrementelles OLDA. Es wird jeweils die benötigte Laufzeit für drei Anpassungen und das Lernen des initialen Modells dargestellt. Alle Zeiten wurden auf einer virtuellen Ma-

schine[2] gemessen. Dabei wurde nur die Laufzeit der LDA- und OLDA-Operationen ohne die benötigte Zeit für die Vorbereitung der Korpora einbezogen.

Die Messung der Zeit fand für drei verschiedene Größen von Korpora statt $|\mathcal{D}| = 18\ 000, |\mathcal{D}| = 8\ 000$ und $|\mathcal{D}| = 4\ 000$. Für $|\mathcal{D}| = 8\ 000$ und $|\mathcal{D}| = 4\ 000$ wurde $\mathcal{D}$ in zwei gleich große Teile $\mathcal{I}_0$ und $\mathcal{E}$ geteilt. Bei $|\mathcal{D}| = 18\ 000$ folgen wir dem Verhältnis 20% der Dokumente in $\mathcal{E}$ zu 80% in $\mathcal{I}_0$. Für die drei Anpassungen wird $\mathcal{E}$ weiterhin in drei gleich große Teile $e_1, e_2, e_3$ unterteilt. Nur die Variation von $|\mathcal{D}|$ führt zu deutlich unterschiedlichen Laufzeiten, Variationen bei den Themengebieten der Texte haben keinen Einfluss auf die Laufzeiten der Verfahren.

Jeder Balken in Abbildung 4.7 beschreibt die Summe der Laufzeiten für einen initialen Lernvorgang und drei Anpassungen. Der grüne Balken für klassisches LDA stellt somit die Summe der benötigten Zeit für die folgenden vier Operationen dar:

$$\mathrm{LDA}(\mathcal{I}_0),\quad \mathrm{LDA}(\mathcal{I}_0 \cup e_1),\quad \mathrm{LDA}(\mathcal{I}_0 \cup e_1 \cup e_2),\quad \mathrm{LDA}(\mathcal{I}_0 \cup e_1 \cup e_2 \cup e_3)$$

Bei der Betrachtung der verschiedenen Korpusgrößen stellen wir zuerst fest, dass die Laufzeiten der vier Verfahren bei wechselnden Korpusgrößen zueinander proportional sind. Größere Korpora haben eine allgemein längere Laufzeit als kleinere, die Verhältnisse der vier Verfahren untereinander bleiben erhalten.

Betrachten wir nun die erste Gruppe von Balken, so erreicht LDA die längste Laufzeit. Der Balken von LDA ist um ein Vielfaches höher als die anderen drei Balken. Der Balken für FIGS ist nur ungefähr ein Viertel so hoch wie der Balken für LDA. Der Faktor vier folgt aus der Tatsache, dass FIFS nur einmal ein Topic-Modell lernen muss und LDA vier Mal.
Vergleichen wir nun die Balken der beiden OLDA-Verfahren, so fällt uns auf, dass inkrementelles OLDA ein wenig schneller als initiales OLDA ist. Da initiales OLDA immer das Modell für den Korpus $\mathcal{I}_0$ an neue Dokumente anpasst und somit einige Dokumente mehrfach lernen muss, fällt der Abstand überraschend klein aus. Im Vergleich zu FIGS ist die Laufzeit beider OLDA-Verfahren nur wenig länger und erreicht nur ca. ein Drittel der Laufzeit von LDA.

## 4.6. Verfahrensvergleiche

Wir haben die Verfahren LDA, FIGS, inkrementelles OLDA und initiales OLDA unter den Gesichtspunkten Perplexität, Klassifikationsleistung und Laufzeit untersucht. Wir wollen die Ergebnisse nun zusammenfassen und die besten Ansätze in verschiedenen Situationen bestimmen.

[2] 4 Intel Xeon E5-2620 v3 Kerne mit je 2.40GHz und 16 GB RAM, Ubuntu 14.04 LTS

### FIGS und LDA

LDA führt zu der besten Perplexität und stellt das Optimum bei der Klassifikationsleistung dar. Obwohl FIGS keine Änderungen an dem einmalig initial gelernten Modell durchführt, sind die Ergebnisse doch überraschend gut. Gerade bei nur wenigen hinzugefügten Dokumenten oder Dokumenten aus einem ähnlichen Themengebiet liefert FIGS ein schnelles Verfahren zur Anpassung von Topic-Modellen.

Wer ein optimales Topic-Modell benötigt, sollte auf LDA zurückgreifen. Falls aber kein aktuelles Modell verfügbar ist, kann FIGS zur Überbrückung genutzt werden.

### LDA und OLDA

OLDA ist eine oft verwendete Weiterentwicklung von LDA. Gerade bei kleinen Mengen von hinzugefügten Dokumenten schnitt OLDA bei der Perplexität sehr schlecht ab. OLDA ist somit kein Verfahren um mehrmals ein Topic-Modell an einzelne Dokumenten anzupassen. Für größere Mengen an hinzugefügten Dokumenten erreicht OLDA gute Perplexitäten bei einer im Vergleich zu LDA unschlagbaren Laufzeit und einer guten Klassifikationsleistung. Statt mit inkrementellem OLDA immer das zuletzt errechnete Modell um Dokumente zu erweitern, ist initiales OLDA ein Kompromiss aus neu lernen und anpassen. Mit etwas schlechteren Laufzeiten schneidet initiales OLDA in der Perplexität und Klassifikationsleistung besser als inkrementelles OLDA ab.

Die Nutzung von OLDA und LDA zusammen ist weniger sinnvoll, da OLDA alle Funktionen von LDA bietet und wie LDA genutzt werden kann, indem ein leerer Korpus um Dokumente erweitert wird. Viele der verfügbaren Bibliotheken verzichten auf eine Implementierung von LDA und bieten nur OLDA an.
Lernen wir regelmäßig ein neues Modell, so können wir in der Zwischenzeit gute Ergebnisse mit initialem OLDA erreichen.

### OLDA und FIGS

FIGS erreicht die beste Laufzeit und gute Perplexitäten. Die Perplexität der Modelle wird erst nach vielen hinzugefügten Dokumenten eines anderen Themengebietes schlecht. Bei der Klassifikationsleistung kann FIGS nicht mit OLDA mithalten. Die bei FIGS fehlenden Änderungen am Modell zeigen sich durch die schlechtere Klassifikationsleistung. OLDA passt das Modell an neue Dokumente an und kann so Dokumente aus unbekannten Themengebieten besser klassifizieren.

Für wenige hinzugefügte Dokumenten ist FIGS ein geeignetes Verfahren zur Erweiterung eines Topic-Modells. Sobald eine größere Menge hinzugefügte Dokumente vorhanden ist, sollte initiales OLDA durchgeführt werden.

# 5. Zusammenfassung und Ausblick

Wir haben verschiedene Verfahren zur Erweiterung von Topic-Modellen um neue Dokumente untersucht. Zuerst haben wir die theoretischen Grundlagen (Kapitel 2) betrachtet und Anwendungsfälle für Topic-Modelle (Kapitel 1) kennengelernt. Bevor wir die Verfahren zur Erweiterung von Topic-Modellen vergleichen (Unterabschnitt 3.2.4) konnten, mussten wir einen Algorithmus zur Berechnung von Mappings (Unterabschnitt 3.2.3) festlegen. Außerdem mussten wir geeignete Kriterien (Unterabschnitt 3.2.4) Perplexität, Klassifikationsleistung und Laufzeit für die Bewertung von Topic-Modellen finden. In Kapitel 4 konnten wir dann Topic-Modelle mit LDA, OLDA und FIGS lernen und bewerten.

Fassen wir die Ergebnisse aus Abschnitt 4.6 zusammen, so ist LDA gut dazu geeignet, ein neues Topic-Modell für einen Korpus zu lernen. Ist bereits ein Topic-Modell vorhanden, lassen sich OLDA und FIGS kombinieren. Für kleine Mengen an hinzugefügten Dokumenten ist FIGS ein schnelles Verfahren um kurzfristig Aussagen über die Dokumente treffen zu können. Für größere Mengen wird mittels initialem OLDA das Topic-Modell angepasst. In sehr großen Abständen kann es sinnvoll sein, ein neues Modell mit LDA zu lernen.

Als Fortsetzung dieser Arbeit wäre die Nutzung weiterer Metriken, wie der Kullback-Leibler-Divergenz [KL51], zum Vergleich der mit OLDA, LDA und FIGS erweiterten Topic-Modelle interessant. Auch eine Wiederholung der Untersuchungen mit anderen Typen von Korpora, wie Bildern oder Tondateien, wäre möglich.

Neben Änderungen an den Korpora ist auch der Wechsel auf eine erweiterte Form des zugrundeliegenden Topic-Modells interessant. Dynamische Topic-Modelle [BL06] wären eine mögliche Erweiterung. Mit DTMs können wir der Fragestellung nachgehen, ob das Lernen eines weiteren Intervalls im DTM eine gute Möglichkeit ist, die Topicverteilungen für neue Dokumente zu bestimmen. Weiterhin ist bei DTMs zu prüfen, ob Dokumente aus verschiedenen Intervallen untereinander sinnvoll verglichen werden können.

Wir haben uns bisher auf das Hinzufügen von Dokumenten beschränkt, aber manchmal müssen auch Dokumente aus einem Korpus entfernt werden. Das Löschen von Dokumenten stellt uns vor das gleiche Problem. Nach jedem Entfernen eines Dokumentes müsste ein neues Modell gelernt werden. Der dabei entstehende Rechenaufwand kann vielleicht durch Verfahren zum Entfernen von Dokumenten verringert

werden. Ein potentieller Ansatz wäre es, ein inverses Dokument zu erstellen und mit OLDA zu lernen. Wie ein inverses Dokument aussieht, wäre Teil der Untersuchung. Das inverse Dokument sollte sich mit den Topicverteilungen und dem Bag of Words eines Topic-Modells bestimmen lassen.

# A. Anhang

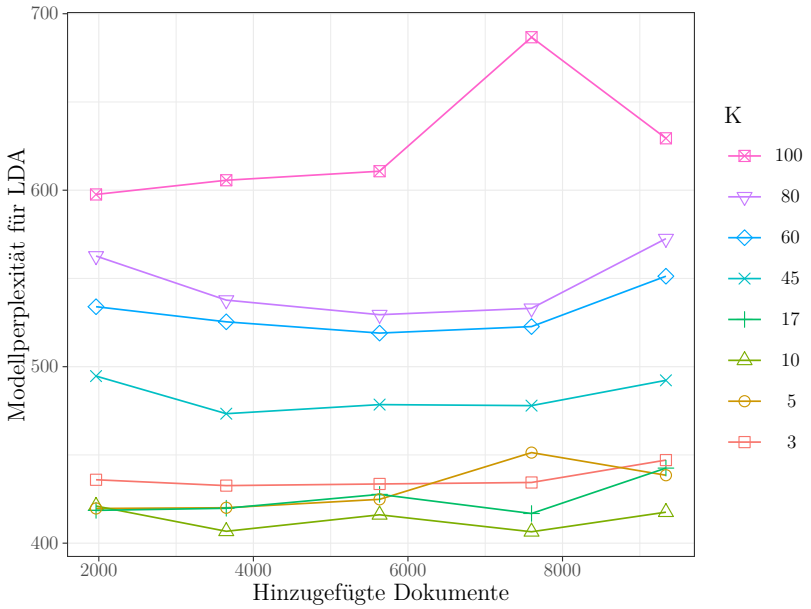


**Abbildung A.1.:** Modellperplexität von mit LDA berechneten Topic-Modellen für verschiedene $K$ auf dem 20 Newsgroups Datensatz und den symmetrischen Hyperparamtern $\alpha = \frac{1}{K}, \beta = \frac{1}{K}$.

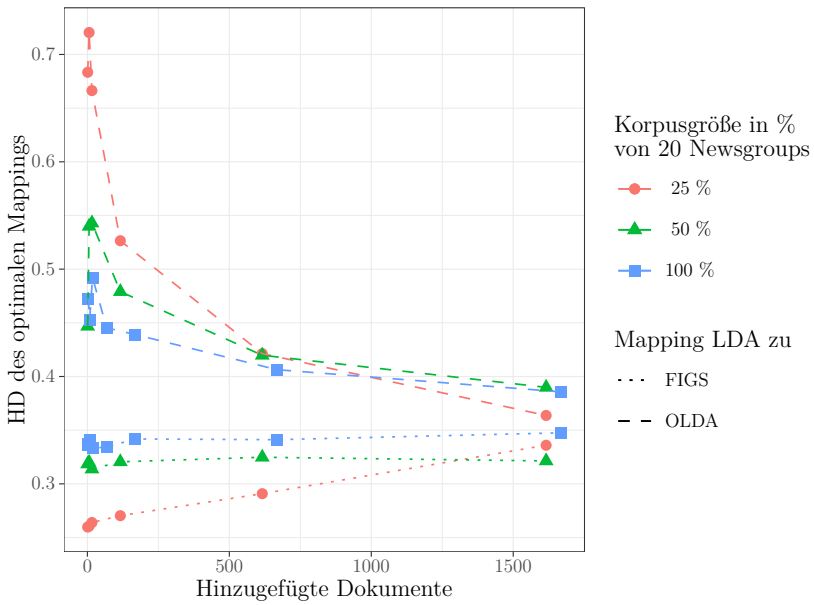


**Abbildung A.2.:** Beste Mappings für verschiedene Korpusgrößen und Typ 1. Es wird immer ein Mapping zwischen einem mit LDA und einem mit FIGS oder initialem OLDA berechneten Modell dargestellt.

# Abbildungsverzeichnis

# Literaturverzeichnis


[BL06] Blei, David M. ; Lafferty, John D.: Dynamic topic models. In: *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, 2006, 113–120

[BNJ03] Blei, David M. ; Ng, Andrew Y. ; Jordan, Michael I.: Latent Dirichlet Allocation. In: *Journal of Machine Learning Research* 3 (2003), 993–1022. `http://jmlr.org/papers/v3/blei03a.html`

[GG84] Geman, Stuart ; Geman, Donald: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 6 (1984), Nr. 6, S. 721–741

[Gri02] Griffiths, Tom: *Gibbs Sampling in the Generative Model of Latent Dirichlet Allocation.* 2002

[HBB10] Hoffman, Matthew D. ; Blei, David M. ; Bach, Francis R.: Online Learning for Latent Dirichlet Allocation. In: *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, 2010, 856–864

[Hel09] Hellinger, Ernst: Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen. In: *Journal für die reine und angewandte Mathematik* (1909), S. 210–271

[Jac01] Jaccard, Paul: Étude comparative de la distribution florale dans une portion des Alpes et des Jura. In: *Bulletin del la Société Vaudoise des Sciences Naturelles* 37 (1901), S. 547–579

[KL51] Kullback, S. ; Leibler, R. A.: On information and sufficiency. In: *Annals of Mathematical Statistics, Band 22, Nr. 1*, 1951, S. 79–86

[ML02] Minka, Thomas P. ; Lafferty, John D.: Expectation-Propagation for the Generative Aspect Model. In: *UAI '02, Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002*, 2002, 352–359

[Ple13] Pleplé, Quentin: *Perplexity To Evaluate Topic Models*. http://qpleple.com/perplexity-to-evaluate-topic-models/, 2013. – Zugriff auf die Webseite am 27 Juni 2019 um 17:25 Uhr

[ŘS10] Řehůřek, Radim ; Sojka, Petr: Software Framework for Topic Modelling with Large Corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta : ELRA, Mai 2010, S. 45–50. – http://is.muni.cz/publication/884893/en

[SDK11] Stone, Benjamin ; Dennis, Simon ; Kwantes, Peter J.: Comparing Methods for Single Paragraph Similarity Analysis. In: *Topics in Cognitive Science* 3 (2011), Nr. 1, 92-122. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1756-8765.2010.01108.x. – Gensim nutzt eine leicht angepasste Version der Liste aus Apendix C des Papers, https://github.com/RaRe-Technologies/gensim/blob/a3dbdcc59f3823b530db3ac38afdbb3df2761d6f/gensim/parsing/preprocessing.py#L46

[Vie97] *Kapitel* 5. In: Viertl, Reinhard: *Einführung in die Stochastik - mit Elementen der Bayes-Statistik und Ansätzen für die Analyse unscharfer Daten, 2. Auflage*. Springer, 1997 (Springer Lehrbuch Mathematik). – ISBN 978–3–211–83027–7, S. 110

[ZCP+15] Zhao, Weizhong ; Chen, James J. ; Perkins, Roger ; Liu, Zhichao ; Ge, Weigong ; Ding, Yijun ; Zou, Wen: A heuristic approach to determine an appropriate number of topics in topic modeling. In: *BMC Bioinformatics* 16 (2015), Dez, Nr. 13, S8. https://doi.org/10.1186/1471-2105-16-S13-S8

[Zha15] Zhao, Weizhong: *Best Practices in Building Topic Models with LDA for Mining Regulatory Textual Documents*. http://phusewiki.org/wiki/images/c/c9/Weizhong_Presentation_CDER_Nov_9th.pdf, 2015. – Zugriff auf das PDF am 13 Juni 2019 um 15:02 Uhr