„Dokumentation“

Topic-Modeling

Erstellt von:	Mats Hester / 920843
Am:	17.06.2020
Version:	0.0.7
Unterstützt von:	Benjamin Müller (LSA u. NMF)

Inhalt

Erläuterung

Token

Eine beliebige Zeichenreihenfolge innerhalb eines Dokumentes, es muss kein Wort sein.

Dokument

Einzelner Datensatz bestehend aus mehreren Tokens.

Korpus

Die vollständige Datenbasis bestehend aus mehreren Dokumenten.

Word-Dokument-Matrix

In einer Wort-Dokument-Matrix entsprechen die Spalten einem Dokument und die Zeilen einem Wort/Token.

1 Einleitung

1.1 Überblick

Das Topic-Modeling gehört zum Natural Language Processing und ist eine Disziplin, in der es darum geht mit stochastischen Methoden Textsammlungen zu analysieren und Ähnlichkeiten zu finden. Es gibt eine Vielzahl von Anwendungsmöglichkeiten in unterschiedlichen Branchen und Abteilungen. In der Wissenschaft wird das Topic-Modeling unteranderem genutzt, um wissenschaftliche Publikationen zu gruppieren. Für die IT-Branche kann mithilfe von Topic-Modeling die Benutzerfreundlichkeit von Webseiten erhöht werden. Beispielsweise kann das geschehen, indem Nachrichten oder Pressemitteilungen gruppiert werden, so dass der Benutzer die Informationen erhält, welche ihn interessieren. Aufgrund der steigenden Menge an Textdaten wird das Topic-Modeling immer wichtiger.

1.2 Ziel des Projektes

Das Ziel dieses Projektes ist es unterschiedliche Machine-Learning-Algorithmen zur Topic-Modellierung miteinander zur vergleichen und herauszufinden, welches dieser Algorithmen am besten geeignet sei, um Topics (Themen) aus einem Korpus zu extrahieren. Außerdem soll der Vorprozess des Topic-Modelings mit einigen seiner Methoden sowie die Topic-Modeling Algorithmen an sich erläutert werden. Die Validierung der Algorithmen wird ausschließlich anhand der Daten aus der Nachrichtenstelle der Bundesregierung erfolgen. Die zuvor mittels eines sogenannten Scraper erhoben werden müssen.

1.3 Gang der Untersuchung

Im zweiten Abschnitt dieser Dokumentation findet sich ein kurzer Überblick über die Theorie der Datenbeschaffung mittels Webcrawlern beziehungsweise Scrapern. Im nächsten Kapitel werden die Themen: Data-Mining, Text-Minings und im spezielleren dem Naturel Language Processing, zu dem auch das Topic-Modeling angerechnet wird, beschrieben. Dazu wird kurz auf Validierungsmöglichkeiten eingegangen und besonders auf die Methoden der Data-Preparation. Danach folgt eine kurze Erläuterung über die angewandten Algorithmen. Im vierten Kapitel findet sich eine Datenanalyse wieder, sowie die Bewertungen der Algorithmen anhand ihrer erbrachten Ergebnisse. Zum Abschluss findet ein Vergleich der Algorithmen statt und es werden Verbesserungsmöglichkeiten aufgezeigt.

2 Datenbeschaffung mittels Webcrawlern / Scraper

Der Webcrawler ist ein Bot, welcher das Sammeln von bestimmten Daten und Informationen aus dem Internet automatisiert. Solche Bots können auf unterschiedliche Server-Typen wie Beispielsweise Webserver oder Dateiserver zugreifen. Sie durchlaufen für die Datenerhebung einen Server und wechseln dann zum nächsten Server in der Liste. Aufgrund der Vielfältigkeit der Server und deren interne Struktur ist die Anzahl von Daten, welche von einem Webcrawler der viele Server durchlaufen soll, meist beschränkt. Häufig handelt es sich bei den Daten um Metadaten. Webcrawler haben einige Aufgabengebiete zum Beispiel die Indexierung von Webseiten für die Suchmaschinenanbieter oder für die Suche von Produktpreisen für Preisvergleichsportalen.

Scraper sind ebenfalls programmierte Bots die spezieller für einzelne Webseiten und deren Unterseiten entworfen werden. Die Anpassung an eine Website und an den HTML-Text sowie den JavaScript-Ausgaben, bei dynamisch generierten Inhalten, ermöglicht den Scrapern mehr und genauere Daten von einer Website zu erheben als es ein Webcrawler tun könnte. Die durch den Scrapern erhobenen Daten werden Beispielsweise für Analysen verwendet aber auch für bloße Kopien der Inhalte auf andere Webseiten. Der Einsatz von Scrapern ist umstritten solange keine Genehmigung des Websitebetreibers oder der Autoren vorliegt. Außerdem kann der Scraper hohen

Traffic verursachen, weil meistens viele Unterseiten in kurzer Zeit aufgerufen werden, was wiederum den Webseitenbetreiber stören könnte. Aus den genannten und anderen Gründen versuchen einige Betreiber von Webseiten die Datenerhebung durch Scraper zu verhindern.

3 Data-Mining

Data-Mining ist die Anwendung von Algorithmen zur automatischen oder halbautomatischen Auffindung von nützlichen Mustern in den erhobenen Daten (Witten und Frank 2005, S. 5; Omari 2008, S. 11–12). Dem Data-Mining zugehörigen Aufgaben sind:

1. Assoziationsanalyse : Finden von Abhängigkeiten und Zusammenhängen
2. Klassifikation : Einordnung von Entitäten in Klassen nach bestimmten Kriterien
3. Clusteranalyse : Verfahren, um Entitäten zu gruppieren
4. Vorhersage : Voraussage künftiger Entwicklungen
5. Regressionsanalyse : Verfahren zur Modellierung von Beziehungen zwischen unterschiedlichen Variablen und ebenfalls die Vorhersage
6. Sequenzmusteranalyse : Finden von Abhängigkeiten und Zusammenhänge innerhalb von Sequenzen
7. Generalisieren : Reduktion von Daten ohne entscheidenden Informationsverlust
8. Ausreiser-Erkennung : Identifizierung von abweichenden Daten
9. Visualisierung : Grafische Aufbereitung der Daten

Diese Aufzählung setzt sich aus verschieden Listen zusammen (Ngai et al. 2009, S. 2593; Fayyad et al. 1996, S. 44; sEster und Sander 2000, S. 4–5).

3.1 Methodiken des Data-Minings

Einige Autoren verstehen Data-Mining als den reinen Analyseschritt innerhalb der „Knowledge Discovery in Database“-Methodik kurz KDD. Der Prozess des Models beinhaltet folgende Phasen sEster und Sander 2000, S. 2:

1. Fokussierung
2. Vorverarbeitung
3. Transformation
4. Data-Mining
5. Evaluation

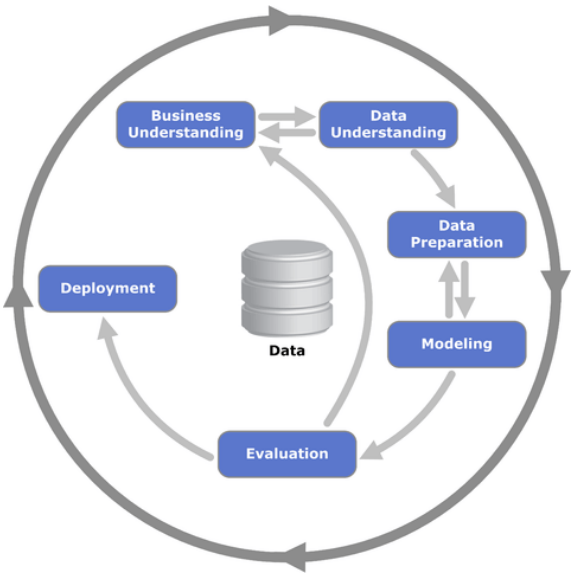


Abbildung 1: CRISP-DM-Standard (Smart Vision Europe Ltd.)

Im Gegensatz dazu wird Data-Mining auch als Oberbegriff für den Prozess verstanden. Auf dieser Meinung aufbauend entwickelte ein Konsortium aus über 200 Unternehmen den Offenen-Standard „Cross-Industry Standard Process for Data Mining“ in der bekannteren Form auch CRISP-DM genannt. Dieser Standard und die darin enthaltende Methodik setzte sich in der Branche durch (Brown 2015). Der CRISP-Standard (Abbildung 1) ist als Lebenszyklus zu verstehen, indem die Prozesse sich für jedes Data-Mining Projekt wiederholen.

Nebenher bietet die Methodik Rückschrittmöglichkeiten zur Verbesserung an. Die Aufgaben der einzelnen Phasen nach der Smart Vision Europe Ltd. sind:

- Business Understanding: Im ersten Schritt ist es wichtig eine Aufgabenstellung zu erarbeiten woraus dann Ziele und Anforderungen abgeleitet werden können, unter der Berücksichtigung der zur Verfügung stehenden Ressourcen und der möglichen Risiken und Chancen. Am Ende dieser Phase sollte ein Projektplan erstellt sein, welcher die erarbeiteten Informationen beinhaltet.
- Data-Understanding: Nach der Erhebung der Daten folgt die erste Sichtung und damit einhergehend die Anwendung der Methoden der Deskriptiven- und der Explorativen Statistik. Außerdem die Ermittlung der Datenqualität mit dem Fokus auf Ausreißer, fehlende und fehlerhafte Werte. Anschließende Beurteilung, ob die erarbeiteten Ziele mit den Daten zu erreichen sind.
- Data-Preparation: In der Phase geht es primär darum die Daten den Algorithmen anzupassen, um das optimale Ergebnis zu erzielen. Der erste Unterschritt ist die Selektion der wichtigen Variablen auch als „Feature-Selection“ bekannt. Danach folgt die Datenbereinigung (Data-Cleaning). D.h. die Bearbeitung von Ausreißern, fehlende und fehlerhaften Werten. An dieser Stelle können Variablen mit logischen oder mathematischen Operationen verändert oder erschaffen werden. Außerdem werden in dieser Phase die Datenquellen zusammengefasst, falls mehrere vorhanden sind. Die Formatierung, falls nicht vorher schon geschehen, und die Transformation[1] der Daten sind die letzte Teilschritte.
  In der Regel wird diese Phase mehrmals ausgeführt, um die beste Kombination der Features und Verfahren zur Bearbeitung von Features zu ermitteln, die sich erst bei der Validierung des Models zeigt.
- Modeling: Zuerst erfolgt die Auswahl geeigneter Data-Mining-Verfahren (Modelle) und entsprechender Techniken zur Validierung der trainierten Modelle. Darauf aufbauend die Bestimmung der Parameterwerte und das Training der Modelle. Danach folgt die Validierung und gegebenenfalls erneute Trainings bei angepassten Parameterwerten. Falls eine Verbesserung ausbleibt wird die vorherige Phase Data-Preparation wiederholen.
- Evaluation: Die trainierten Modelle und deren Resultate bewerten und miteinander vergleichen, mit dem Bezug zu den gesetzten Zielen. Bei ausreichenden Ressourcen eventueller Test unter realen Bedingungen. Außerdem sollte der vollzogene Prozess bis hierhin überprüft werden, um mögliche Fehler zu erkennen. Zum Beispiel, ob die zum Training des Modells verwendeten Daten auch in Zukunft erhoben werden können oder ob die Wahl der Validierungstechnik richtig war. Nach der Erstellung des Fazits steht die Entscheidung offen, wie weiter verfahren wird. Im besten Falle geht der Prozess weiter in die „Deployment“-Phase. Der schlechteste Fall bedeutet, dass das Ziel momentan oder gar nicht mit den Daten zu erreichen ist.
- Deployment: Die Implementierung des Modells sollte in Verbindung mit einer vorher angefertigten Strategie erfolgen. Stichwort Software-Einführung. Während der Laufzeit des Modells sollte stehts überprüft werden, ob die Ergebnisse konstant bleiben. Wenn diese sich verschlechtern sind weitere Schritte einzuleiten. Wie beispielsweise ein erneutes Training des Modells. Weiterhin empfiehlt sich das Anfertigen eines finalen Berichts und die Auswertung der gewonnen Erfahrungen (Lessons Learned).

3.2 Spezialisierung im Data-Mining

Die CRISP-DM-Methodik ist auch anwendbar in den einzelnen Spezialisierungen im Data-Mining, die sich aufgrund ihrer hohen Anzahl an Problemstellungen in den bestimmten Bereichen gebildet

[1] Transformation: Anwendung von Dimensionsreduktions- und Standardisierungsverfahren.

haben. Das Hauptaugenmerk liegt im Web-Mining, weshalb die anderen beiden nur grob definiert werden.

3.2.1 Zeitreihenanalyse

Umfasst neben der Analyse von zeitlich geordneten Daten (Zeitreihe) auf Muster und deren Einflussfaktoren, auch die Erstellung einer Prognose über deren Werteverlauf.

3.2.2 Text-Mining

Text-Mining ist die Kunst und Technologie zur Extraktion von Information und Wissen aus Textdokumenten.

3.2.3 Web-Mining

Web-Mining ist die Anwendung von Data-Mining-Techniken zur automatischen oder halbautomatischen Auffindung von nützlichen Mustern in den Daten, welche zuvor im Internet erhoben wurden.

3.3 Natural Language Processing

Natural Language Processing (NLP) ist eine Methode des Text-Mining in der es darum geht, die natürliche Sprache sowohl in Wort und in Schrift mithilfe von Algorithmen zu verarbeiten. Die Aufgaben des Natural Language Processing sind vielfältig und beinhalten unteranderem auch das hier in der Arbeit angewandte Topic-Modeling und deren Methoden zur Data-Preperation wie der Grammatischen Analyse, Lemmatisierung und Stemming. Aufgrund der Vielfältigkeit und der Komplexität der natürlichen Sprache sind die Daten unstrukturiert. Die Überführung in eine strukturierte Datenform, durch eine Transformation der Texte in Vektoren mit denen Algorithmen arbeiten können, ist eine der Hauptaufgaben in dem Bereich. Ein zu nennendes Problem innerhalb des Natural Language Processing ist die große Bandbreite an existierenden und genutzten Sprachen. Viele der Methoden sind abhängig von der Sprache, für die sie entwickelt wurden und können für keine andere Sprache verwendet werden. Im Vergleich zu dem Englischen gibt es für das Deutsche zum Beispiel deutlich weniger Variationen in den Methoden, welche angewandt werden können.

3.3.1 Topic-Modeling

Topic-Modeling ist ein Unsupervised Machine-Learning Verfahren[2] mit dem Ziel Cluster von Wortgruppen (Topics) innerhalb der Textdaten zu finden, indem nach Wörtern gesucht wird, welche häufig gemeinsam auftreten. Zum Beispiel könnte die Ausgabe durch das Topic-Modeling anhand von Sportnachrichten wie folgt aussehen: „Point Guard, Shot, Block, Crossover“ was zum Topic Basketball führen würde. Sind die Textdaten allerdings sehr homogen sprich es findet sich nur ein Thema wieder, ist die Suche nach Topics vergebens. Aber die Ausgabe des Topic-Modelings muss nicht so eindeutig sein wie in dem vorgebrachten Beispiel, selbst wenn die Textdaten unterschiedliche, klar voneinander trennbare Themen beinhalten. Dieser Umstand kann mehrere Ursachen haben wie eine zu geringe Datenmenge, eine unausgereifte Datenvorverarbeitung oder die falsche Wahl des Algorithmus.

3.3.1.1 Validierung

Aufgrund das es sich bei dem Topic-Modeling, wie oben angesprochen, um ein Unsupervised Verfahren handelt, ist die Sichtung des Resultates also der Topics unabdingbar, um Güte einzuschätzen. Solch eine Überprüfung ist subjektiv und kann auch dazu führen, dass der Prüfer die Zusammenhänge der einzelnen Wörter nicht erkennt. Zur Einschätzung kann auch auf gewisse

[2] Unüberwachtes Lernen: Der Algorithmus muss ohne einen Ergebnisvektor die Muster in den Daten finden.

Metriken zurückgegriffen werden wie zum Beispiel der Coherence-Score, welcher die durchschnittliche Distanz der Wörter innerhalb der Topics misst (Röder, M. 2015). Es gibt verschiedene Arten von Berechnungen des Coherence-Scores zum Beispiel u_mass welcher unter null liegt und c_v der zwischen null und eins liegt. Es gilt aber immer: Je höhere desto kohärenter und besser sind die Topics. Neben diesem Score gibt es auch noch die Perplexity (Ratlosigkeit) für den Algorithmus Latent-Dirichlet Allocation. Dieser Score basiert auf der durchschnittlichen Wahrscheinlichkeit, mit der das Model Sätze vervollständigen kann. Ein Model mit dem geringsten Perplexity-Score ist anderen Modellen vorzuziehen (Obacht: Bei der „Log_perplexity“ Funktion von Gensim verhält es sich umgekehrt).

3.3.2 Data-Preparation

Die Methoden des Data-Preparation unterscheiden sich stark von denen anderer Data-Mining Bereiche, nicht nur aufgrund der Daten, sondern vor allem auch wie die Algorithmen arbeiten. Wie im vorherigen Unterkapitel angesprochen ist die Häufigkeit der Wörter ein essenzieller Faktor im Topic-Modeling. Innerhalb der Data-Perparation wird nun versucht diesen Faktor positiv zu beeinflussen, indem die Wörter beziehungsweise Tokens bearbeitet werden. Nicht relevante Tokens werden aussortiert und die Häufigkeit anderer Tokens wird durch geschickte Manipulation erhöht. Welche Methoden einzusetzen sind und in welcher Reihenfolge, ist abhängig von den Daten und kann im Vorfeld nur grob eingeschätzt werden.

3.3.2.1 Feature Selection und Engineering

Die Auswahl geeigneter Features gestaltet sich in diesem Bereich einfacher als in den anderen Disziplinen. Die Features müssen lediglich Texte beinhalten, welche logisch zu der Bildung von den zu erwarteten Topics beitragen. Sollen etwa zu Pressemitteilungen Topics gefunden werden, sollten die Pressemitteilungen als Feature selbstredend vorkommen und zum Beispiel nicht der lokale Wetterbericht des jeweiligen Tages, an dem die Pressemitteilung veröffentlicht wurde. Die ausgewählten Features können dann als ein Feature zusammengefasst werden.

3.3.2.2 Entfernen von Zeichen

Dieser Schritt wird in den meisten Fällen durchgeführt, weil die Wörter ihre Bedeutung nicht verlieren und die Häufigkeit der Wörter zunimmt. Ein Wort mit einem Zeichen am Ende oder am Anfang wird als eigenständiges Wort verarbeitet, auch wenn dieses Wort ohne Zeichen in den Daten vorkommt. Neben den sichtbaren Satzzeichen wie Beispielsweise das Ausrufezeichen oder das Fragezeichen existieren in Textdateien ebenso Zeichen für Zeilenumbrüche („\n“), Tabulatoren („\t“) und so weiter. Solche Zeichen können ohne weiteres entfernt werden. Außerdem sollten sprachspezifische Schriftzeichen angepasst werden. Sprich aus dem Umlaut „ae“ wird „ä“ oder umgekehrt, um ein einheitliches Bild zu schaffen; eine entsprechende Zeichencodierung vorausgesetzt.

3.3.2.3 Entfernen von Wörtern/Tokens

Das Entfernen von Wörtern, die besonders häufig in einer Sprache vorkommen, sogenannte Stoppwörter kann ebenfalls zu besseren Topics führen. Das Herausfiltern dieser Wörter basiert auf vordefinierte Stoppwortlisten für die jeweilige Sprache. In einer Deutschen Stoppwortliste befinden sich zum Beispiel: „Der, die, das, meine, deine, seine, …“. Ein Kritikpunkt der Stoppwörter ist, dass Eigennamen, die solch ein Stoppwort verwenden, verändert werden. Aus dem Buch „Der Hobbit“ wird nach dem Filtern „Hobbit“ das Wesen. Ergänzend oder als Ersatz können auch die häufigsten Wörter innerhalb der zu verarbeitenden Textdaten herausgefiltert werden. Dabei muss die Grenze, ab wann ein Wort aussortiert wird, vordefiniert werden. Das kann sowohl vorteil- als auch Nachteilhaft sein.

Sinnvoll kann auch das Aussieben nach bestimmten Wortarten sein, weil aus ihnen eventuell nicht klar erkennbar ist zu welchem Themenbereich sich der Token zuordnen lässt. Beispielsweise bei Adverbien „hier, da, morgen, gern, darum“. Das Filtern nach Wortarten erfordert den Einsatz eines Wörterbuches bzw. einer Programmbibliothek, die jene Funktionalität unterstützt.

3.3.2.4 N-Grams

Unter dem Erstellen von Bigrams und Trigrams wird die Zusammenführung von häufig in Reihenfolge auftreten Tokens verstanden. Damit können solche Phrasen wie „Weiße Haus“ als ein Token verarbeitet werden (Bigram). Ein Trigram besteht aus drei Wörtern.

3.3.2.5 Bilden von Wortstämmen

Eine weitere Möglichkeit Wörter anzugleichen und somit die Worthäufigkeit zu erhöhen, ist die Überführung der Wörter in ihre Grundform bzw. in ihren Wortstamm. Für diesen Zweck existieren zwei Methoden die Lemmatisierung und das Stemming, welche substitutiv ihren Einsatz finden. Die Lemmatisierung oder besser bekannt als Lemmatization basiert auf die Nutzung von Lexika und auch, bei einigen Implementierungen, auf Regeln für die Rückführung der Wörter in ihre Lemmata (Grundformen). Das Stemming erstellt dagegen die Grundform allein mithilfe von Regeln also Algorithmen. Aus diesem Grund entstehen in einigen Fällen, durch die Herleitung mithilfe des Stemmingverfahren, allerdings ‚fiktive‘ also keine richtigen Wörter wie im unteren Beispiel gezeigt.

Beispiel Lemmatisierung: Liefen -> Laufen, Häuser -> Haus

Beispiel Stemming: Speicherbehälter -> Speicherbehält, Grenzpostens -> Grenzpost

3.3.2.6 BagsOfWord (BoW)

Da die Topic-Modeling Algorithmen nicht explizit mit den Wörtern arbeiten können, müssen diese umgewandelt werden. Ein bekanntes und häufig eingesetztes Verfahren ist die einfache Erstellung eines Bags of Words Modells. Dazu nochmal die Erklärung der Struktur: Jeder Datensatz innerhalb des Textkorpus mit seinem Textfeature stellt ein Dokument dar. Diese Dokumente bestehen aus Tokens, welche wiederum einfache Zeichenketten sind und somit auch Wörter darstellen können. Die Übertragung dieser Daten in solch ein BoW-Modell geschieht, indem erst einmal jeder einzigartige Token eine Identifikationsnummer erhält. Im nächsten Schritt wird jedes Dokument analysiert und die Häufigkeit der Tokens gezählt. Am Ende enthält jedes Dokument die Ids der enthaltenden Tokens und deren Häufigkeit.

Beispiel:

Dokument: Wenn Fliegen hinter Fliegen fliegen, fliegen Fliegen Fliegen nach. (Case Sensitive)

Token	Id	Häufigkeit
Wenn	1	1
Fliegen	2	4
hinter	3	1
fliegen	4	2
nach	5	1

Das Ergebnis dieses Dokuments sähe dann so aus: [(1,1), (2,4), (3,1), (4,2), (5,1)]. Übrigens: In Case Unsensitive wäre „fliegen“ sechs Mal vertreten. Mit solchen Listen können die Topic-Modeling Algorithmen umgehen.

3.3.2.7 Term frequency - Inverse document frequency

Term frequency - inverse document frequency kurz tf-idf ist ein statistisches Verfahren, um herauszufinden wie wichtig ein Token für ein bestimmtes Dokument in einer Sammlung von

Dokumenten ist. Ist ein Token in einem Dokument häufig vertreten aber in allen anderen Dokumenten nicht, so ist der Wert hoch. Wenn der Token in allen Dokumenten enthalten ist nähert sich der Wert null. Die Methode kann für das Topic-Modeling sinnvoll sein, weil Tokens, die in vielen Dokumenten vorkommen, wenig über das Topic eines speziellen Dokumentes aussagen.

3.4 Algorithmen / Modele

3.4.1 Latent Dirichlet Allocation

Das Latent Dirichlet Allokation (LDA) ist ein generatives Wahrscheinlichkeitsmodel und daher ist das Ziel von LDA ein Model zu finden, welches die höchste Wahrscheinlichkeit besitzt die Dokumente im Korpus erstellt zuhaben. Dieses ist ähnlich zu der Maximum likelihood Methode. Damit dieser Algorithmus funktioniert müssen einige Annahmen getroffen werden zum Beispiel, dass die Reihenfolge der Wörter keine Relevanz hat und das jedes Dokument aus mehreren Themen besteht und diese wiederum durch eine Wahrscheinlichkeitsverteilung beschrieben werden können. Der interne Prozess von LDA sieht wie folgt aus:

1. Wähle die Wortverteilung $\beta \sim Dir(\eta)$
2. Für jedes Dokument d in dem Textkorpus
    a. Wähle Anzahl der Token $N_d \sim Poisson(\xi)$
    b. Wähle eine Topic-Verteilung für ein Dokument $\theta \sim Dir(\alpha)$
    c. Für jeden Token w_{dn} der Anzahl der Token N_d
        i. Wähle ein Topic $z_{dn} \sim Multinomial(\theta)$
        ii. Wähle ein Token $w_{dn} \sim Multinomial(\beta_{z_{dn}})$

In (Chang et al. 2009) wurde LDA als die Methode eingestuft, welche die Topics am ähnlichsten einschätzt wie es Menschen tun würden.

3.4.1.1 Sequential latent Dirichlet Allocation

Dieser Algorithmus (kurz auch ldaseq genannt) basiert auf den „Latent Dirichlet Allocation“ mit dem Unterschied, dass dieser die Möglichkeit hat eine Zeitkomponente zu berücksichtigen. Dies ist insofern wichtig, wenn die Dokumente im Korpus Themen behandeln welche zeitspezifisch sind, wie Nachrichtentexte. LDASeq trainiert für jede angegebene Zeitperiode ein eigenes Model[3] (Blei et al 2006). Diese sind während des Trainings durch einige Parameter miteinander verbunden, so dass die gefunden Themen der vorherigen Zeitspanne in der nächsten Zeitspanne berücksichtig werden und wahrscheinlicher auftreten können.

3.4.1.2 Hierarchical Dirichlet Process (HDP)

Ist ein Mixture Model und eine Erweiterung von LDA. Das Besondere an HDP ist, dass es keinen Parameter für die Anzahl der Themen, die in dem Dokument vorhanden sein sollen, benötigt. Somit hebt sich dieser Algorithmus von den meisten anderen Topic-Modeling Algorithmen ab. Die von dem Algorithmus gefundenen Themen sind nicht hierarchisch angeordnet.

3.4.2 Non-negative Matrix Factorization (NMF)

Die generelle Idee des NMF-Algorithmus ist es, eine Wort-Dokument Matrix V (engl. “term-document”) in zwei Faktoren zu zerlegen. In einer Wort-Dokument Matrix ist in jeder Spalte ein Dokument gespeichert, während die Zeilen für jedes mögliche Wort stehen, das in den Dokumenten vorkommt. Die Werte in einer Spalte geben an, wie häufig ein Wort in einem Dokument vorkommt. Die Höhe des Werts kann auch als Rang des Wortes in dem Dokument interpretiert werden.

Die Wort-Dokument Matrix V soll in zwei kleinere Matrizen W und H durch Faktorisierung, d.h. $V = W\ x\ H$, zerlegt werden. W steht dabei für eine Matrix mit den Dimensionen Anzahl der Wörter im

[3] Unter einem „Hauptmodel“

Wörterbuch und der Anzahl von Archetypen. Die Archetypen stellen typische Dokumente dar, die ein Thema repräsentieren. D.h. in einem Dokument zum Thema Finanzen werden bspw. 5-mal das Wort „Aktien" erwartet und 3-mal das Wort „Euro". Die Anzahl der Archetypen ist typischerweise gering. Die Matrix H hat die Dimensionen Anzahl der Dokumente mal Anzahl der Archetypen. Diese Matrix beinhaltet die Gewichte der Linearkombination der verschiedenen Archetypen dar. Diese Linearkombination der Archetypen soll die originalen Dokumente möglichst gut repräsentieren bzw. ohne Reduzierungen völlig wiederherstellen. Je höher die Werte in H bezüglich eines bestimmten Archetyps, desto stärker fließt dieser Archetyp mit in die Reproduzierung des originalen Dokuments mit ein. Wenn das Originaldokument ein Artikel über ökologische Aktien ist, dann könnte „Finanzen" der höchstrangige Archetyp sein und „Umwelt" der zweitrangige.

Diese Überlegungen haben alle NMF Algorithmen gemein. Sie unterschieden sich lediglich in der Art und Weise wie diese Faktorisierung vorgenommen wird. Generell wird in diesen Algorithmen keine perfekte Faktorisierung vorgenommen, sondern eine Approximation. Dann ist das Ziel des Algorithmus die Differenz zwischen V und $W\ x\ H$ zu minimieren. Als Fehlerterm kann die quadratische Differenz oder auch die Kullback-Leibler Divergenz und viele weitere genommen werden (Zhao Tan 2016).

3.4.3 Latent Semantic Analysis (Latent semantic Indexing)

Latent Semantic Analysis (LSA/LSI) ist ein Algorithmus zur Zerlegung der Wort-Dokument-Matrix. Wenn die erste Spalte ein Dokument angibt, so steht in der Wortzeile die Anzahl der Worte, die in diesem Dokument vorkommen. Durch diese Zerlegung wird eine Vergleichbarkeit von verschiedenen Dokumenten und letztendlich auch ein Clustering möglich gemacht.

Die Existenz einer Wort-Dokument-Matrix X wird als gegeben angenommen. In einem ersten Schritt wird die Korrelation zwischen verschiedenen Wörtern berechnet, also der der Zeilenvektoren. Dies kann durch die Matrixmultiplikation XX^T erreicht werden. Parallel kann die Matrixmultiplikation X^TX zur Berechnung der Korrelation von Dokumenten bzw. Spaltenvektoren verwendet werden. Durch die SIngulärwertzerlegung kann die Matrix X in drei einzelne Matrizen zerlegt werden, also $X = U\Sigma V^T$. Hier sind U und V orthogonale Matrizen und Σ ist eine diagonale Matrix. Diese Singulärwertzerlegung kann genutzt werden, um die obigen Produkte umzuformen. Im Falle der Zeilenvektoren XX^T zu $XX^T = U\Sigma\Sigma^TU^T$, im Falle der Spaltenvektoren X^TX zu $X^TX = V\Sigma^T\Sigma V^T$. U sowie V enthalten die Eigenvektoren und das Produkt der Σ beinhaltet die quadrierten Eigenwerte. Die Korrelation der Dokumente basiert nur auf V und die der Worte nur auf U. Nur der i-te Zeilenvektor von U hat einen Einfluss auf den i-ten Zeilenvektor von X. Es ist möglich die Auswahl der Anzahl der Spalten- oder Zeilenvektoren auf die k höchsten Eigenwerte einzugrenzen.

Aufgrund obiger Überlegung können die Ergebnisse aus der reduzierten Dimension auch auf die höhere Dimension übertragen werden. Wenn nun $\hat{d}_i$ der Spaltenvektor in V ist, also der Vektor in reduzierter Dimension, so lassen sich nun die Vektoren $\Sigma_k\hat{d}_j$ und $\Sigma_k\hat{d}_q$ miteinander vergleichen und damit implizit auch die Dokumente in X. Als Distanzmaß wird üblicherweise die Kosinus-Ähnlichkeit verwendet. Die Möglichkeit Distanzmaße in geringeren Dimensionen zu berechnen, vereinfacht eine Clusteranalyse.

3.4.4 K-Means

K-means gehört zu den partitionierenden Clusteralgorithmen. Diese Art der Algorithmen benötigt neben den Daten auch die Anzahl der Cluster „K", die in den Daten vorhanden sein sollen und unterscheidet sich somit nicht von den expliziten Topic-Modeling Algorithmen. „means" bezieht sich auf die Clustercenter, welche die durchschnittliche Distanz der Objekte eines Cluster repräsentieren (Tsiptsis und Chorianopoulos 2010, S. 85). Aufgrund der Geschwindigkeit mit denen auch großen Datenmengen mit einer hohen Dimensionalität analysiert werden können, findet der Algorithmus

häufig Anwendung im Big-Data Bereich. Mittels des BagOfWords-Models, womit eine hohe Dimensionalität einhergeht und der tf-idf Methode ist es möglich diesen Algorithmus auch für das Topic-Modeling anzuwenden (Alhawarat, M. & Hegazi, M. 2018). K-Means ist ein Hardclusteringverfahren und weißt einem Dokument daher, als einziger Algorithmus in dieser Liste, nur ein Thema zu.

Der Algorithmus geht wie folgt vor:

Pseudocode:

1. Anzahl der Cluster k bestimmen und Dataset einlesen.
2. Zufälliges platzieren der Centroids $(C_1, C_2, C_3, \ldots, C_k)$
3. Wiederhole eingerückte Schritte bis zur Konvergenz oder bis eine vorgegebene Anzahl an Iterationen erreicht wurde.
    a. Für jedes Objekt x_o:
        i. Finde den nächsten Centroid $(C_1, C_2, C_3, \ldots, C_k)$
        ii. Weise das Objekt dem Cluster zu.
    b. Für jeden Cluster j = 1, 2, 3, …, k:
        i. Neuer Centroid = Durchschnitt aller Objekte innerhalb des Clusters.
4. Ende

Die Distanzberechnung im Schritt i erfolgt üblicherweise mit dem euklidischen Abstand (Tsiptsis und Chorianopoulos 2010, S. 85). Die optimale Clusteranzahl kann unter anderem mithilfe des Silhouettenkoeffizient ermittelt werden.

4 Praxis

4.1 Datenbeschaffung

Das Ziel war es die Pressemitteilungen der Bundesregierungen zu erheben. Die Erhebung erfolgte durch ein Scraper, welcher Mithilfe der Bibliotheken Scrapy und SplashRequest programmiert wurde. Für die Entwicklung des Bots sind zwei Arten von Unterseiten interessant, einmal die Übersichtseiten auf denen ca. zehn Pressemitteilungen pro Seite verlinkt sind und dann die Seiten für die Pressemitteilungen selbst. Die Übersichtsseiten können mit dem Query-Parameter am Ende der Adresse „?page=<int>“ aufgerufen werden. Um die Pressemitteilungsseiten aufzurufen ist es nötig die Links aus der Übersichtsseite zu extrahieren. Eine Extraktion erfolgt indem der Quellcode der Seite nachdem spezifischen Element durchsucht wird. Der Abschnitt im HTML-Code, der die Links beinhaltet, wird allerdings dynamisch durch Javascript generiert. Weil die Programmbibliothek Scrapy kein Javascript verarbeiten kann und somit nicht in der Lage ist auf die Links zuzugreifen muss SplashRequest eingebunden werden. Diese Bibliothek ruft die Seite innerhalb eines Docker-Containers auf und übermittelt die geforderten Daten an Scrapy weiter. Die Pressemitteilungsseiten sind statisch und die Daten wie der Titel, der Überblick, der Haupttext, das Datum und so weiter können daher mittels Scrapy erhoben werden.

Neben den dynamischen Inhalten war auch ein Problem, dass nicht alle benötigten Elemente der Seiten mit nur einem Selector (XPath, CSS) anzusprechen waren. Für einige Elemente war es nötig auf CSS zurückzugreifen. Außerdem musste eine Pause zwischen den Aufrufen/Requests eingerichtet werden, um nicht auf einen Fehler zu stoßen. Das allerdings erhöht stark die Zeit für die Datenerhebung.

Ablauf des Scrapers:

1. Für jede Übersichtsseite:
    a. Extrahiere die Links zu den Pressemitteilungen.
    b. Für jeden erhobenen Link:
        i. Rufe die Pressemitteilung auf.
        ii. Erhebe und speichere die Daten.
    c. Ermittle die maximale Anzahl an Übersichtsseiten.
    d. Wenn die aktuelle Seite nicht die letzte Seite ist dann gehe zur nächsten Übersichtsseite ansonsten stoppe.

4.2 Data-Understanding

Die Daten, mit denen die Algorithmen trainiert wurden, sind Nachrichten der Bundespressestelle. Die Themen der Nachrichten sind unterschiedlich. Sie handeln von Außenpolitik, Innenpolitik, Finanzpolitik aber auch über Migration und Kultur. Die Bundespressestelle hat 2009 angefangen Nachrichten auf dieser Website online zu stellen. Eine Nachricht besteht aus einem Titel, dem Autor, dem Datum der Veröffentlichung, einen Überblick und den Haupttext. Insgesamt wurden bis heute über 7500 Nachrichten veröffentlicht. Im Durchschnitt beinhaltet eine Nachricht 140 Wörter im Haupttext, dabei ist aber zu beachten, dass die Anzahl der Wörter von Jahr zu Jahr steigt. 2009 waren es gerade einmal 50 Wörter und 2020 sind es 233 Wörter pro Haupttext. Bei dem Betrachten des 25ten Quartil und des 50ten Quartil über die Jahre hinweg, fällt auf das in den ersten Jahren die Nachrichten meist nur aus drei Wörtern bestehen. Dabei handelt es sich um den Text: „Zur externen Meldung“ und ist ein Link auf eine andere Nachrichtenseite wie zum Beispiel der „Frankfurter Allgemeinen Zeitung“ kurz FAZ. Von diesen Nachrichten gibt es über 3500 Stück und sind in jedem Jahr zu ungefähr 50% vertreten. Einige von diesen besitzen neben den oben erwähnten Text noch einen Satz. Filtert man diese Mitteilungen heraus bleiben 4000 Nachrichten mit einem Haupttext und einer durchschnittlichen Wörteranzahl von 245. Außerdem wird klar, dass die Wörteranzahl der Nachrichten mit Haupttext über die Jahre nicht ansteigt. Bei der genaueren Betrachtung ist allerdings zu Erkennen das es neben den Nachrichten mit dem Text „Zur externen Meldung“ auch einige gibt mit gar keinem Text (Insgesamt nur 33). Übrigens die längste Nachricht wurde 2016 veröffentlicht und hat 5755 Wörter. Wie bereits erwähnt existiert neben dem Haupttext auch ein Überblick und zwar bei 5688 Nachrichten. Die durchschnittliche Anzahl an Wörtern beträgt 22. Betrachtet man die Anzahl der Nachrichten ohne einen Überblick über die Zeit fällt auf, dass diese Prozentual zu den gesamten Nachrichten im Jahr ansteigt, sprich es wird vermehrt auf den Überblick verzichtet. Im Jahr 2019 sind es 67,2% Nachrichten ohne Überblick. Weiterhin gibt es auch ein paar Nachrichten ohne Titel. Insgesamt sind es 97 und sind vorwiegend in den Jahren 2010 und 2011 vorhanden. Die Nachrichten werden von 21 verschiedenen Autoren veröffentlicht. Die Autoren sind alle staatlichen Institutionen wie Beispielsweise: ‚Presse- und Informationsamt der Bundesregierung (BPA)‘ oder das ‚Bundesministerium für Umwelt, Naturschutz, Bau und Reaktorsicherht‘. Dabei wundert es nicht, dass das ‚Presse- und Informationsamt der Bundesregierung‘ die meisten Nachrichten nämlich 3782 veröffentlicht hat. Erst mit großem Abstand folgt das ‚Auswertige Amt‘ mit 639 Nachrichten und dann mit weiterem Abstand ‚Wirtschaft und Energie‘ mit 359 Nachrichten. Danach folgen dicht andere Ämter.

Leider entfallen ungefähr die Hälfte an Daten aufgrund des nicht existierenden Haupttextes. Weiterhin schwankt die Anzahl an Wörtern von unter 50 Wörtern bis zu über 5000. Größere Texte stellen in der Regel kein Problem dar und sind eher hilfreich, weil wichtige bzw. prägnante Wörter häufiger vorkommen können. Das macht es Algorithmen einfacher Dokumenten einen Thema zuzuweisen. Bei kürzeren Texten sind die prägnanten Wörter seltener und das wiederum erschwert das Topic-Modeling.

4.3 Data-Preperation

4.3.1 Feature-Selection

Hauptsächlich sind drei Spalten bzw. Features für das Topic-Modeling interessant nämlich die Titel, der Überblick und der Haupttext. Während die Titel und der Überblick entweder vorhanden sind oder nicht, bedarf es bei dem Haupttext eine Datenbereinigung. Wie in der Datenanalyse angesprochen gibt es in dem Haupttext die Zeile ‚Zur externen Meldung' über 3500-mal und sieben Mal die Zeile ‚Zur externen Pressemitteilung', diese Meldungen sollten vorher aussortiert werden, falls nicht auf die Funktion ‚gensim.filter_extremes ' zurückgegriffen wird. Diese Funktion filtern häufig auftreten und sehr seltene Tokens. Außerdem beinhaltet der Haupttext Fotos und Links. Die Fotos wurden nicht explizit erhoben, aber dennoch sind einige Artefakte in Form von Zeilenumbrüchen und Links in den Haupttexten zu finden. Damit das Ergebnis nicht verfälscht wird, bietet sich an auch diese Artefakte herauszufiltern.

Durch die Zusammenlegung der drei genannten Features kann die große Menge an Datensätzen ohne nützlichen Haupttext dennoch verwendet werden, weil diese nun durch ihren Inhalt von dem Model berücksichtigt werden. Die Fusion der Features gelingt durch das simple aneinanderreihen der unterschiedlichen Texte zu einem Text.

4.3.2 Zusammenfassung

Die in dem theoretischen Abschnitt vorgestellten Methoden für die Data-Preparation wurden hier in unterschiedlichen Kombinationen angewandt sowohl auf den Haupttext, dem Haupttext + Überblick und dem Haupttext + Überblick + Titel. Es wurden aber stets die Methoden aus dem Kapitel „Entfernen von Zeichen" und die Stoppwörter rausgefiltert verwendet, weil diese immer einen positiven Einfluss haben. Eine Kombination kann zum Beispiel sein: „Entfernen von Zeichen" + „Entfernen von Token" (Stoppwörter und nur Nomen und Verben) + „N-Grams" (Trigrams mittels Gensim) + „Bilden von Wortstämmen" (Stemming) + BoW + tf-idf. Eine grafische 2D-Visualisierung der Daten sieht wie im rechten Plot aus (Abbildung 2: Koprus). Dieses Bild ist typisch für die 2D-Visualisierung von Textdaten. Die fehlenden Achsenbeschreibungen ist Resultat der Dimensionsreduktion mittels scikit-learns TruncatedSVD. Auf dem Bild sind keine richtigen Cluster zu erkennen und weil das Gezeigte relativ generisch ist, hilft die Visualisierung nicht bei Interpretation oder anderen Schritten.

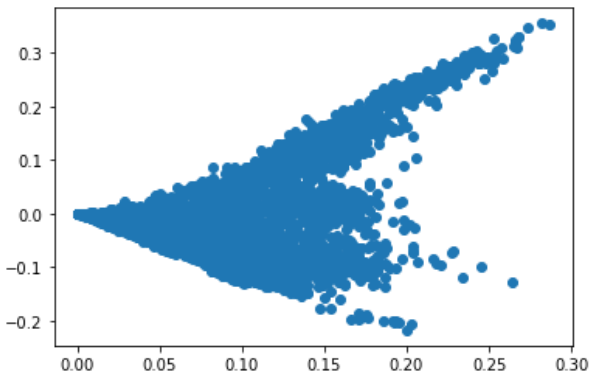


Abbildung 2: Koprus

Anmerkung: Das Filtern von Wörtern mit Umlauten bereitet in Spyder Probleme. So wird das Wort „für" nicht herausgefiltert.

4.3.2.1 Kommentar zu den Bibliotheken

Für die Data-Preparation wurde zum Entfernen von Stoppwörtern, Filtern nach Wortarten und das Bilden von Lemmata die Bibliothek „Spacy" eingesetzt. Diese Bibliothek erfreut sich einer hohen Beliebtheit und wird weiterentwickelt. Aber für dieses Projekt ist Spacy vor allem interessant, weil es die deutsche Sprache unterstützt und einfach einzubinden ist. Für die Lemmatization gibt es wie auch für die Filterung von Stoppwörtern andere Bibliotheken wie „GermaLemma" von Markus Konrad oder „Hanta" – Hanover Tagger von Christian Wartena. Die Methode für das verwendete Stemming heißt Cistem und ist Bestandteil des „Natural Language Toolkits" kurz nltk. Daneben gibt es weitere Stemming-Verfahren in der Bibliothek, allerdings wird von Weißweiler L. und Fraser A., dieser

Stemmer empfohlen. Das Erstellen von N-Grams kann von der Bibliothek „Scikit-Learn“ übernommen werden, bietet sich allerdings aufgrund der sehr beschränkten Auswahl an Parametern nicht an. Die Gensim Bibliothek kann mittels einer Pipeline ebenfalls N-Grams erstellen und ist dabei flexibler. Scikit-Learn bietet außerdem das Filtern von Stoppwörtern und andere Filtern in ihren Vectorizern (Count als auch Tf-IDf) an und ist durch einen einzigen Aufruf einfach zu bedienen. Allerdings raten die Entwickler den Einsatz des Stoppwörterfilters selbst ab. Daher muss mindestens eine weitere Bibliothek verwendet werden, falls einen die Funktionalität der Vectorizer genügt.

4.4 Modeling

Ebenso wie bei dem vorhergehenden Schritt der Data-Preperation wurden einige Modelle und Hyperparameter Kombinationen ausprobiert. Dabei ist die Anzahl an vorhandenen Parameter von Algorithmus zu Algorithmus unterschiedlich. In allen außer bei dem Hierarchical Dirichlet process-Algorithmus ist es notwendig die Anzahl von Themen vorzudefinieren.

4.4.1 LDA

Die Implementierung dieses Algorithmus findet sich unteranderem in Scikit-Learn als auch in Gensim wieder, die unterschiedlich performen. Der größte Unterschied zwischen diesen beiden ist die Updatefähigkeit der Gensim Implementierung d.h. der Algorithmus muss nicht erneut trainiert werden, sondern kann durch das Hinzufügen neuer Dokumente aktualisiert werden. Weiterhin ist der Umfang an Parametern, mit dem man den Algorithmus anpassen kann, geringer bei Scikit-Learn. Ein großer Vorteil dieser beiden Implementierung gegenüber den anderen ist die Möglichkeit auf die Bibliothek LDAviz zurückzugreifen. LDAviz ermöglicht es durch die Visualisierung einen schnellen Überblick über die einzelnen Themen/Cluster zu erhalten und deren Distanz zueinander. Weiterhin ist anzumerken das beide Bibliotheken für LDA multicore-Unterstützung bieten. Der Parameter mit dem größten Einfluss ist die Anzahl der Topics.

4.4.1.1 GensimLDA

Dieser Algorithmus stellt den Fokus dieses Projektes dar. Die meisten Kombinationen der Daten-Vorverarbeitung wurden mit diesem Model getestet und zum Teil auch gespeichert. Dieser Teil finden sich in dem resultRecordDict.Json wieder. Die gewonnen Resultate waren eher ernüchtern und keines davon konnte mit dem Perplexity-Score, Coherence-Score oder viel wichtiger der optischen Prüfung der Topics überzeugen. Um die optimale Anzahl der Topics zu erhalten wurde der Algorithmus mit aufsteigenden Werten für die Topics trainiert und der Coherence (u_mass) und der Perlexplexity-Score ermittelt:

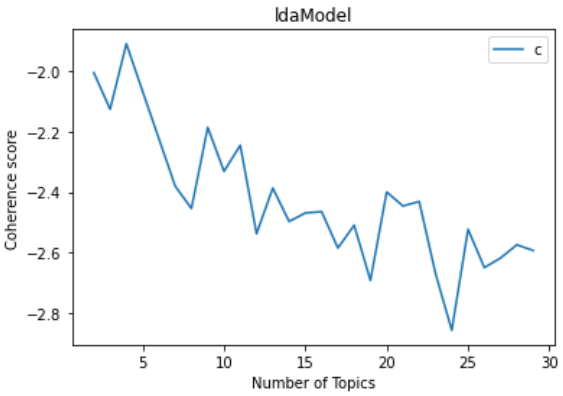


Abbildung 3: Coherence u_mass - TopicNr.

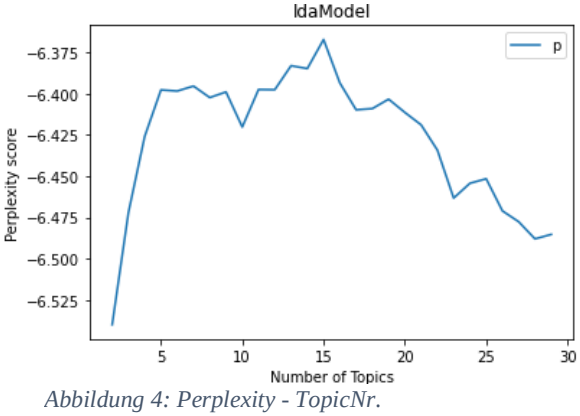


Abbildung 4: Perplexity - TopicNr.

Nachdem Coherence-Score müssten drei Topics gewählt werden, was allerdings in Anbetracht der Tatsache das deutlich mehr Themen im Korpus sein müssten falsch ist (Abbildung 3: Coherence u_mass - TopicNr.). Allerdings eignet sich die Coherence-Berechnung mittels u_mass nicht, um die Themenanzahl zu bestimmen. Aufgrund eines BrokenPipe-Errors war es nicht möglich die an verschiedenen Stellen empfohlene „c_v“ oder

eine andere Berechnung durchführen zu lassen[4]. Bei dem log Perplexity-Score wurde das Ergebnis bei 15 erzielt (Abbildung 4: Perplexity - TopicNr.). Die Differenzen sowohl bei der Perplexity als auch bei dem Coherence-Score sind sehr gering und spielen daher eine eher untergeordnete Rolle bei der Entscheidung. Weiterhin verändert sich der Kurvenverlauf bei erneutem Aufruf der Funktion und es könnte sein, dass die beste Wahl für die Anzahl der Topics ein anderes ist.

Im Anhang unter GensimLDA findet sich ein etwas überdurchschnittliches Resultat was mit Coherence-Score u_mass: -3,16 und c_v: 0,56 sowie einer Perlexity von -7,5 aufwartet. Es können einige Oberbegriffe zu den Topics gefunden werden wie Beispielsweise zu dem 0 Topic: Umweltschutz | 1 Topic: ‚Feiertage' | 2, 3, 4, 6 und evtl. 12 Topic: Außenpolitik | 5 Topic: Digitalisierung | 7 und 8 Topic: Migrations | 9, 10 und 14 Topic: Kultur | 11 Topic: Forschung | 13 Topic: Gesetzesentwurf/beschluss. Wie man hier erkennen kann ist noch sehr viel Potential nach oben, viele der Oberbegriffe passen nur sehr schwammig auf die Wörter in den Topics. Es gibt aber auch klar erkennbare Topics wie Topic 10 (nr. 6) wie in der Abbildung 5: LDAviz gezeigt.

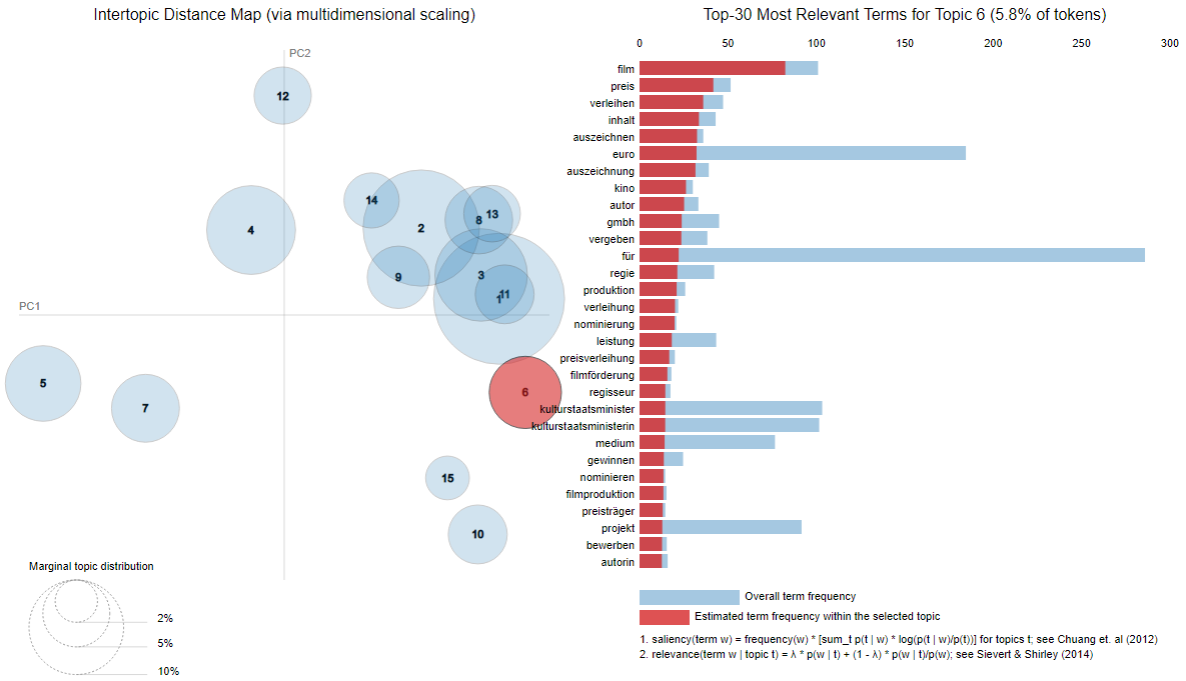


Abbildung 5: LDAviz – Dynamischer Plot

Jeder Kreis in der linken Abbildung ist ein Topic. Die Größe des Kreises gibt in Prozent an wie häufig dieses Thema im Korpus vorhanden ist. Die Reihenfolge ist auch zu erkennen an der Nummerierung. Der Kreis mit der Nummer 1 ist das Topic 14 also Kultur und ist damit am häufigsten im Korpus enthalten. Die Distanz zwischen den Kreisen gibt die Ähnlichkeit an. Je höher der Abstand desto unwahrscheinlicher treten diese beiden Topics zusammen auf. Der Kreis Nummer 5 ist das Topic 6 mit Außenpolitik und besitzt einen großen Abstand zu dem Topic 14 (Nr. 1). Überschneidungen sollten bestenfalls vermieden werden. In der rechten Abbildung sind die wichtigsten Wörter der Topics aufgelistet. Der blaue Balken zeigt die reale Häufigkeit an mit der ein Token in den Dokumenten enthalten ist und der rote Balken zeigt die von LDA geschätzte Häufigkeit. Im Kreis Nr. 6 kann LDA die Anzahl von dem Token ‚auszeichnen' sehr gut einschätzen. Dagegen ist die Schätzung von dem Token „euro" mangelhaft. Das Ergebnis ist dann gut, wenn die Häufigkeit der meisten Token annähernd richtig geschätzt wird. Dieses Beispiel Model eignet sich noch nicht für eine Verwendung,

[4] Update 15.06: Es ist ein Windows Multicore-Fehler und kann mit einer If-Anweisung „*if __name__ == "__main__":*" umgangen werden.

dazu gibt es zu viele Überlagerungen und die Themen und Oberbegriffe sind zu willkürlich. Soll heißen, wenn Außenpolitik enthalten ist, sollte im besten Falle auch Innenpolitik vorhanden sein.

4.4.1.2 Scikit-LearnLDA

Diese Implementierung bietet deutlich weniger Anpassungsmöglichkeiten, wie bereits erwähnt. Versucht wurde mittels GridSearch und dem Coherence-Score das beste Model zu finden. Leider waren die Resultate eher ernüchtern vor allem, weil sie Wörter beinhalteten, die nur in einer gewissen Zeitspanne auftraten wie Beispielsweise „Guido Westerwelle“. Es wurden keine Resultate gespeichert.

4.4.2 Sequential latent Dirichlet Allocation (LDAseq)

Dieser Algorithmus wurde ausprobiert, weil Pressemitteilungen, welche über Jahre veröffentlicht wurden, einen Zeitbezug haben. Dieser Zeitbezug war aber nicht in den Resultaten zu erkennen, da die gefunden Topics für jedes Jahr annähernd identisch waren[5]. Zwar ist es Bestandteil von LDAseq, dass die Topics der Vorperiode eine höhere Wahrscheinlichkeit besitzen auch in der nächsten Periode aufzutreten. Dieser geringe Unterschied und die relativ hohe Trainingszeit für diesen eigentlich kleinen Korpus führen aber zum Schluss, dass dieser Algorithmus nicht für diesen Korpus geeignet ist. Außerdem scheint die Implementierung in Gensim noch nicht vollständig abgeschossen zu sein. Die eher ernüchternden Ergebnisse wurden unteranderem aufgrund ihrer Größe und Struktur nicht in der Ergebnisdatei gespeichert.

4.4.3 Hierarchical Dirichlet Process

Die Anzahl der Topics muss hier zwar nicht vorgegeben werden allerdings spiegelt der Parameter ‚T‘ der den „Top level truncation level“ vorgibt die erstellten Topics wider. Dieses ist zu erkennen an den Vergleich des vordefinierten Parameter ‚T‘ und der Ausgabe durch den Befehl: model.get_topics().shape[0]. Der Coherence-Score (u_mass) liegt bei ca. -7,5 von den getesteten Kombinationen. Soll eine Benennung durch ein Oberbegriff erfolgen sind die gefunden Topics aufgrund der Vielzahl, wenn nicht durch ‚T‘ begrenzt, eher unhandlich. Ein weiterer Nachteil der vielen Topics ist die hohe Anzahl an Überschneidungen die vorkommen. Ein sinnvoller Einsatz dieses Algorithmus speziell für diese Daten ist nicht zu erkennen.

4.4.4 Non-negative Matrix Factorization

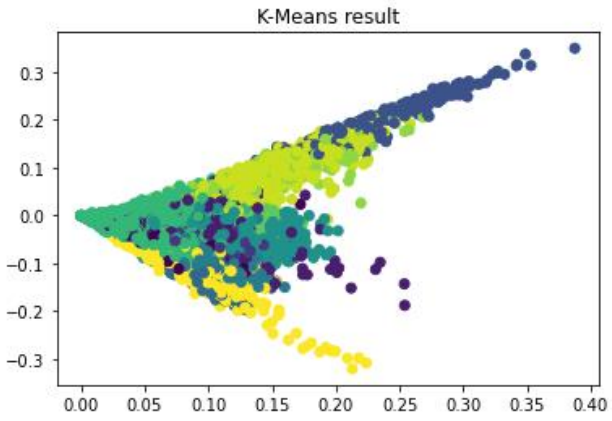
Die von dieser Methode gefunden Topics konnten nur zum Teil überzeugen (Beispiel Output im Anhang). Allerdings wurde dieser Algorithmus nur sehr grob behandelt. Mit einer ähnlichen Pipeline wie für die Gensim Algorithmen könne das Ergebnis eventuell verbessert werden.

4.4.5 Latent Semantic Analysis

Latent Semantic Analysis konnte einen Coherence-Score (u_mass) von -2,67 im Durschnitt erbringen. Die von dem Algorithmus zusammengefügten Wörter in den Topics ergaben zum Teil einen logischen Sinn. Im Anhang unter LSAGensim findet sich ein Beispiel dort ist zu erkennen, dass das Thema elf gut zur Außenpolitik passt sowie das Topic 7 zu Kunst & Kultur. Die anderen Topics sind schwerer einzuordnen. Neben der fehlenden Visualisierung, wie sie bei LDA existiert, gibt es auch nicht so viele Parameter, die denn Algorithmus beeinflussen. Das heißt die Data-Preparation hat einen weitaus größeren Einfluss auf den Algorithmus und auf dessen Performance.

[5] Ein Beispiel eines Topics pro Periode findet sich im Anhang unter GensimLDAseq wieder.

4.4.6 K-Means



Dieser für das Topic-Modeling eher atypische Algorithmus konnte in einem anderen Projekt gute Cluster bilden, weshalb der Einsatz auch hier sinnvoll erschien. Allerdings waren die generierten Topics unbrauchbar (siehe Beispiel im Anhang K-Means). Außerdem wurden Umlaute entfernt und ganze Wörter getrennt, obwohl der Encoder UTF8 als Default besitzt. Das Resultat bei 13 Clustern ist im rechten Bild zu sehen und zeigt das viele Cluster sich vermischen. Da allerdings die Topics aussageschwach sind ist jegliche Einteilung obsolet. Das Überprüfen mittels dem Silhouettenkoeffizient schlug fehl. Dieser kann aber anscheinend angewandt werden (Mantyla et al 2018).

5 Fazit

Die Gensim Implementierung von LDA hat nach dem Coherence-Score (u_mass) die besten Modelle hervorgebracht. Auch nach der Sichtung der Topics waren diese von allen anderen Algorithmen die besseren. Allerdings waren es nicht die Modelle mit dem geringsten Coherence-Score die überzeugen konnten, sondern eher solche Modelle wie das im Kapitel 4.4.1. besprochene, welches einen höheren Score aufweist aber dafür verständlichere Themen bietet. Die Auswahl an Parametern, die in einigen Fällen nur geringe Auswirkung haben, ermöglichen auch eine bessere Anpassung des Models an die jeweilige Problemstellung. Weiterhin sind die trainierten LDA-Model updatefähig, was bei einer großen Anzahl von Daten ein entscheidender Faktor sein kann. Die Updatefähigkeit wurde aber in diesem Projekt nicht getestet. Zusätzlich zu den Gensim internen Funktionen für LDA kann mithilfe der Bibliothek LDAviz sehr schnell ein Urteil über die Qualität der Topics getroffen werden. Daher wundert es nicht, dass diese Implementierung häufig Verwendung findet und den Standard im Bereich des Topic-Modeling darstellt. Allerdings ist generell davon abzuraten nur einen Algorithmus zu trainieren und zu validieren. Denn die Performance eines Algorithmus hängt, wie im Data-Science Bereich üblich, stark von den zur Verfügung stehenden Trainingsdaten und der Data-Preparation ab. So könnte, in einigen Fällen, zum Beispiel der LSA-Algorithmus, der in diesem Projekt etwas schlechter abschnitt sowohl vom Coherence-Score als auch von der Beurteilung der Topics, bessere Ergebnisse liefern. Andere Algorithmen wie LDAseq sind allein aufgrund der langen Trainingszeiten eher unattraktiv. Die Clusterverfahren wie K-Means und Non-negative Matrix Factorization sowie die LDA Implementierung von Scikit-Learn wurden in diesem Projekt sehr rudimentär ausprobiert und getestet. Aufgrund der fehlenden Implementierung der Pipeline für die Algorithmen wurden auch keine Parameter und Ergebnisse gespeichert außer einige Modelle[6]. Daher ist es schwer deren Güte für dieses Projekt konkret zu beurteilen. Generell sollte gesagt werden, dass die eher moderaten Ergebnisse und Topics unteranderem auch zurückzuführen sind auf die geringe Datenbasis.

Die Kombination von Entfernen von Stoppwörtern, Nummern, Satzzeichen und Computerzeichen sowie das Bilden von Tri-Gram und Lemma konnte am ehesten überzeugen (Auch nach den Coherence-Score u_mass) Die Stemming-Methode hinterließ, wie zu erwarten, auch Wörter die stark zerteilt waren. Allerdings ist dieser Faktor vor allem für die relevant, die jene Oberbegriffe erstellen

[6] Für einige dieser Modelle müsste auch der Coherence Score implementiert werden.

müssen. Die N-Gram Auswirkungen waren gering, allerdings gibt es Parameter, mit denen die Bildung erhöht werden kann.

Die in dem Projekt entwickelten Funktionen für die Data-Preparation mit Verbund der Bibliotheken NLTK, Spacy und Gensim sollte auch für andere Projekte nutzbar sein ebenso wie die ModelPipelines, welche die Data-Preparation-Funktionen beinhalten[7].

6 Ausblick

Dieses Projekt ist bei weitem nicht abgeschlossen und müsste in einigen Bereichen noch verbessert und vervollständigt werden. Zwar ist es zweifelhaft, ob die Algorithmen aus der Bibliothek Scikit-Learn bei vollständiger Implementierung besser abschneiden als der hier ausgiebig getestete LDA-Algorithmus. Dennoch sollte eine konkretere Überprüfung stattfinden. Eine Aktualisierung der Datenbasis steht ebenso aus, diese könnte auch noch zur Verbesserung der Ergebnisse führen. Aufgrund der moderaten Ergebnisse und der Zeit fehlt auch noch ein Programm, welches dann eine Pressemitteilung entgegennimmt und dann mittels eines Models die Topics dieser Mitteilung ermittelt. Dazu zählt dann auch die Validierung, ob die Topics tatsächlich in der Pressemitteilung enthalten sind, wie es ein Leser dieser Mitteilung erwarten würde[8]. Ist dieser Umstand gegeben kann das Model eingesetzt werden. Eine Überprüfung der Ergebnisse sollte regelmäßig stattfinden und wenn nötig muss der Prozess erneut beginnen.

7 Projektdokumentation

7.1 Data (Order)

In diesem Ordner befindet sich sowohl die Orginaldaten, die mittels Scrapy und SplashRequest erhoben wurden (press42.json) als auch die bearbeiteten Daten welche durch die Datei cleanBiTrigram.py entstanden sind. Der Unterordner „autoCreation“ gehört zu der Funktion „dataCleaningPipeline“ aus der Datei dataProcessHelper.py und speichert in diesem Ordner die von den „xPipeline.py“-Dateien erstellten Datenbasen.

7.2 Model (Ordner)

Dieser Order wurde zum Speichern der Modelle verwendet. Die Unterordner beinhalten die automatisch gespeicherten Modelle während die im Hauptordner durch stets überschriebene Versuche sind.

7.3 Result (Ordner)

In dem Ordner existiert die Datei „resultRecordDict.json“ und diese beinhaltet alle nachträglichen ausprobierten Kombinationen mit ihren Resultaten, wenn möglich mit Perplexity und Coherence-Score als auch alle nicht Default-Parametern. Außerdem findet sich der Json-Datei der Name der Datenbasen (data/autoCreateion) und der Modelle (model/{algorithmusName}).

7.4 Topic_Modeling (Ordner)

In dem Ordner befinden sich die Programmdateien von dem Scraper, welcher die Datenerhebung übernimmt. Wird ein Scrapy-Project initialisiert werden die Dateien angelegt und müssen nur noch angepasst werden. Daher sind nur zwei Dateien interessant einmal die Settings.py und die QuoteSpider.py im Unterordner Spiders. In der ersteren sind wie der Name es verspricht die Einstellungen zu dem Scraper zu finden. Die Einstellungen, welche verübt wurden, betreffen einzig

[7] Die Funktion „combineColumnStrings“ müsste nur angepasst werden, da dieser eine Filtermethode aufruft für speziell diese Datenbasis.

[8] Dieser Schritt ist nur sinnvoll, wenn die Ergebnisse des Models auch interpretierbar sind; sprich man Topics diesen Oberbegriffen wie ‚Finanzen', 'Migration' zuweisen kann.

das Hinzufügen von Einstellungen für die Bibliothek SplashRequest. Die Bibliothek ermöglicht das Erheben von dynamisch generierten Inhalten von Webseiten. Die Datei QuoteSpider.py enthält den eigentlichen Scraper. Der Programmablauf wurde in dem Kapitel 4.1 beschrieben. Gestartet werden kann der Scraper mittels der Konsole durch folgende Befehle:

```
docker pull scrapinghub/splash
docker run -p 8050:8050 scrapinghub/splash
scrapy crawl quotes -o press.json
```

Quotes ist der Name des Scrapers und -o press.json speichert die erhobenen Daten in eine Datei ‚press.json'.

7.5 CleanBiTrigram.py

Diese Datei beinhaltet eine Pipeline zum Erstellen verschiedener Textkorpera und speichert die Zwischenergebnisse. Das Laden dieser Ergebnisse kann durch die If-Anweisungen erfolgen, damit der Prozess nicht immer erneut beginnen muss, falls Anpassungen vorgenommen bzw. Kombinationen ausprobiert werden.

7.6 dataProcessHelper.py

DataProcessHelper dient mit seinen Funktionen als Hilfsmittel für die Data-Preperation-Schritte, zum Speichern und Laden von trainierten Modellen und für die Ausgabe von Topics für bestimmte Modele. Siehe Datei für die Dokumentation der Funktionen.

7.7 GensimHdpPipeline, GensimLdaPipeline, GensimLsiPipeline

Diese sehr ähnlich aufgebauten Dateien bilden die Pipeline für die Gensim-Algorithmen Hdp, Lda und Lsi. Im ersten Schritt ist es stets nötigt die folgenden Dictionaries SettingsDict, coprusDict und extraDict zu definieren, da diese Parameter für Funktionen beinhalten. Nachdem die Data-Preparation abgeschlossen wurde, wird das jeweilige Model trainiert. Danach erfolgt die Validierung mit dem Coherence-Score (u_mass und c_v) und wenn es sich um LDA handelt auch mit der Perplexity. Die Ergebnisse werden in einem weiteren Dictionary gespeichert und an das bestehende resultRecordDict.json angehängt. Das resultRecordDict.json beinhaltet die vorher gewonnen Ergebnisse und Parameter.

7.7.1 SettingsDict:

Das settingsDict enthält die Informationen über den ersten Schritt der Data-Preperation.

```
settingsDict = {'columnList': ['title', 'shortText', 'richText'],
                'gensimPreProcess': 'triStemm',
                'allowedTags': ['NOUN', 'VERB'],}
```

Mögliche Einträge mit Erklärung:
columnList: definiert die Spalten, welche zusammengefügt werden sollen.

gensimPreProcess: gibt der Funktion die Information, ob sie zum Beispiel n-Grams erstellen und ob die Wörter lemmatisieren soll.
Parameter:

- Clean -> Nur Filtern von Stoppwörtern, Zeichen, etc.
- Lemma -> Clean + Lemmatization
- Stemm -> Clean + Stemming
- biLemma -> Clean + Bi-Gram + Lemmatization
- biStemm -> Clean + Bi-Grams + Stemming
- TriLemma -> Clean + Tri-Grams + Lemmatization

- BiSstemm -> Clean + Tri-Grams + Stemming

allowedTags: definiert die Wortart, welche im Text verbleiben soll.
biPhrases: 5, 10 by default – Parameter für die Bildung für Bi-Grams: minCount und threshold.
triPhrases: 5, 10 by default – Parameter für die Bildung für Tri-Grams: minCount und threshold.
toLower: false by default – Parameter, ob der Text zu kleingeschrieben werden soll.

7.7.2 coprusDict

Enthält die Werte für die Gensim-Funktion: „Filter_extremes“

corpusDict = {'noBelow': 20, 'noAbove':0.8, 'keepN':1200}

Siehe Gensim-Dokumentation: Filter_extremes.

7.7.3 extraDict

Dieses Dictionary ist dazu gedacht verschiedene Parameter für verschiedene Funktionen zu beinhalten. Zum Beispiel ob Tf-IDF angewandt wurde.

7.8 hdpModelCreationGensim.py

Einfache Implementierung des Algorithmus „Hierarchical Dirichlet Process“, welche auf ein zuvor bereinigte Datenbasis zugreift. Gibt den Coherence-Score (u_mass) aus sowie die Anzahl der Topics als auch die Topics selbst. Modelle können wieder geladen werden.

7.9 kMeansModelCreationGensim.py

Einfache Implementierung des Algorithmus „k-Means“, welche auf ein zuvor bereinigte Datenbasis zugreift und dann mittels dem Tf-idf-Vectorizer für den Algorithmus zugänglich macht. Gibt die Topics aus und erstellt zwei Plots einen ohne Label und deren anderen mit Label. Gespeicherte Modelle können wieder geladen werden.

7.10 ldaModelCreationGensim.py

Einfache Implementierung des Algorithmus „Latent Dirichlet Allocation“, welche auf ein zuvor bereinigte Datenbasis zugreift. Gibt den Coherence-Score (c_v) und die Perplexity aus sowie die Topics. Das Laden von gespeicherten Modellen ist möglich. Kann außerdem drei Visualisierungen ausgeben für den Perplexity, Coherence-Score (u_mass, c_v) über die Anzahl der Topics.

7.11 ldaModelCreationSciKitLearn.py

War eine einfache Scikit-Learn LDA Implementierung und wurde dann zu einem Versuch Gridsearch zur Parametersuche zu verwenden. Ausgegeben werden die Perplexity, und die Topics.

7.12 ldaModelCreationYearGensim.py

Erstellt für jedes Jahr ein LDA Modell auf Basis von vorbearbeiteten Daten. Die Datei/Funktion ist obsolet, aufgrund der Tatsache das Gensim dafür ein Algorithmus anbietet und mit diesem nur ein Model entsteht. Unklar wäre, falls es genutzt werden sollte, wie die Modelle in ein Programm sinnvoll eingebracht werden.

7.13 ldaSeqModelCreationGensim.py

In der Datei befindet sich eine Pipeline mit dem LDASeq Algorithmus. Übergeben werden eine vorbearbeitete Datenbasis und die Anzahl an Dokumente für jede Periode. Ausgeben wird ein Topic für jede Periode. Andere Ausgaben sind mögliche siehe Gensim-Dokumentation.

7.14 lsiModelCreationGensim.py

Einfache Implementierung des Latent Semantic Analysis Algorithmus. Ausgegeben werden der Coherence-Score und die Topics sowie zwei Visualisierungen jeweils für die zwei Coherence-Scores (u_mass, c_v) über die Anzahl der Topics.

7.15 NmfModelCreationSciKitLearn.py

Scikit-Learn Nmf Implementierung mit Themenausgabe und deren Gewichte.

7.16 ResultAnalyse.ipynp

Diente zur Analyse des resultRecordsDict.json

7.17 TopicAnalyse.ipynp

Diente zur Analyse der Daten. Die gewonnen Informationen wurden im Kapitel Data-Understanding aufgezeigt.

7.18 topicVisualization.py

Beinhaltet zwei Funktionen, mit denen der Perplexity und die Coherence-Scores über die Anzahl der Topics visualisiert werden. Ursprünglich für mehr gedacht.

Quellen

Webcrawler

https://www.bigdata-insider.de/was-ist-ein-webcrawler-a-704217/

https://de.ryte.com/wiki/Crawler

https://www.myrasecurity.com/de/was-ist-web-scraping/?gclid=Cj0KCQjwz4z3BRCgARIsAES_OVfg07xLyZEJ_ptqvXL7Cr9xXglyZj3D3i5hQjMxKKknjs4WzX6iWlwaAj9pEALw_wcB

https://medium.com/@gajus/do-not-protect-your-website-from-scraping-part-1-technology-barriers-b0ced398d16d

Tf-idf

https://www.sistrix.de/frag-sistrix/inverse-document-frequency/

https://webvana.eu/wie-sie-tf-idf-fuer-seo-verwenden-koennen/

Richtige Quellen:

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993-1022.

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems* (pp. 288-296).

Brown, Meta S. (2015): What IT Needs To Know About The Data Mining Process. Forbes. Online verfügbar unter https://www.forbes.com/sites/metabrown/2015/07/29/what-it-needs-to-know-about-the-data-mining-process/#60bd9c10515f, Dateiname: What IT Needs To Know About The Data Mining Process

Blei, D. M., & Lafferty, J. D. (2006, June). Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning* (pp. 113-120).

Weißweiler, L. & Fraser, A. (2018, January). Developing a Stemmer for German Based on a Comparative Analysis of Publicly Available Stemmers

Alhawarat, M. & Hegazi, M. (2018, June). Revisiting K-Means and Topic Modeling, a Comparison Study to Cluster Arabic Documents

Röder, M., Both, A., Hinneburg, A. (2015 Febuary) Exploring the Space of Topic Coherence Measures

Mantyla, M. V., Claes, M., Farooq, U. (2018, October). Measuring LDA Topic Stability from Clusters of Replicated Runs

Witten, Ian H.; Frank, Eibe (2005): Data mining. Practical machine learning tools and techniques. 2nd ed. Amsterdam, Boston, MA: Morgan Kaufman

Ngai, E.W.T.; Xiu, Li; Chau, D.C.K. (2009): Application of data mining techniques in customer relationship management: A literature review and classification.

sEster, Martin; Sander, Jörg (2000): Knowledge Discovery in Databases. Techniken und Anwendungen. Berlin, Heidelberg, s.l.: Springer Berlin Heidelberg.

Smart Vision Europe Ltd. (Hg.) (2015): CRISP-DM by Smart Vision Europe. Smart Vision Europe Ltd. Online verfügbar unter http://crisp-dm.eu/

Tsiptsis, Konstantinos; Chorianopoulos, Antonios (2010): Data Mining Techniques in CRM. Inside Customer Segmentation. 1. Aufl. s.l.: Wiley.

Zhao, R., & Tan, V. Y. (2016). Online nonnegative matrix factorization with outliers. *IEEE Transactions on Signal Processing*, *65*(3), 555-570.

Anhang:

GensimLDA:

CoherenceCV Score: 0.5415027281788578

Coherence Score: -3.160895562605508

Perplexity: -7.520371536947622

Topic: 0
Words: 0.028*"starten" + 0.027*"klimaschutz" + 0.026*"energie" + 0.026*"forschung" + 0.025*"wirtschaft" + 0.021*"unternehmen" + 0.020*"bildung" + 0.019*"für" + 0.019*"ziel" + 0.018*"nachhaltigkeit"
Topic: 1
Words: 0.035*"senden" + 0.026*"einheit" + 0.020*"gedenken" + 0.018*"freiheit" + 0.017*"tod" + 0.017*"demokratie" + 0.016*"erinnerung" + 0.016*"gedenkstätte" + 0.015*"jahrestag" + 0.014*"erinnern"
Topic: 2
Words: 0.036*"gratulieren" + 0.031*"herr" + 0.027*"republik" + 0.024*"premierminister" + 0.022*"erfolg" + 0.022*"bundesrepublik" + 0.021*"grüßenangela" + 0.021*"freundlich" + 0.019*"opfer" + 0.017*"aufgabe"
Topic: 3
Words: 0.044*"bericht" + 0.025*"begrüßen" + 0.023*"staatssekretär" + 0.020*"entwurf" + 0.019*"regelung" + 0.018*"gesundheit" + 0.018*"beraten" + 0.018*"handel" + 0.017*"migrationshintergrund" + 0.017*"meldung"
Topic: 4
Words: 0.090*"außenminister" + 0.045*"verteidigungsminister" + 0.044*"minister" + 0.035*"amtskollegen" + 0.029*"verurteilen" + 0.028*"unterzeichnen" + 0.027*"nation" + 0.025*"gewalt" + 0.025*"gabriel" + 0.025*"fordern"
Topic: 5
Words: 0.025*"kind" + 0.014*"können" + 0.013*"video-podcast" + 0.011*"frage" + 0.011*"für" + 0.011*"kanzlerin" + 0.011*"internet" + 0.011*"mensch" + 0.010*"netz" + 0.010*"finden"
Topic: 6
Words: 0.052*"teilen" + 0.051*"telefonieren" + 0.051*"empfangen" + 0.041*"sprecher" + 0.033*"mittelpunkt" + 0.029*"beziehung" + 0.028*"republik" + 0.025*"präsidenten" + 0.023*"sprecherin" + 0.023*"präsident"
Topic: 7
Words: 0.100*"soldat" + 0.064*"hilfe" + 0.040*"million" + 0.037*"euro" + 0.033*"einsatz" + 0.023*"helfen" + 0.023*"stellen" + 0.023*"ausbildung" + 0.023*"flüchtlinge" + 0.019*"bevölkerung"
Topic: 8
Words: 0.014*"integration" + 0.011*"für" + 0.011*"migranten" + 0.008*"land" + 0.008*"staatsministerin" + 0.007*"müssen" + 0.007*"union" + 0.007*"hollande" + 0.007*"zusammenarbeit" + 0.006*"verhandlung"
Topic: 9
Words: 0.074*"betrieb" + 0.063*"verlag" + 0.060*"jugend" + 0.049*"staatsminister" + 0.045*"industrie" + 0.041*"urheberrecht" + 0.036*"vorstellung" + 0.034*"woche" + 0.034*"aufnehmen" + 0.031*"umwelt"
Topic: 10
Words: 0.052*"film" + 0.027*"preis" + 0.023*"verleihen" + 0.021*"inhalt" + 0.021*"auszeichnen" + 0.021*"euro" + 0.020*"auszeichnung" + 0.017*"kino" + 0.016*"autor" + 0.015*"gmbh"
Topic: 11
Words: 0.037*"dr." + 0.031*"frau" + 0.024*"technologie" + 0.024*"studie" + 0.023*"hersteller" + 0.023*"mitglied" + 0.022*"konzept" + 0.022*"professor" + 0.020*"kommission" + 0.019*"regie"
Topic: 12
Words: 0.048*"reisen" + 0.024*"reise" + 0.022*"teilnehmen" + 0.021*"treffen" + 0.019*"besuchen" + 0.019*"teilen" + 0.018*"juli" + 0.016*"besuch" + 0.016*"mai" + 0.015*"sprecher"
Topic: 13
Words: 0.061*"prozent" + 0.041*"beschließen" + 0.037*"milliarde" + 0.031*"steigen" + 0.031*"verbraucher" + 0.028*"landwirtschaft" + 0.026*"kabinett" + 0.025*"zahl" + 0.022*"gesetz" + 0.021*"kraft"
Topic: 14
Words: 0.017*"kulturstaatsminister" + 0.017*"million" + 0.016*"kultur" + 0.016*"kulturstaatsministerin" + 0.016*"euro" + 0.015*"bund" + 0.013*"stiftung" + 0.013*"für" + 0.011*"museum" + 0.011*"ausstellung"

GensimLDAseq:

Time: 0
Words: [('Zusammenarbeit', 0.021951651696167116), ('Seite', 0.011850988231414773), ('Entwicklung', 0.011767324533027248), ('Bereich', 0.0111382891267708), ('Umsetzung', 0.010908437144462057), ('Maßnahme', 0.010350217865791033), ('Ziel', 0.008970120294580658), ('Sicherheit', 0.008720417732906337), ('Nation', 0.008024057720835948), ('Staat', 0.00792154104065588)]
Time: 1
Words: [('Zusammenarbeit', 0.02358592500970 38), ('Seite', 0.01300626725580733), ('Entwicklung', 0.012112967090006988), ('Bereich', 0.01180574098771736), ('Umsetzung', 0.010828755379013379), ('Maßnahme', 0.009344542299349468), ('Ziel', 0.0091325767011177), ('Sicherheit', 0.008592799036704435), ('Nation', 0.00801190491282699), ('Staat', 0.007807763900793889)]
Time: 2
Words: [('Zusammenarbeit', 0.026808144074552627), ('Seite', 0.014793377702142521), ('Bereich', 0.012802804647645043), ('Entwicklung', 0.012390170476158462), ('Umsetzung', 0.010556852447273345), ('Ziel', 0.008795072637685364), ('Sicherheit', 0.008702666129805534), ('Maßnahme', 0.008663236396401102), ('Nation', 0.008027989996177168), ('Staat', 0.007990061203221611)]
Time: 3
Words: [('Zusammenarbeit', 0.027766247673635776), ('Seite', 0.017869094246763716), ('Bereich', 0.013144739001500844), ('Entwicklung', 0.012575960013795018), ('Umsetzung', 0.010207110424861306), ('Ziel', 0.00861946117377904), ('Staat', 0.00842648712011591), ('Regierung', 0.00834381721370032), ('Maßnahme', 0.008131704857874638), ('Nation', 0.008036840477408927)]
Time: 4
Words: [('Zusammenarbeit', 0.027911552485557663), ('Seite', 0.016058161619823364), ('Bereich', 0.01319563393455785), ('Entwicklung', 0.01300301018197678), ('Umsetzung', 0.009480555502754251), ('Regierung', 0.009470742666812828), ('Staat', 0.009155351895345887), ('Ziel', 0.008608671739890349), ('Rahmen', 0.008149470903680487), ('Vereinbarung', 0.008036689610448304)]
Time: 5
Words: [('Zusammenarbeit', 0.026657285916951847), ('Seite', 0.016016488886380187), ('Entwicklung', 0.013451622557576082), ('Bereich', 0.012363607521268642), ('Regierung', 0.010677324248649889), ('Staat', 0.010118050120643662), ('Umsetzung', 0.008696544003457311), ('Ziel', 0.008360510737204736), ('Rahmen', 0.00824825894332941), ('Nation', 0.008075234787516506)]
Time: 6
Words: [('Zusammenarbeit', 0.027162937619161484), ('Seite', 0.017761733338953402), ('Entwicklung', 0.013778315360093078), ('Regierung', 0.012347003182004929), ('Bereich', 0.011701719866918165), ('Staat', 0.010201356230005267), ('Rahmen', 0.008304604797422675), ('Nation', 0.008049247502572165), ('Maßnahme', 0.007713595226933667), ('Ziel', 0.00767859851389206)]
Time: 7
Words: [('Zusammenarbeit', 0.03211353410822323), ('Seite', 0.018879472838910505), ('Regierung', 0.014585467697866785), ('Entwicklung', 0.014094163343385643), ('Bereich', 0.012047284116883598), ('Staat', 0.009712909582059141), ('Rahmen', 0.008502598851992206), ('Nation', 0.007863735409311444), ('Maßnahme', 0.007784848744271782), ('Umsetzung', 0.007271586584504949)]
Time: 8
Words: [('Zusammenarbeit', 0.035296168081951636), ('Seite', 0.017729643155658916), ('Regierung', 0.01721409356543726), ('Entwicklung', 0.014512456154779859), ('Bereich', 0.012454768932853346), ('Staat', 0.009535234504165067), ('Rahmen', 0.008851073086133072), ('Nation', 0.007764841129157938 4), ('Maßnahme', 0.007737109113116002), ('Umsetzung', 0.007114065660992925 5)]
Time: 9
Words: [('Zusammenarbeit', 0.03199181294117686), ('Seite', 0.016088292811611404), ('Regierung', 0.01605417293384027), ('Entwicklung', 0.015274044321688676), ('Bereich', 0.012402422351970456), ('Staat', 0.009656162520583895), ('Rahmen', 0.008948300303070696), ('Nation', 0.007996484168019374), ('Maßnahme', 0.007728395768034443), ('Ziel', 0.007203860895162322)]
Time: 10
Words: [('Zusammenarbeit', 0.03199532490169648), ('Regierung', 0.015869136562337335), ('Entwicklung', 0.015784862597504733), ('Seite', 0.015428009870906369), ('Bereich', 0.012715063828215398), ('Staat', 0.009734051579343855), ('Rahmen', 0.008935034085875886), ('Nation', 0.008171201993256126), ('Maßnahme', 0.007577329031970633 4), ('Ziel', 0.007364919066272325)]
Time: 11
Words: [('Zusammenarbeit', 0.031912434265903455), ('Regierung', 0.016076644639629913), ('Entwicklung', 0.01581944365812005), ('Seite', 0.015488783075046936), ('Bereich', 0.012774120301888146), ('Staat', 0.00972770947645737), ('Rahmen', 0.008915986119678015), ('Nation', 0.008136092751745382), ('Maßnahme', 0.007574915020517458), ('Ziel', 0.007371220819308734)]

Non-negative Matrix Factorization:

Topic #1 with weights
[('bundeskanzlerin', 1.96), ('sprecher bundesregierung', 1.11), ('sprecher', 1.11), ('teilen', 0.96), ('bundesregierung teilen', 0.92), ('teilen bundeskanzlerin', 0.88), ('prasident', 0.68), ('bundesregierung', 0.65), ('prasidenten', 0.62), ('telefonieren', 0.52)]
Topic #2 with weights
[('kultur', 0.74), ('fur', 0.61), ('kulturstaatsministerin', 0.56), ('museum', 0.54), ('ausstellung', 0.52), ('stiftung', 0.51), ('kunst', 0.37), ('projekt', 0.36), ('medium', 0.35), ('geschichte', 0.34)]
Topic #3 with weights
[('grußenangela', 0.6), ('merkelbundeskanzlerin', 0.59), ('merkelbundeskanzlerin bundesrepublik', 0.59), ('grußenangela merkelbundeskanzlerin', 0.59), ('freundlich', 0.58), ('bundesrepublik', 0.56), ('freundlich grußenangela', 0.56), ('erfolg', 0.54), ('herr', 0.5), ('gratulieren', 0.42)]
Topic #4 with weights
[('regie', 1.69), ('gmbh', 0.9), ('verleihforderung', 0.5), ('gmbh regie', 0.49), ('medium', 0.46), ('antrage', 0.46), ('verleih', 0.39), ('verleih projekte', 0.39), ('telefon', 0.37), ('projekte', 0.34)]
Topic #5 with weights
[('opfer', 0.78), ('herr', 0.56), ('bundesrepublik', 0.54), ('angehorigen', 0.51), ('beileid', 0.51), ('erfahren', 0.49), ('mensch', 0.48), ('genesung', 0.47), ('grußangela', 0.46), ('herr prasident', 0.42)]
Topic #6 with weights
[('million', 1.15), ('euro', 1.14), ('million euro', 1.09), ('fur', 0.37), ('euro fur', 0.29), ('bund', 0.29), ('verfugung', 0.26), ('stellen', 0.22), ('milliarde', 0.17), ('milliarde euro', 0.17)]
Topic #7 with weights
[('meldung', 1.25), ('foto', 1.18), ('foto meldung', 1.04), ('außenminister', 0.77), ('alliance', 0.24), ('alliance meldung', 0.2), ('foto alliance', 0.2), ('gabriel', 0.17), ('treffen', 0.15), ('amtskollegen', 0.15)]
Topic #8 with weights
[('mittelpunkt', 0.78), ('teilen bundeskanzlerin', 0.71), ('bundeskanzleramt', 0.71), ('teilen', 0.7), ('beziehung', 0.69), ('bundesregierung teilen', 0.67), ('sprecherin bundesregierung', 0.66), ('sprecherin', 0.66), ('stehen', 0.59), ('mittelpunkt stehen', 0.53)]
Topic #9 with weights
[('euro inhalt', 1.08), ('inhalt', 0.95), ('euro', 0.55), ('autor', 0.46), ('film', 0.36), ('regisseur', 0.31), ('hersteller', 0.29), ('regisseur euro', 0.28), ('autorin', 0.26), ('gmbh', 0.24)]
Topic #10 with weights
[('dr', 1.22), ('professor', 0.73), ('professor dr', 0.54), ('kunst', 0.4), ('musik', 0.39), ('mitglied', 0.34), ('dr dr', 0.34), ('literatur', 0.28), ('akademie', 0.27), ('wissenschaft', 0.25)]
Topic #11 with weights
[('bundeskanzlerin', 0.7), ('podcast', 0.56), ('video podcast', 0.49), ('video', 0.49), ('internetadresse', 0.29), ('text finden', 0.29), ('internetadresse text', 0.29), ('text', 0.29), ('samstag', 0.28), ('samstag internetadresse', 0.28)]
Topic #12 with weights
[('fur', 0.88), ('wirtschaft', 0.43), ('entwicklung', 0.4), ('zusammenarbeit', 0.38), ('bundesregierung', 0.32), ('unternehmen', 0.27), ('fur wirtschaft', 0.24), ('eu', 0.23), ('ziel', 0.23), ('kommission', 0.22)]
Topic #13 with weights
[('film', 0.81), ('minute', 0.54), ('kurzfilmpreis', 0.52), ('laufzeit', 0.37), ('fur', 0.3), ('laufzeit minute', 0.23), ('auszeichnung', 0.22), ('fur film', 0.19), ('filmpreis', 0.19), ('kino', 0.17)]
Topic #14 with weights
[('kind', 0.8), ('integration', 0.66), ('migranten', 0.62), ('fur', 0.56), ('land', 0.35), ('fur kind', 0.34), ('schule', 0.3), ('jugendliche', 0.3), ('konnen', 0.27), ('staatsministerin', 0.26)]
Topic #15 with weights
[('premierminister', 1.41), ('bundeskanzlerin premierminister', 0.49), ('herr premierminister', 0.37), ('bundeskanzlerin', 0.24), ('premierminister republik', 0.18), ('ernennung premierminister', 0.13), ('premierminister ernennung', 0.12), ('zusammenarbeit', 0.1), ('ernennung', 0.1), ('premierminister telefonieren', 0.09)]

LSAGensim:

CoherenceCV Score: 0.44739575197816467

Coherence Score: -3.6225401778178994

Topic: 0
Words: 0.227*"für" + 0.193*"euro" + 0.176*"million" + 0.148*"republik" + 0.127*"stehen" + 0.120*"kultur" + 0.119*"bund" + 0.119*"kulturstaatsministerin" + 0.115*"beziehung" + 0.114*"teilen"
Topic: 1
Words: 0.297*"republik" + -0.236*"euro" + 0.230*"empfangen" + -0.211*"million" + 0.206*"teilen" + 0.205*"beziehung" + 0.201*"premierminister" + 0.188*"gratulieren" + 0.153*"sprecher" + 0.141*"ministerpräsidenten"
Topic: 2
Words: 0.320*"gratulieren" + -0.302*"empfangen" + -0.256*"teilen" + 0.229*"herr" + 0.202*"grüßenangela" + 0.202*"bundesrepublik" + 0.196*"freundlich" + 0.190*"erfolg" + -0.184*"mittelpunkt" + -0.182*"sprecher"
Topic: 3
Words: 0.352*"euro" + 0.298*"million" + 0.181*"republik" + -0.162*"mensch" + -0.150*"außenminister" + 0.148*"film" + 0.142*"empfangen" + 0.138*"gratulieren" + 0.123*"beziehung" + -0.118*"integration"
Topic: 4
Words: 0.305*"opfer" + 0.252*"kondolenztelegramm" + 0.216*"premierminister" + 0.193*"angehörigen" + 0.193*"genesung" + 0.187*"beileid" + 0.182*"euro" + 0.180*"erfahren" + -0.176*"gratulieren" + 0.168*"grußangela"
Topic: 5
Words: 0.357*"million" + -0.283*"film" + 0.278*"euro" + 0.254*"außenminister" + -0.202*"regie" + 0.190*"telefonieren" + 0.183*"hilfe" + -0.153*"kultur" + -0.151*"gmbh" + -0.147*"medium"
Topic: 6
Words: -0.361*"film" + -0.290*"telefonieren" + -0.258*"regie" + 0.229*"premierminister" + -0.215*"außenminister" + -0.195*"präsident" + -0.188*"gmbh" + -0.173*"inhalt" + -0.153*"präsidenten" + 0.149*"empfangen"
Topic: 7
Words: 0.300*"ausstellung" + -0.233*"euro" + -0.227*"premierminister" + 0.221*"museum" + 0.206*"kulturstaatsministerin" + -0.199*"film" + -0.195*"kind" + 0.182*"kunst" + -0.170*"prozent" + 0.169*"stiftung"
Topic: 8
Words: 0.723*"premierminister" + 0.296*"außenminister" + -0.254*"republik" + -0.171*"präsident" + -0.153*"kind" + -0.137*"präsident" + -0.119*"ministerpräsidenten" + -0.113*"präsidenten" + 0.105*"treffen" + -0.103*"prozent"
Topic: 9
Words: 0.582*"außenminister" + -0.367*"telefonieren" + -0.350*"premierminister" + 0.149*"republik" + 0.136*"hilfe" + -0.135*"präsident" + 0.115*"amtskollegen" + -0.114*"präsidenten" + -0.112*"prozent" + -0.105*"sprecher"
Topic: 10
Words: -0.492*"kind" + 0.265*"wirtschaft" + -0.236*"telefonieren" + -0.166*"integration" + -0.165*"migranten" + 0.155*"unternehmen" + 0.144*"entwicklung" + -0.140*"außenminister" + 0.137*"dr." + -0.133*"jugendliche"
Topic: 11
Words: 0.575*"soldat" + -0.323*"außenminister" + -0.206*"prozent" + 0.188*"reisen" + 0.146*"besuchen" + 0.135*"verteidigungsminister" + 0.128*"soldatinnen" + 0.128*"einsatz" + 0.121*"kanzlerin" + -0.118*"republik"
Topic: 12
Words: 0.526*"kind" + -0.279*"integration" + -0.228*"soldat" + -0.215*"migranten" + -0.210*"prozent" + 0.181*"netz" + 0.164*"wirtschaft" + -0.136*"land" + 0.129*"dr." + -0.117*"staatsministerin"
Topic: 13
Words: -0.480*"soldat" + -0.374*"prozent" + 0.214*"mensch" + -0.184*"kind" + 0.180*"video-podcast" + 0.159*"kanzlerin" + -0.146*"dr." + -0.142*"außenminister" + -0.134*"steigen" + -0.127*"verteidigungsminister"
Topic: 14
Words: -0.456*"reisen" + -0.324*"prozent" + 0.288*"soldat" + -0.185*"treffen" + 0.178*"republik" + 0.164*"empfangen" + -0.153*"juli" + -0.142*"teilnehmen" + -0.139*"ministerpräsident" + -0.124*"steigen"

k-means:

Cluster 0: energie kreativwirtschaft fur wirtschaft wirtschaft fur wirtschaft technologie bundesminister fur technologie bundesminister wirtschaft energie initiative kreativwirtschaft initiative bundesregierung erneuerbaren unternehmen

Cluster 1: fur eu kommission zusammenarbeit union staat regierung mitgliedstaaten sicherheit entwicklung bundesregierung seite maßnahme regierungschefs bundeskanzlerin

Cluster 2: dr professor fur professor dr mitglied nachhaltigkeitsstrategie bundesregierung entwicklung wirtschaft rechtsetzung burokratieabbau dr dr kunst sitzung musik

Cluster 3: teilen bundeskanzlerin teilen bundesregierung teilen bundeskanzlerin mittelpunkt beziehung bundeskanzleramt bundesregierung sprecher bundesregierung sprecher sprecherin bundesregierung sprecherin stehen frage mittelpunkt stehen

Cluster 4: regie euro inhalt film inhalt gmbh euro minute kurzfilmpreis autor hersteller fur medium telefon laufzeit kultur

Cluster 5: integration fur kind migranten land staatsministerin mensch fur kind konnen jugendliche schule vielfalt zuwanderer fur integration mussen

Cluster 6: bundesrepublik herr grußenangela merkelbundeskanzlerin merkelbundeskanzlerin bundesrepublik grußenangela merkelbundeskanzlerin freundlich erfolg freundlich grußenangela aufgabe gratulieren beziehung republik premierminister herr prasident

Cluster 7: meldung foto foto meldung außenminister fur alliance alliance meldung foto alliance bundesregierung mensch verteidigungsministerin prozent kulturstaatsministerin ziel euro

Cluster 8: fur bundesregierung außenminister stehen mensch konnen soldat thema erklarte prozent frage zeigen leben bundesaußenminister finden

Cluster 9: bundeskanzlerin podcast video podcast video internetadresse internetadresse text text finden text samstag internetadresse samstag fur podcast samstag kanzlerin merkel hinweis

Cluster 10: bundeskanzlerin prasident bundeskanzlerin prasident prasidenten teilen teilen bundeskanzlerin sprecher bundesregierung bundesregierung teilen sprecher telefonieren bundeskanzlerin prasidenten bundesregierung hollande vereinbarung umsetzung

Cluster 11: bundeskanzlerin teilen bundesregierung teilen teilen bundeskanzlerin sprecher bundesregierung sprecher bundesregierung premierminister ministerprasident thema ministerprasidenten reisen sprecherin sprecherin bundesregierung teilnehmen

Cluster 12: euro million million euro fur bund kultur stiftung kulturstaatsministerin museum projekt ausstellung stellen verfugung kulturstaatsminister euro fur