Universität zu Köln
Philosophische Fakultät
Institut für Digital Humanities

Masterarbeit zu dem Thema

Textanalyse mit dem „CollectionExplorer“. Eine Evaluation der Anwendbarkeit computerlinguistischer Methoden zur archivischen Bewertung großer unstrukturierter Textsammlungen

Zur Erlangung des Grades Master of Arts
von Nasrin Saef

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Noch nie war das Anlegen, Verwerfen, Umsortieren und Kopieren von Dokumenten so einfach wie heute. Ein Computer mit Dateiablage und Textverarbeitungsprogramm reicht, wo lange Zeit Papier, Schreibgeräte, Ordner und Regale von Nöten waren. Dokumente wie Rechnungen, Kontoauszüge oder Steuerbescheide werden in digitaler Form verschickt und abgelegt. Briefe werden durch E-Mails ersetzt oder am PC getippt und ausgedruckt. Kurzum: Die Verwaltung von Schriftgut findet in großem Ausmaß digital statt. Das gilt nicht nur für Privatpersonen, die ihre Briefe an den Vermieter oder den Punktestand der wöchentlichen Kegelrunde digital festhalten, auch die öffentliche Verwaltung nutzt Textverarbeitungsprogramme, E-Mails und abteilungsinterne Dateiablagen zur Verwaltung ihrer alltäglichen Geschäfte.

Für spezialisierte Aufgaben existieren dezidierte Softwareanwendungen, und für die Führung aktenrelevanter Unterlagen werden elektronische Dokumentenmanagementsysteme eingeführt. Doch außerhalb dieser spezialisierten und kontrollierten Systeme florieren Ordnersysteme voller Dateien, angelegt von Sachbearbeitern ohne eine Ausbildung in Schriftgutverwaltung.[1] Solche Ablagen wachsen häufig über Jahren und werden dabei groß, unübersichtlich und individuell. Trotz ihrer ungewöhnlichen Form handelt es sich bei ihnen aber um Verwaltungsschriftgut, welches den deutschen Archivgesetzen unterliegt. Archivare müssen also darüber entscheiden, ob Bestandteile der Ablagen auf Dauer im Archiv aufbewahrt und zugänglich gemacht werden sollen. Dafür ist ein Überblick über die in ihnen enthaltenen Dateien und deren Inhalte nötig – eine große Herausforderung, wenn es um tausende bis zehntausende Dokumente geht und keine offensichtliche Struktur zu erkennen ist. Herkömmliche archivische Methoden verlassen sich für das Schaffen dieses Überblicks auf die Gegenwart von Aktenplänen oder ähnlichen strukturierenden Elementen, die es für die digitalen Ablagen in der Regel nicht gibt – und sind somit nicht mehr ohne Weiteres anwendbar.

Vorliegende Arbeit versucht deshalb, einen anderen methodischen Ansatz für die Erfassung des Inhalts solcher Dateisammlungen zu finden. Statt im Vorfeld erstellter

[1] Kenntnisse der Schriftgutverwaltung seitens Verwaltungsmitarbeitern sind seit dem Verschwinden der Berufsbilder des Sekretärs und des Registrators stark rückgängig. Vgl. Berwinkel, Holger: Zur Kanzleigeschichte des 20. Jahrhunderts. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde: *Moderne Aktenkunde*. Marburg 2016.

Übersichten sollen die Volltexte der in Dateiablagen enthaltenen Dokumente herangezogen werden, um ihre Struktur und die Aussagekraft der Dokumente einzuschätzen. Die großen, unstrukturierten Dokumentsammlungen sollen mit Hilfe von Methoden aus den Bereichen Computerlinguistik, Information Retrieval und Text Mining verarbeitet und versucht werden, sie inhaltlich greifbar zu machen. Ziel der Textanalyse ist es, eine Vorstellung vom Inhalt der Ablage zu vermitteln, Vorschläge zu ihrer Strukturierung zu machen und potentiell sensible Informationen wie Personennamen zu finden. So soll Archivaren die Entscheidung über die Archivwürdigkeit von Dokumenten aus Dateiablagen erleichtert werden.

Mit der Problematik der Bewertung digitaler Dateiablagen werden Archive erst seit vergleichsweise kurzer Zeit konfrontiert. Schriftgut wird erst dann dem zuständigen Archiv angeboten, wenn es im laufenden Betrieb nicht mehr benötigt wird und alle Aufbewahrungsfristen abgelaufen sind, und digitale Dateiablagen sind im großen Stil erst seit Mitte der 1990er Jahre entstanden.[2] Bei ihnen stellt sich außerdem die Frage, wann die Dateien als geschlossen und somit anbietungspflichtig gelten.[3] Zudem muss vor der Anbietung ans Archiv die Infrastruktur für die Übernahme digitaler Objekte vorhanden sein. Lange herrschte in Archiven außerdem die Ansicht, dass alle relevanten Unterlagen zu den Akten gegeben würden und in Dateiablagen ergo nur unwichtige oder redundante Dokumente zu finden wären.[4]

In der archivarischen Diskussion wird die Bewertung von Dateiablagen dementsprechend erst seit Kurzem behandelt. Auf dem Deutschen Archivtag 2015 behandelte ein Vortrag Dateiablagen als archivische Quellen und besprach auch explizit ihre Bewertung, beschrieb aber vor allem die dabei auftretenden Schwierigkeiten: Die Ablagen seien häufig groß, unstrukturiert und unübersichtlich und entzögen sich herkömmlichen Bewertungsmethoden.[5] Im Jahr darauf erschien der von der Archivschule Marburg herausgegebene Band „Moderne Aktenkunde“, welcher

[2] Vgl. Naumann, Kai: Dateisammlungen.
[3] Vgl. Miegel, Annekathrin und Eva Rödel: Wege aus dem Daten-Dschungel – Bewertung und Übernahme großer Dateisammlungen. In Klara Deecke, Ewald Grothe: *Massenakten - Massendaten*. Fulda 2018, S. 30.
[4] Vgl. Miegel, Annekathrin, Sigrid Schieber und Christoph Schmidt: Vom richtigen Umgang mit kreativen digitalen Ablagen. In Michael Puchta, Kai Naumann: *Kreative digitale Ablagen und die Archive*. München 2017, S. 7.
[5] Vgl. Wendt, Gunnar und Sina Westphal: Eine Herausforderung des Übergangs. In Monika Storm: *Transformation ins Digitale*. Fulda 2017.

neben analogem Schriftgut des 20. Jahrhunderts auch beispielsweise E-Mails, Datenbanken und Dateiablagen aktenkundlich untersucht.[6] Darin erfolgte die erste ausführliche Charakterisierung von Dateiablagen durch einen (schon länger als die Verwaltungsarchivare mit diesem Quellentypus konfrontierten) Wirtschaftsarchivar. Ebenfalls 2016 fand der Workshop „Kreative digitale Ablagen und die Archive“ der Staatlichen Archive Bayerns statt.[7] Dieser befasste sich ausdrücklich mit Dateiablagen und stellte Strategien und Tools zu ihrer Handhabung ins Zentrum. Allerdings beschränkten sich die vorgestellten Ansätze weitgehend darauf, mit Hilfe von Formatanalysen und Duplikaterkennung die im Anschluss manuell durchzusehenden Bestände zu reduzieren. In der Fachzeitschrift „Der Archivar“ findet sich der erste Beitrag zu dem Thema im Heft vom Juli 2017 (stattdessen wurden in Heften zur digitalen Archivierung beispielweise die Digitalisierung analoger Dokumente oder die Langzeitarchivierung von Daten aus sogenannten Fachverfahren[8] behandelt).[9] Im gleichen Jahr befasste sich ein Vortrag auf dem Deutschen Archivtag mit der Bewertung großer Dateiablagen.[10] Darin beschrieben Miegel und Rödel die Erfahrungen des Hessischen Hauptstaatsarchivs Wiesbaden mit der Bewertung und Übernahme digitaler Dateiablagen und kamen zu dem Schluss, dass Softwarelösungen benötigt werden, um die inhaltliche Bewertung von Dateiablagen effizient durchzuführen.[11]

In mehreren Beiträgen wird der Ansatz verfolgt, die Gesamtmenge der Dateiablage im Vorfeld durch den Ausschluss nicht archivfähiger Dateien[12] sowie das Filtern von Duplikaten zu reduzieren. Danach soll auf Ordnerebene bewertet werden.[13] Eine Strategie zur Behandlung dieser voraussichtlich in Zukunft immer häufiger auftretenden Quellen, die die technischen Möglichkeiten der letzten Jahre ausschöpft,

---

[6] Vgl. Berwinkel, Holger, Robert Kretzschmar und Karsten Uhde: Moderne Aktenkunde. Marburg 2016.
[7] Vgl. Puchta, Michael und Kai Naumann: Kreative digitale Ablagen und die Archive. München 2017.
[8] Der Begriff „Fachverfahren“ bezeichnet von Behörden eingesetzte Software, die für die Erledigung spezialisierter Aufgaben entwickelt wurde. Vgl. Birn, Marco: Fachverfahren - Terminologie der Archivwissenschaft.
[9] Vgl. Jaeger, Karina und Maria Kobold: Zwischen Datenwust und arbeitsökonomischer Bewertung. In *Der Archivar*.
[10] Vgl. Miegel und Rödel.
[11] Vgl. ebd., S. 35.
[12] Die Archivfähigkeit einer Unterlage beschreibt, ob sie im Archiv dauerhaft erhalten und zugänglich gemacht werden kann.
[13] Diesen Ansatz beschreiben zum Beispiel Miegel et al., Belovari sowie Jaeger und Kobold.

schlägt aber keiner der Beiträge vor; stattdessen wird an den Methoden für die Bewertung von Papierakten festgehalten. Daher wird in vorliegender Arbeit versucht, mit Hilfe computerlinguistischer Methoden einen Ansatz zum Umgang mit Dateiablagen in Archiven zu entwickeln. Sie wird von einem Softwareprojekt begleitet, welches ausgewählte Methoden implementiert und ihre Nützlichkeit für das vorliegende Problem anhand von Testdaten evaluiert: dem Hessischen Hauptstaatsarchiv Wiesbaden angebotenen Dateiablagen unterschiedlicher Registraturbildner.

Der Einsatz computerlinguistischer Methoden wäre auch an anderen Stellen des archivischen Arbeitsprozesses denkbar. Hier wird die Bewertung betrachtet, da diese den Grundstein für die weitere archivische Arbeit legt; ohne sie gäbe es keine oder zu viele Archivalien für die nachfolgenden Schritte. Zudem wird hier der größte Gewinn erwartet: Denn das Treffen einer Entscheidung über den bleibenden Wert von Unterlagen wäre kaum möglich, ohne Inhalt und Struktur der Dateien zu kennen und sie in ihren Kontext einzuordnen.

Um sich dem Problem anzunähern, wird in Kapitel 2 zunächst sein archivischer Kontext erläutert. Ein Versuch der quellenkundlichen Charakterisierung von Dateiablagen wird unternommen und das Problem der archivischen Bewertung umrissen. Zu diesem Zweck werden sowohl die Gesetzesgrundlage und die Bedeutung der Bewertung im archivischen Arbeitsprozess als auch gängige Bewertungsmethoden beschrieben. Zuletzt werden die Schwierigkeiten bei der Bewertung von Dateiablagen mit herkömmlichen Mitteln erläutert.

Kapitel 3 ist den computerlinguistischen Methoden gewidmet, die zur Analyse der Dateiablagen eingesetzt werden sollen. Es beschreibt zu Anfang notwendige Vorverarbeitungsschritte, danach werden die implementierten Verfahren in ihren Grundzügen erläutert und der Versuch einer Einschätzung unternommen, welche Informationen aus ihnen gewonnen werden könnten und welche Probleme bei der Umsetzung zu erwarten sind.

Die Softwareanwendung, in der die oben beschriebenen Methoden umgesetzt werden, wird in Kapitel 4 vorgestellt. Es werden zunächst die technische Architektur und das Datenmodell beschrieben und dann Workflows für den Upload von Dateien, die Vorbereitung der Datenanalysen und ihre Durchführung präsentiert.

Die Anwendung selbst wird in Kapitel 5 evaluiert. Sie wird zunächst kritisch auf Benutzbarkeit überprüft, dann werden die Ergebnisse der Verarbeitung und Analyse von insgesamt fünf Testbeständen beschrieben. Zum Abschluss wird evaluiert, welche Methoden auf welche Arten von Beständen anwendbar sind und welche Erkenntnisse aus ihnen generiert werden können.

Kapitel 6 beleuchtet die Potentiale der angewandten Methoden für andere archivische Arbeitsfelder wie die Erschließung und Nutzerrecherchen, bevor in Kapitel 7 die Ergebnisse zusammengefasst werden und ein Fazit gezogen wird.

2 Archivwissenschaftlicher Kontext

Das folgende Kapitel wird die Problemstellung dieser Arbeit archivwissenschaftlich einordnen. Dafür werden zunächst die behandelten Quellen, also digitale Dateiablagen, aktenkundlich betrachtet. Da das in vorliegender Arbeit behandelte Problem im Kontext der archivischen Bewertung, also der Auswahl archivwürdiger Unterlagen, auftritt, werden danach Sinn und Zweck der Bewertung sowie grundlegende Vorgehensweisen umrissen.[14] Zum Abschluss wird erläutert, wo die Schwierigkeiten bei der Bewertung von Dateiablagen liegen.

2.1 Aktenkundliche Charakterisierung von Dateiablagen

Dateiablagen finden sich wohl seit Beginn der Nutzung von Personal Computern auf jeder Festplatte. Als archivische Quelle sind sie aber vergleichsweise neu, etablierte Methoden zu ihrer Handhabung fehlen noch.

Eine erste Beschreibung von Dateiablagen und den mit ihrer Bewertung einhergehenden Problemen in der deutschen archivwissenschaftlichen Diskussion haben Wendt und Westphal auf dem Deutschen Archivtag 2015 vorgenommen. In ihrem Beitrag stellen sie fest, dass Dokumentenmanagementsysteme noch lange nicht flächendeckend in der Verwaltung eingeführt wurden, Dateiablagen dagegen sehr verbreitet seien, und dass in ihnen aktenrelevante Informationen enthalten sein können.[15] Sie charakterisieren Dateiablagen als „gemeinsame

[14] Dabei wird nur auf die absolut notwendigen Grundlagen eingegangen. Eine Überblicksdarstellung, die sowohl auf die Geschichte der deutschen Bewertungsdiskussion als auch auf aktuell verwendete Methoden im Detail eingeht, findet sich zum Beispiel bei Buchholz. Vgl. Buchholz, Matthias: Archivische Überlieferungsbildung im Spiegel von Bewertungsdiskussion und Repräsentativität. Köln 2011.

[15] Vgl. Wendt und Westphal, S. 105 f.

Sachbearbeiterablagen“[16] und beschreiben, dass die gemeinsame Verwaltung durch mehrere Bearbeiter über lange Zeiträume hinweg häufig zu sehr unübersichtlichen Strukturen führe.[17] Ihr Fazit ist verheerend:

> „Aus Sicht des Archivars nimmt dies nicht selten Formen an, welche die vorgefundene Ordnung als geradezu archivfeindlich und als einer sinnvollen Nutzung nicht zuführbar erscheinen lassen.“[18]

Die typische Struktur solcher Ablagen beschreiben sie als Sammlung von „[…] mehrere[n] tausend Dateien in zahlreichen Ordnern mit beliebig vielen Unterordnern […]“[19] und konstatieren, dass dies den inhaltlichen Zugriff auf die darin enthaltenen Dateien sehr schwierig mache.[20]

Eine ausführlichere Charakterisierung von Dateiablagen wurde 2016 im Band „Moderne Aktenkunde“ vorgenommen. Dort weist Kretschmar darauf hin, dass klassische Aktenkunde auf digitale Unterlagen schwer anzuwenden sei, da ihnen die klare Struktur von Papierakten fehle.[21] Die intensivste Auseinandersetzung mit Dateiablagen nimmt folgerichtig nicht ein Staats-, sondern ein Wirtschaftsarchivar vor. Schludi beschreibt zunächst, inwiefern sich Dateiablagen von Akten unterscheiden. Freie Hand bei der Struktur des Ordnersystems und hohe Verfügbarkeit von Speicherplatz führen dazu, dass die durchdachte Ordnung des Schriftguts entfällt, da sie nicht mehr nötig scheine.[22] Dazu trage auch die Hoffnung bei, über die Suchfunktion der Computer alle Dateien wiederzufinden.[23] Auch Aktenzeichen oder andere strukturgebende Metadaten werden selten vergeben, der Zusammenhang zwischen Dateien bestehe einzig aus der Ablage im gleichen Ordner.[24] Weiterhin seien diese Ablagen und Ordnersysteme sehr individuell: Berechtigungen im Dateisystem verhindern eine gemeinsame, abteilungsübergreifende Schriftgutverwaltung.[25] Während Papierdokumente nach Abschluss eines Vorgangs zu Akten zusammengefasst und gesondert gelagert werden, verbleiben elektronische

---

[16] Vgl. Wendt und Westphal, S. 106.
[17] Vgl. ebd.
[18] Ebd.
[19] Vgl. ebd., S. 107.
[20] Vgl. ebd.
[21] Vgl. Kretzschmar, Robert: „Akten“. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde: *Moderne Aktenkunde*. Marburg 2016, S. 17.
[22] Vgl. Schludi, Ulrich: Das Schriftgut der Wirtschaft. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde: *Moderne Aktenkunde*. Marburg 2016, S. 97 f.
[23] Vgl. ebd., S. 96.
[24] Vgl. ebd., S. 99.
[25] Vgl. ebd.

Dokumente in der Regel an ihrem Lagerort im Dateisystem.[26] Der Prozess, der zum Abfassen der Dokumente führt, sei schlecht nachvollziehbar, da Anweisungen oft mündlich gegeben werden und die auf Papierakten üblicherweise aufgebrachten Vermerke und Verfügungen fehlen.[27] Auch die Entstehungsstufen verschwimmen. Es gebe keine Unterscheidung zwischen Konzept und Reinschrift, frühe Versionen gehen oft verloren, da sie nicht separat gespeichert werden, und es sei schwer zu ermitteln, in welchem Stadium sich ein Dokument befindet.[28] Miegel und Rödel ergänzen, dass in Dateiablagen häufig mehrere Backups von unterschiedlichen Zeitpunkten enthalten seien und so Parallelüberlieferungen mit Dubletten oder Versionen der gleichen Datei entstehen.[29] Trotz all dieser Widrigkeiten ist eine Beschäftigung mit Dateiablagen für Archivare alternativlos: Miegel und Rödel stellen fest, dass diese in einigen Behörden „die analoge Akte (...) schlicht abgelöst haben"[30].

Dateiablagen sind also wesentlich unstrukturierter und heterogener als Papierakten. Ihr Entstehungsprozess ist schwierig nachvollziehbar, sie werden individuell geführt, und sie liegen häufig in sehr großen Mengen vor. All das macht sie zu einer Herausforderung für Archive, die normalerweise mit über einen Aktenplan strukturierten, physisch in ihrem Umfang und ihrer Anzahl begrenzten Papierakten arbeiten.

2.2 Grundlagen archivischer Bewertung

Zur besseren Einordnung der Problemstellung wird in den folgenden Abschnitten ein Überblick über die archivische Bewertung gegeben. Ihre Rolle in der archivarischen Arbeit wird dargelegt und die wichtigsten Bewertungsmethoden in ihren Grundzügen vorgestellt.

2.2.1 Gesetzliche Grundlagen und Zweck der archivischen Bewertung

Zweck und Rolle der archivischen Bewertung sollen nachfolgend am Bundesarchivgesetz verdeutlicht werden.[31] Laut § 3 Bundesarchivgesetz[32] ist es

[26] Vgl. Schludi, S. 96.
[27] Vgl. ebd., S. 101 f.
[28] Vgl. ebd., S. 103.
[29] Vgl. Miegel und Rödel, S. 35.
[30] Ebd., S. 29.
[31] In den Archivgesetzen der Bundesländer existieren entsprechende Vorschriften.
[32] Vgl. Deutscher Bundestag: Gesetz über die Nutzung und Sicherung von Archivgut des Bundes (Bundesarchivgesetz).

Aufgabe des Bundesarchivs „[…] das Archivgut des Bundes auf Dauer zu sichern, nutzbar zu machen und wissenschaftlich zu verwerten“.[33] Zu diesem Zweck sind alle Bundesbehörden verpflichtet, ihre Unterlagen nach Aktenschluss dem Bundesarchiv anzubieten, damit es sie auf bleibenden Wert prüfen kann.[34] Im Idealfall vereinbaren Archiv und Behörde einen regelmäßigen Turnus, zu dem ein Archivar die anbietungspflichtigen Unterlagen begutachtet. Sofern er sie für archivwürdig befindet, werden sie dem zuständigen Archiv übergeben. Ansonsten werden die Akten vernichtet. All dies gilt gleichermaßen für Papier- und elektronische Unterlagen.[35]

Die Bewertung ist also ein entscheidender Schritt im archivarischen Arbeitsprozess. Sie entscheidet darüber, aus welchem Behördenschriftgut Archivgut wird, das auf Dauer für die Öffentlichkeit zugänglich und verwertbar gemacht wird. Was ins Archiv übernommen wird, wird zur Grundlage für zukünftige Forschung; was nicht übernommen wird, wird vernichtet. Es handelt sich also um eine Entscheidung von Tragweite. Aus Vorsicht den größten Teil des angebotenen Schriftguts zu übernehmen, wäre allerdings keine Lösung: Die ins Archiv übernommenen Mengen müssen für die Archivare handhabbar bleiben, schließlich müssen die Archivalien fachgerecht gelagert und erschlossen werden.

2.2.2 Bewertungsmethoden und -kriterien

Ein klassisches Hilfsmittel zur Bewertung ist der Aktenplan der aktenführenden Behörde. Er gibt einen Überblick über das Aufgabenspektrum einer Behörde sowie über die zu den Aufgaben entstehenden Unterlagen. Anhand des Aktenplans können die Aufgaben und Themenfelder identifiziert werden, bei denen voraussichtlich Unterlagen von bleibendem Wert anfallen werden oder zu denen gesetzliche Aufbewahrungsfristen existieren. Ganze Aktenplanpunkte können so als archivwürdig gekennzeichnet oder zur Kassation freigegeben werden. Die Akten zu den übrigen Punkten müssen bewertet werden. Neben dem Aktenplan werden häufig Organigramme, Geschäftsverteilungspläne und Abgabelisten zur Unterstützung der Bewertung herangezogen – also Unterlagen, die über die Organisation der Behörde und der bei ihr anfallenden Akten Auskunft geben.

---

[33] § 3 (1) Bundesarchivgesetz.
[34] § 5 (1) und (2) Bundesarchivgesetz.
[35] § 1 Nr. 9 Bundesarchivgesetz.

Was genau den bleibenden Wert einer Akte ausmacht, war lange eine lange Zeit Frage des archivarischen „Fingerspitzengefühls“. Kriterien waren (und sind auch heute noch) zum Beispiel öffentlichkeitswirksame oder die gesellschaftliche Debatte prägende Themen und das Vorkommen von Personen des öffentlichen Lebens. Insbesondere Kommunalarchive zielen häufig darauf ab, mit den übernommenen Akten ein Bild der Lebenswirklichkeit in ihrem Sprengel zu vermitteln. Auch soll das Verwaltungshandeln reflektiert werden. Mitarbeiterinnen der Behörde werden in der Regel bei der Bewertung konsultiert, um sehr typische oder herausstechende Vorgänge aufzuzeigen. Akten werden nur von der federführenden Behörde übernommen, Auszüge und Duplikate der anderen beteiligten Behörden werden vernichtet.

Während die Kriterien sich nicht essentiell geändert haben, werden Bewertungsentscheidungen inzwischen häufig nicht mehr von einer einzelnen Archivarin nach ihrem Gutdünken getroffen, sondern mit Hilfe strukturierter Konzepte wie beispielsweise Dokumentationsprofilen[36] oder Archivierungsmodellen[37] von Teams erarbeitet und transparent gemacht.

Die Strukturen und Abläufe der klassischen Schriftgutverwaltung sind für die Bewertung also essentiell. Denn mit ihrer Hilfe können größere Mengen von Schriftgut anhand von Titel und (ungefährem) Inhalt bewertet werden, ohne einen Blick in jede einzelne Akte werfen zu müssen.

---

[36] Dokumentationsprofile haben zum Ziel, die lokale Lebenswelt zu kategorisieren und die sie abbildenden Unterlagen möglichst vollständig zu erfassen. Es werden auch nicht im Zuständigkeitsbereich des Archivs liegende Registraturbildner einbezogen, sofern die bei ihnen entstehenden Quellen zur Erfüllung anfangs festgelegter Dokumentationsziele beitragen. Der Ansatz geht von Themenfeldern aus und wird vor allem in Kommunalarchiven verfolgt. Mehr Informationen bietet die Arbeitshilfe „Erstellung eines Dokumentationsprofils für Kommunalarchive“ der Bundeskonferenz der Kommunalarchive.

[37] Archivierungsmodelle betrachten ganze Verwaltungszweige und gleichen sowohl vertikal (zwischen einander nachgeordneten Behörden) als auch horizontal (zwischen sich auf gleicher Ebene befindenden Behörden) die dort entstehenden Unterlagen ab, um die aussagekräftigste Überlieferung zu identifizieren. Sie legen Bewertungskriterien fest und dokumentieren diese, genau wie die getroffenen Bewertungsentscheidungen. Dadurch soll die Effizienz der Bewertung gesteigert, Redundanz minimiert und Transparenz geschaffen werden. Vgl. Landesarchiv Nordrhein-Westfalen: Steuerung der Überlieferungsbildung mit Archivierungsmodellen – Eine Konzeption für das Landesarchiv Nordrhein-Westfalen.

2.3 Probleme bei der Bewertung von Dateiablagen

Die Inhalte von Dateiablagen unterliegen genau wie Papierdokumente der Anbietungspflicht. Aufgrund der Masse der in ihnen enthaltenen Dateien ist bei den Dateiablagen eine gründliche Bewertung besonders wichtig.[38]

Die in Kapitel 2.1 beschriebene Heterogenität und ihre fehlende Einordnung in die herkömmlichen Strukturen der Schriftgutverwaltung erschweren die Nutzung gebräuchlicher Bewertungsmethoden. Im Gegensatz zu elektronischen Akten wird Dokumenten in einer Dateiablage, die nie zu einer Akte gegeben wurden, in der Regel kein Aktenzeichen und somit keine Position im Aktenplan zugeordnet. Es besteht ebenfalls kein Überblick darüber, welche Themen und Aufgaben in der Dateiablage behandelt werden. Selbst die Einordnung in eine anscheinend klare und durchdachte Ordnerstruktur bedeutet nicht, dass keine Parallelstrukturen voller Dubletten existieren können, oder dass alle Unterordner ebenfalls bis auf die unterste Ebene hinab durchstrukturiert sind. Da die Entstehungsprozesse nicht abgebildet werden, ist nicht immer klar, welche Kopie oder Version einer Datei die führende ist. Dateibenennungen können aussagekräftig sein, aber auch dies ist nicht zwingend der Fall und hängt stark vom Bearbeiter ab. Die Dateien analog zu Papierakten zu überfliegen oder durchzublättern, um sich ein Bild von ihrem Inhalt zu verschaffen, ist langwierig – insbesondere, wenn es sich um große Dateien mit spürbaren Ladezeiten handelt. Zudem die Menge an digitalen Dateien oft ungleich größer ist als die Anzahl analoger Akten in einer typischen Ablieferung, die durch Aktenschlussdaten, Aufbewahrungsfristen und Regalplatz begrenzt wird.

Die oben beschriebenen Bewertungsmethoden geraten unter diesen Umständen an ihre Grenzen. Wendt und Westphal schlagen daher vor, die Gesamtübernahme oder -kassation von Dateiablagen in Erwägung zu ziehen und den Nutzern das Auffinden relevanter Informationen in dem Dateienwust zu überlassen.[39] Miegel und Rödel ziehen ebenfalls die Gesamtübernahme oder -kassation in Betracht, sehen aber deutliche Nachteile dabei, wie zum Bespiel die Übernahme nicht archivfähiger Formate

[38] Im Folgenden wird nur auf inhaltliche Bewertungskriterien eingegangen. Technische Kriterien, wie beispielsweise das Dateiformat, -größe oder -verschlüsselung, werden nicht betrachtet, da sie für die vorliegende Problemstellung nicht relevant sind.
[39] Vgl. Wendt und Westphal, S. 108.

und redundanter Dateien[40] sowie den so entstehenden, für den Nutzer kaum zu überblickenden Datendschungel[41]. Für realistischer halten sie eine Mischung von Bewertung auf Ordnerebene und – wo nötig – Einzelbewertung von Dateien[42] nach vorheriger Reduktion der Gesamtmenge[43]. Sie betonen die Notwendigkeit effizienter Bewertungsmethoden, da der Arbeitsaufwand bei der Bewertung hunderttausender Dateien sonst ins Unermessliche steigen würde.[44] Dieses Ziel verfolgt vorliegende Arbeit: Methoden für die inhaltliche Bewertung von Dateiablagen zu finden, die sowohl Strukturen als auch interessante Einzeldokumente offen legen können, ohne dass jedes Dokument einzeln betrachtet werden muss. Dafür muss ein Überblick über die Inhalte der Ablage geschaffen, eine Struktur in ihr aufgedeckt werden.

3 Inhaltlicher Zugriff auf unsortierte Dokumentenmengen über computerlinguistische Methoden

Herkömmliche archivische Bewertungsmethoden scheitern an Dateiablagen, da die Unterlagen strukturierende Hilfsmittel fehlen und so keine Übersicht über den Gesamtbestand erlangt werden kann. Eine Möglichkeit, solche Hilfsmittel zu ersetzen und die Bewertung wieder zu ermöglichen, könnte die Erkundung der Dateiablagen mit Hilfe maschineller Verfahren sein. Vorliegende Arbeit unternimmt daher den Versuch, Methoden vorrangig aus dem Bereich der Computerlinguistik einzusetzen, um einen explorativen Zugang zu Dateisammlungen zu schaffen und ihre Bewertung zu unterstützen. Dies soll über eine Webanwendung realisiert werden (siehe Kapitel 4). Die zur Implementierung in der Anwendung vorgesehenen Verfahren werden in den folgenden Abschnitten kurz erläutert. Zunächst wird auf das Einlesen der Textdateien und ihre Verarbeitung zur Vorbereitung der weiteren Nutzung eingegangen. Als grundlegender Zugang zu den Texten ist eine Volltextsuche vorgesehen. Über das Verfahren MinHash sollen verschiedene Versionen einer Datei identifiziert werden. Mit der Analyse von Worthäufigkeiten und N-Grammen sollen dominante Themen im Korpus und über Named Entity Recognition Eigennamen identifiziert werden. Der Versuch, Strukturen im Bestand offenzulegen, erfolgt über

[40] Vgl. Miegel und Rödel, S. 33.
[41] Vgl. ebd., S. 36.
[42] Vgl. ebd., S. 34.
[43] Vgl. ebd., S. 31 f.
[44] Vgl. ebd., S. 34.

Topic Modelling und Clustering. Neben der Funktionsweise der Verfahren werden auch ihr jeweiliger Zweck vorgestellt und die Anwendbarkeit auf die zu nutzenden Daten evaluiert.

3.1 Dateiupload und -verarbeitung

Zum Lesen von Dokumenten wird die Python-Bibliothek textract[45] eingesetzt. Sie bildet einen Wrapper um eine Ansammlung von Bibliotheken, die verschiedenste Formate auslesen können. Laut Dokumentation werden .csv, .doc, .docx, .eml, .epub, .gif, .jpg, .jpeg, .json, .html, .htm, .mp3, .msg, .odt, .ogg, .pdf, .png, .pptx, .ps, .rtf, .tiff, .tif, .txt, .wav, .xlsx und .xls unterstützt. Nicht alle diese Formate sollen im begleitenden Projekt verarbeitet werden: Auf das Einlesen der Bild- und Audio-Formate, aus denen mit Optical Character Recognition (OCR) und automatischer Spracherkennung Text extrahiert werden kann, wird bewusst verzichtet, da keine Ressourcen für die Qualitätskontrolle vorhanden sind, die vor der Nutzung so erzeugter Textdaten notwendig wäre. Manche der aufgeführten Textformate (zum Beispiel .eml oder .odt) haben sich in Vorfeldtests zudem als nicht tatsächlich lesbar herausgestellt.

Zur Verarbeitung vorgesehen sind deshalb nur Dateien in den Formaten .csv, .doc, .docx, .epub, .htm, .html, .json, .pdf, .pptx, .tsv, .txt, .xls und .xlsx. Mit Hilfe der Python-Standard-Bibliothek können außerdem zip-Archive und tar-Archive entpackt und die darin enthaltenen Dateien gelesen werden. Dabei werden nur die Volltexte und die Erstellungsdaten der Dateien ausgelesen. Weitere Metadaten, wie zum Beispiel Autoren oder Dokumenttitel[46], sind zu inkonsistent (eingetragen und auslesbar), um verwertbar zu sein.[47]

In einigen Fällen können auch Formate aus obiger Positivliste nicht gelesen werden. Verschlüsselte PDFs müssen übersprungen werden, und scheinbare Word-Dokumente liegen nicht immer tatsächlich im .doc-Format (oder in einer sehr alten, von der Bibliothek nicht lesbaren Variante) vor. Auch nicht lesbare temporäre Dateien

[45] Vgl. Malmgren, Dean: textract.
[46] Nicht zu verwechseln mit dem Dateinamen, der Teil des Pfads und somit bekannt ist.
[47] Eine Schwierigkeit ist, dass textract und andere Bibliotheken das Auslesen dieser Metadaten nicht oder nur teilweise anbieten. Dies würde bei Inklusion der Metadaten dazu führen, dass beispielsweise aus PDF- und docx-Dateien erzeugte Dokumente mit Metadaten versehen wären, doc-basierte Dokumente dagegen nicht. Deswegen fiel die Entscheidung schon vor der Implementierung von textract gegen die Auswertung von Metadaten.

finden sich in den Dateisammlungen.[48] Diese Faktoren reduzieren die Menge an tatsächlich zu verarbeitenden Dokumenten im Gesamtkorpus.

Welcher Anteil eines Korpus‘ eingelesen wird, ist schwer nachzuvollziehen. Denn in Datei-Archiven können beliebig viele Dateien enthalten sein, die ohne weitere Hilfsmittel nicht im Vorfeld bekannt sind und die vom Explorer und ähnlichen Tools nicht gezählt werden können. Es wird also ein unvollständiger Ausschnitt der im Korpus enthaltenen Textdateien verarbeitet.[49]

3.2 Vorverarbeitung

Um die Volltexte verarbeiten zu können, müssen sie zunächst in eine für das jeweilige Programm verwertbare Form gebracht werden. Es werden insgesamt drei verschiedene Repräsentationen der Texte benötigt, um die weiteren Verarbeitungsschritte durchzuführen.

Für die Erkennung von Eigennamen, die unter anderem auf der Einordnung von Wörtern in Satzstrukturen beruht (siehe Kapitel 3.7), muss jedes Dokument in Sätze aufgeteilt vorliegen. Diese Aufgabe ist weniger trivial, als sie auf den ersten Blick scheinen mag.[50] So sind zwar bestimmte Satzzeichen wie Punkte, Frage- oder Ausrufezeichen Hinweise auf ein mögliches Satzende. Aber Frage- und Ausrufezeichen können auch in Einschüben stehen, Punkte hinter Daten, Ordinalzahlen oder Abkürzungen. Diese regelbasiert zu erkennen, würde ebenfalls nicht für das zuverlässige Erkennen von Satzenden ausreichen – denn ein Satz kann auch mit einer Abkürzung enden.

Es müssen also zwei Probleme gelöst werden: Erstens müssen Punkte innerhalb einer Abkürzung (oder eines ähnlichen Konstrukts, wie Ordinalzahlen) gefunden werden, und zweitens müssen Abkürzungen im Satz und Abkürzungen am Satzende disambiguiert werden. Kiss und Strunk schlagen dafür vor, Abkürzungen als strenge

---

[48] Von ähnlichen Erfahrungen berichten auch mit der Verarbeitung von Dateiablagen betraute Archivarinnen. Vgl. Miegel und Rödel, S. 32.

[49] Denkbar wäre es, zur Verbesserung der Transparenz beim Auslesen ein Protokoll der eingelesenen sowie der gescheiterten oder ignorierten Dateien anzulegen. Dies könnte als Orientierung dienen; bei Bestandsgrößen von hunderttausenden bis Millionen Dateien aber würde vermutlich niemand dieses Protokoll komplett durchsehen, außerdem ist das Schreiben in eine Datei ist eine langsame Operation, die die Verarbeitungszeit weiter in die Höhe treiben könnte.

[50] Vgl. Kiss, Tibor und Jan Strunk: Unsupervised Multilingual Sentence Boundary Detection. In *Computational Linguistics*, S. 485 f.

Kollokationen von einem gekürztem Wort und einem Punkt zu betrachten, die tendenziell aus einer kurzen Zeichenfolge bestehen und Punkte innerhalb des Worts enthalten.[51] Ein Likelihood-Ratio-Klassifikator[52], dessen Parameter an die Kollokations-Hypothese angepasst wurden, wird zum Erkennen von Satzende-Zeichen eingesetzt. Implementiert wurde dieser sogenannte Punkt-Tokenizer durch die Python-Bibliothek Natural Language Tool Kit (NLTK)[53], welche auch für die weiteren Vorverarbeitungsschritte eingesetzt wird. Sie stellt trainierte Modelle für die Satzerkennung in verschiedenen Sprachen zur Verfügung, unter anderem auch Deutsch.

Die meisten der nachfolgend eingesetzten Analyseschritte benötigen die Einteilung der Volltexte nicht in Sätze, sondern in Worte. Aus diesem Grund ist eine weitere notwendige Repräsentation die des Korpus‘ in Form von tokenisierten Dokumenten. Tokenisierung bezeichnet die Einteilung eines Texts in Tokens, die grob mit Worten korrelieren.[54] Auch hier können die Texte nicht bloß bei Leerzeichen geteilt werden, es muss beispielsweise auch mit Satzzeichen oder im Englischen mit Kontraktionen umgegangen werden. Das NLTK benutzt einen Tokenizer, der mit regulären Ausdrücken arbeitet und so ein zuvor in Sätze eingeteiltes Dokument nach festgelegten Regeln in Tokens splittet. Um die Bearbeitungsdauer der später erfolgenden Analysen abzukürzen, werden zwei derartige Repräsentationen erzeugt: Ein tokenisiertes Korpus mit und eines ohne Stoppwörter, also nicht bedeutungstragenden Tokens wie Artikeln, Konjunktionen oder Pronomen.

---

[51] Vgl. Kiss und Strunk, S. 486 f.

[52] Ein Likelihood-Ratio-Klassifikator nimmt an, dass dem vorliegenden Text ein Sprachmodell zugrunde liegt, dessen Parameter noch unbekannt sind. Es werden die Parameter-Werte ermittelt, die mit der höchsten Wahrscheinlichkeit den vorliegenden Text erzeugt haben. Dafür wird eine Nullhypothese mit einer alternativen Hypothese verglichen. Bei Kiss und Strunk ist die Nullhypothese, dass kein Zusammenhang zwischen einem Punkt und dem davorstehenden Wort existiert, die alternative Hypothese nimmt einen Zusammenhang an. Auf Grundlage der Textdaten wird die Wahrscheinlichkeit beider Hypothesen berechnet und ein Quotient aus ihnen gebildet (vgl. ebd., S. 489 f.). Dieser Quotient ist (neben anderen Faktoren) Grundlage für die später vorgenommene Klassifikation (vgl. ebd., S. 492), also für die automatische Kategorisierung von Daten auf Grundlage eines annotierten Korpus. Aus dem Korpus werden die Merkmale der Kategorien algorithmisch gelernt und auf neue Datensätze angewandt. Anschließend kann überprüft werden, wie viele Datensätze der Annotation entsprechend – also korrekt – eingeordnet wurden.

[53] Vgl. Bird, Steven, Ewan Klein und Edward Loper: Natural language processing with python. Sebastopol 2016.

[54] Vgl. Manning, Christopher, Prabhakar Raghavan und Hinrich Schütze: Introduction to Information Retrieval. Cambridge 2009, S. 22–26.

Alle drei so erzeugten Korpora behalten die Groß- und Kleinschreibung der Originaltexte bei, da diese sowohl für die Erkennung von Namen relevant ist als auch für die Unterscheidung ansonsten homonymer Wörter.

Die Vorverarbeitung ist als einer von wenigen Schritten in vorliegender Arbeit sprachspezifisch. Die Regeln für Interpunktion unterscheiden sich von Sprache zu Sprache. Stoppwörter werden in Form von Listen übergeben und sind ebenfalls sprachspezifisch. Da mit Textsammlungen aus deutschen Quellen gearbeitet werden soll, werden für diese Komponenten auf Deutsch spezialisierte Ressourcen verwendet. Sollten in den zu verarbeitenden Korpora nichtdeutsche Texte auftauchen, ist mit schlechten Ergebnissen zu rechnen, da die Vorverarbeitung nicht ordnungsgemäß durchgeführt werden kann.

3.3 Volltextsuche

Eine Volltextsuche ist ein grundlegender Zugang zu einer Sammlung digitaler Texte. Sie ermöglicht es, alle Dokumente in der Sammlung zu finden, in der bestimmte Suchworte vorkommen.

Im diese Arbeit begleitenden Softwareprojekt wurde die Python-Bibliothek Whoosh[55] für die Implementierung einer Suche gewählt. Whoosh erzeugt aus Dokumente repräsentierenden Python-Objekten einen Index aller darin vorkommenden Worte und den korrespondierenden Dokumenten.[56]

Neben dem offensichtlichen Nutzen fungiert die Suchfunktion auch als verknüpfendes Element zwischen allen Dokumenten, die ein bestimmtes Merkmal teilen. Wird beispielsweise über die Named Entity Recognition (siehe Kapitel 3.7) ein Personenname erkannt, können über die Suche alle Dokumente gefunden werden, in denen er vorkommt.

[55] Vgl. Chaput, Matt: Whoosh.

[56] Die Library beinhaltet neben einer Suchfunktion mit Operatoren auch weiterführende Funktionen wie Lemmatisierung, den Umgang mit Variationen des Suchworts, das Ignorieren von Akzenten auf Buchstaben oder die Suche nach N-Grammen. Auf die Implementation dieser Funktionen wird verzichtet, weil das Ziel der Arbeit nicht über eine Suchfunktion erreicht werden kann und der Fokus auf den die Dokumente strukturierenden Verfahren liegt. Nur der Kern von Whoosh, die Suche, wird verwendet.

3.4 Versionserkennung mit MinHash

Ein Problem im Umgang mit Dateiablagen ist die Erkennung von Duplikaten oder Versionen der gleichen Datei. Einmal erkannt können Duplikate ausgeblendet und Versionen des gleichen Dokuments miteinander verknüpft werden, um die finale Fassung eines Dokuments zu identifizieren und so Redundanzen bei der Bewertung zu minimieren.[57]

Um Versionen und Duplikate effizient zu erkennen, wird die Python-Library datasketch[58] herangezogen, welche für die Verarbeitung großer Datenmengen mit probabilistischen Verfahren gemacht wurde. Datasketch kann unter anderem zur Berechnung des Jaccard-Indexes zweier Dokumente genutzt werden. Dieser beschreibt die Ähnlichkeit zweier Mengen A und B, indem ihre Schnittmenge durch die Gesamtmenge von A und B geteilt wird. Ein Wert von 1 beschreibt Duplikate, ein Wert von 0 zwei Mengen ohne Schnittmenge. Um die Ähnlichkeit von Dokumenten zu berechnen, können die in ihnen vorkommenden Worte oder N-Gramme als die zu vergleichenden Mengen verwendet werden.[59]

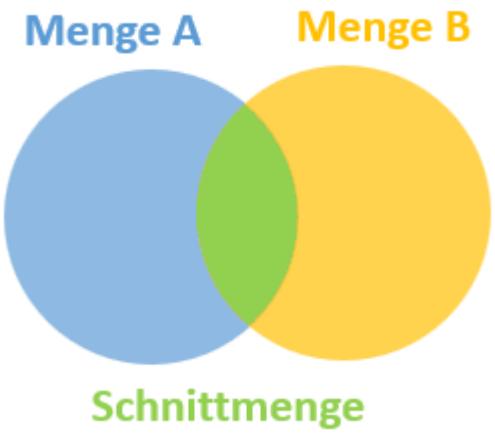


Abbildung 1: Jaccard-Index

Datasketch behandelt den Volltext jedes Dokuments als eine Menge zusammengesetzt aus Tokens, die in Python als Set von Strings repräsentiert wird. Da in einem Set nur einzigartige Objekte enthalten sein dürfen, geht die Wortreihenfolge von mehrfach vorkommenden Worten verloren; es wird ein Bag-of-Words-Verfahren[60] genutzt. Aus den die Dokumente

[57] Denkbar wäre auch gewesen, die Duplikate von der weiteren Verarbeitung auszuschließen. Darauf wurde sowohl aus praktischen als auch aus inhaltlichen Gründen verzichtet. Denn zum Ausschluss der Versionen wäre zunächst eine Durchführung der Versionserkennung notwendig, welche wiederum eine tokenisierte Version des Korpus‘ benötigt. Nach Erkennung der Duplikate müssten alle Vorverarbeitungsschritte (siehe Kapitel 3.2) erneut durchgeführt werden, um ein Korpus ohne Duplikate zu erzeugen; erst dann könnten die Analysen ausgeführt werden. Außerdem kann das mehrfache Vorkommen einer Datei Absicht sein und auf ihre Bedeutung hinweisen. Alle Kopien standardmäßig auszuschließen, würde dies verzerren.

[58] Vgl. Eric Zhu und Vadim Markovtsev: Ekzhu/Datasketch: First Stable Release.

[59] Vgl. Liu, Bing: Web Data Mining. Berlin, Heidelberg 2011, S. 231 f.

[60] Vgl. Harris, Zellig S.: Distributional Structure. In *WORD*.

repräsentierenden Sets werden mit der Methode MinHash LSH[61] Hashwerte errechnet, die als Mengen für die Ermittlung des Jaccard-Indexes eingesetzt werden. Die Ähnlichkeit wird über den Jaccard-Index der Hashwerte berechnet, die das Vokabular der Dokumente repräsentieren.

Die oben beschriebene Berechnung funktioniert vor allem dann gut, wenn es sich um Versionen desselben Dokuments mit ähnlicher Länge handelt. Liegen dagegen mehrere Versionen eines Dokuments vor, die dessen Entstehungsgeschichte abdecken, und während Version 1 nur zehn Seiten umfasst, enthält Version 2 einhundert Seiten, dann handelt es sich zwar eindeutig um Versionen desselben Dokuments. Aber ihr Jaccard-Index wird niedrig sein, weil die Schnittmenge der Dokumente im Vergleich zur ihrer Gesamtmenge sehr klein ist – neunzig Seiten aus Version 2 fehlen in Version 1.[62]

Zur Erkennung von Duplikaten ist der Jaccard-Index ein sehr zuverlässiges Mittel. Bei der Identifikation von Versionen hängt die Trefferquote davon ab, ab welchem Ähnlichkeitswert ein Dokument als Version eines anderen gezählt wird. Abgesehen vom Problem mit unterschiedlich langen Versionen desselben Dokuments ist aber davon auszugehen, dass es sich beim Jaccard-Index auf Basis von MinHash um ein geeignetes Mittel für die Versionserkennung handelt.

3.5 Worthäufigkeit

Das Zählen von Worthäufgkeiten ist ein sehr einfaches Mittel, um einen Eindruck von den in einem Korpus behandelten Themen zu erlangen. Um sinnvolle Ergebnisse zu erhalten, müssen zunächst die Stoppworte (siehe Kapitel 3.2) gefiltert werden.

Anstelle der häufigsten können auch die spezifischsten Worte gesucht werden, indem sie über das Verfahren Tf-idf (Termfrequenz – inverse Dokumentfrequenz) gewichtet werden.[63] Tf-idf legt die Annahme zugrunde, dass Worte, die in allen

---

[61] Vgl. Leskovec, Jure, Anand Rajaraman und Jeffrey David Ullman: Mining of Massive Datasets. Cambridge 2014, S. 87–91.

[62] Datasketch bietet für dieses Problem eine Alternative zu MinHash LSH an. Bei MinHash LSH Ensemble wird die Schnittmenge durch die Größe der kleineren Menge geteilt, um das oben geschilderte Problem zu vermeiden. In der zur Verfügung stehenden Zeit ist allerdings keine sinnvolle Ergebnisse liefernde Implementierung dieses Algorithmus‘ gelungen. Da sich das Problem auch nach Sichtung der Testbestände (siehe Kapitel 5.2) nicht als allzu dringlich herausgestellt hat, wurde auf weitere Umsetzungsversuche verzichtet.

[63] Vgl. Liu 2011, S. 217.

Dokumenten eines Korpus‘ häufig vorkommen, nicht spezifisch für den Inhalt eines bestimmten Dokuments sind. Kommt ein Wort dagegen im Gesamtkorpus selten vor, aber in einem Dokument sehr häufig, ist es vermutlich für dieses Dokument sehr aussagekräftig. Für die Bestimmung der normalisierten Termfrequenz *tf* wird die Häufigkeit f_t des zu analysierendes Terms *t* in Dokument *d* durch die Anzahl der Vorkommen des häufigsten Terms in *d* geteilt: [64]

$$tf = \frac{f_t}{f_{max}}$$

Die inverse Dokumentfrequenz von *t* wird über den Logarithmus der Gesamtzahl aller Dokumente im Korpus (*N*) geteilt durch die Anzahl der Dokumente, in denen *t* vorkommt (df_t), ermittelt. Um sicherzustellen, dass *idf* nie null ist, wird Add-one Smoothing vorgenommen:

$$idf = \log\frac{N}{df_t} + 1$$

Die Gewichtung *w* wird errechnet über die Multiplikation von *tf* und *idf*:

$$w = tf \times idf$$

Tf-idf wird eingesetzt, um das für ein Exemplar aus einer Menge spezifische Vokabular zu identifizieren. Also kann es die aussagekräftigsten Worte eines Dokuments innerhalb einer Sammlung aufzeigen. Über die gesamte Dokumentsammlung könnte mit Tf-idf nur eine Aussage getroffen werden, wenn eine Vielzahl solcher Sammlungen vorläge und das Vergleichskorpus bilden würde.

Die Worthäufigkeit und -spezifität werden vermutlich bei der Untersuchung einzelner Dokumente hilfreicher sein als für ganze Sammlungen, sofern über letztere bereits grundlegende Informationen vorhanden sind. Es ist nicht zu erwarten, dass viele der durch diese Methoden ermittelten Worte den Nutzer überraschen werden. Sollten aber sehr wenige Informationen zu den Inhalten der Dokumente vorhanden sein, oder wenn der Inhalt eines einzelnen Dokuments schnell ermittelt werden muss, können die Worthäufigkeiten sich als nützlich erweisen.

[64] Die Normalisierung dient dem Vorbeugen von Verzerrungen durch sehr lange Dokumente, in denen Begriffe dementsprechend häufiger auftreten können.

3.6 N-Gramme

Ein N-Gramm ist eine Abfolge von *n* aufeinander folgenden Tokens. Ein einzelnes Token ist also ein Unigramm, zwei Tokens bilden ein Bigramm, drei ein Trigramm. N-Gramme zeigen typische oder häufige Wortfolgen auf.

Im vorhergegangenen Kapitel wurde die Häufigkeit von Unigrammen diskutiert. Womöglich sind aber auch die häufigsten Bi- und Trigramme für die Ermittlung des Inhalts eines Dokuments oder einer Dokumentensammlung relevant, denn damit finden sich mehrere Worte umspannende Begriffe.[65] Die Identifikation von Bi- und Trigrammen wird mit dem entsprechenden Modul des NLTK vorgenommen, welches Bi- und Trigramme in einem tokenisierten Korpus findet und nach verschiedenen Relevanzkriterien sortiert ausgibt. Das einfachste dieser Kriterien ist die absolute Vorkommenshäufigkeit. Außerdem genutzt werden die Verfahren Pointwise Mutal Information[66], Chi Square[67] und Likelihood Ratio[68]. Diese (hier nicht im Detail beschriebenen) Verfahren haben alle zum Ziel, besonders signifikante N-Gramme zu identifizieren. Signifikant ist ein N-Gramm dann, wenn ein starker Zusammenhang zwischen den Tokens im N-Gramm besteht. Dies ist zum Beispiel der Fall, wenn zwei Tokens mit sehr hoher Wahrscheinlichkeit gemeinsam auftreten, und vergleichsweise selten mit anderen Tokens zusammenstehen. Die genaue Berechnung der Signifikanz unterscheidet sich je nach Verfahren.

Das Potential der N-Gramme ist ähnlich wie das der Worthäufigkeiten: Insbesondere auf Dokumentebene und für Sammlungen, zu denen wenig bekannt ist, können sie Hinweise auf die dominanten Themen geben. Ansonsten werden sie vermutlich viele branchenspezifische Phrasen identifizieren, die anders als die allgemeingültigeren Stoppworte nicht von vorneherein ausgeschlossen werden können.

---

[65] Auf N-Gramme mit $n > 3$ wird in dieser Arbeit verzichtet, da nicht nur die benötigte Rechenleistung mit höheren Werten deutlich steigt, sondern zugleich die Aussagekraft der N-Gramme sinkt. Je länger ein N-Gramm ist, desto seltener werden Wiederholungen davon gefunden werden, was zu immer weniger verwertbaren Ergebnissen führt.

[66] Vgl. Manning, Christopher und Hinrich Schütze: Foundations of statistical natural language processing. Cambridge 2005, S. 178–183.

[67] Vgl. ebd., S. 169–172.

[68] Vgl. Dunning, Ted: Accurate Methods for the Statistics of Surprise and Coincidence. In *Computational Linguistics*.

3.7 Named Entitiy Recognition

Named Entitiys bezeichnen beispielsweise natürliche und juristische Personen sowie Orte. Ihre Erkennung ist für Archivare nicht nur deshalb interessant, weil sie Aufschluss über den Inhalt eines Dokuments geben können. Es ist auch davon auszugehen, dass Personennamen (insbesondere von Personen, die nicht in der Öffentlichkeit stehen) auf schutzwürdige Informationen hinweisen, die von den Archivarinnen im Falle der Übernahme berücksichtigt werden sollten.

Die Named Entitity Recognition (NER) wird mit Hilfe des NLTK vorgenommen. Es nutzt dafür den Recognizer der Stanford Natural Language Processing Group[69], für den unter anderem ein mit deutschen Texten trainiertes Modell zur Verfügung steht. NER ist ein Klassifikationsproblem, es bewegt sich also im Feld des überwachten maschinellen Lernens. Ein annotiertes Korpus wird als Trainingsmaterial genutzt, um Muster in den Daten zu erkennen, anhand derer sie kategorisiert werden können. Diese werden anschließend auf unbekannte Daten angewandt. Für das Training des Stanford NER wurden CoNLL-2003-Daten[70] und der Huge German Corpus[71] genutzt. Die Klassifikation erfolgt mit Hilfe von Conditional Random Field (CRF), einem probabilistischen Modell zur Segmentierung und Auszeichnung sequentieller Daten.[72] Den erkannten Named Entitys wird eine von vier Kategorien zugewiesen: Person, Ort, Organisation oder Weiteres.

Der Tagger erkennt auch beieinanderstehende Namen als unterschiedliche Entitäten. Stehen also beispielsweise Vor- und Nachname einer Person hintereinander, werden sie als zwei Entitäten erfasst. Dies soll im begleitenden

---

[69] Vgl. Finkel, Jenny Rose, Trond Grenager und Christopher Manning: Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics*.

[70] Vgl. Tjong Kim Sang, Erik und Fien de Meulder: Introduction to the CoNLL-2003 Shared Task. In *Proceedings of CoNLL-2003*.

[71] Vgl. Schiller, Anne, Simone Teufel, Christine Stöckert et al.: Guidelines für das Tagging deutscher Textcorpora mit STTS.

[72] CRF unterscheidet sich von anderen üblichen Verfahren wie beispielsweise Markov-Modellen dadurch, dass die Übergangswahrscheinlichkeit von einem Zustand zum nächsten nicht für ein einzelnes Element, sondern eine Sequenz von voneinander abhängigen Elementen maximiert wird. Dadurch eignet es sich besonders gut für die Auszeichnung von linearen, voneinander abhängigen Daten wie Texten, und wird folgerichtig für Aufgaben wie wie Part-of-Speech-Tagging und NER eingesetzt (vgl. Lafferty, John, Andrew McCallum und Fernando C.N. Pereira: Conditional Random Fields. In *Proceedings of the 18th International Conference on Machine Learning*, S. 282 f., vgl. McCallum, Andrew und Wei Li: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, S. 188).

Softwareprojekt abgefangen werden, indem zusammenstehende Entitäten des gleichen Typs zu einer einzelnen Entität zusammengefasst werden. Dies bedeutet zugleich, dass mehrere nicht durch ein Satzzeichen voneinander getrennt stehende Entitäten, die tatsächlich unterschiedliche Personen bezeichnen, als eine einzige Entität erfasst werden.[73] Dies kommt aber aller Voraussicht nach seltener vor als der umgekehrte Fall und wird daher in Kauf genommen. Auch flektierte Formen des gleichen Worts werden als unterschiedliche Entitäten gezählt. Eine Disambiguierung von gleichnamigen Named Entitys findet nicht statt. Die Zählung der Entitäten ist demnach nicht präzise, sollte aber einen Eindruck der vorkommenden Namen, Personen und Organisationen vermitteln.

3.8 Topic Modelling

Ziel des Topic Modellings ist es, ein Korpus von Dokumenten auf in ihnen vorkommende sprachliche Muster zu untersuchen (sogenannte Topics). Es erfreut sich vor allem unter Literaturwissenschaftlern und Linguisten steigender Beliebtheit, wird aber auch in anderen geisteswissenschaftlichen Feldern vermehrt eingesetzt.[74] Eine der gängigsten Methoden für Topic Modelling ist Latent Dirichlet Allocation (LDA)[75], die für das begleitende Projekt über die Python-Library scikit-learn[76] implementiert wurde.

LDA ist eine probabilistische Methode, die latente Muster in den Daten aufdecken soll.[77] Es handelt sich um ein generatives Modell, das zunächst eine Annahme über die Entstehung von Dokumenten trifft: Es existiere eine Menge von Topics, denen verschiedene Wörter mit variierender Häufigkeit zugeordnet seien. Ein Dokument habe eine festgelegte Größe und bestehe zu je einem bestimmten Prozentsatz aus diesen Topics. Dem Bag-of-Words-Modell folgend werden ohne Rücksicht auf Syntax zu den Topics gehörige Wörter entsprechend der Wahrscheinlich des Topics im Dokument

[73] Beim Satz „Morgen hilft Alice Bob beim Umzug“ würde also eine Entität namens „Alice Bob“ erfasst werden. Dafür würden allerdings auch „Max Mustermann“ oder „Friedrich Wilhelm IV“ als je eine Entität aufgenommen.
[74] Vgl. Fechner, Martin und Andreas Weiß: Einsatz von Topic Modeling in den Geschichtswissenschaften. In *Zeitschrift für digitale Geisteswissenschaften*, S. 2.
[75] Vgl. Blei, David M., Andrew Y. Ng und Michael I. Jordan: Latent Dirichlet Allocation. In *Journal of Machine Learning Research*. Die folgenden Erläuterungen zu LDA nach Blei 2012.
[76] Vgl. Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort et al.: Scikit-learn. In *Journal of Machine Learning Research*.
[77] Vgl. Blei, S. 78 f.

und des Worts im Topic dem Dokument hinzugefügt, bis die anfangs gesetzte Länge des Dokuments erreicht ist.

Dieser Prozess könne umgekehrt werden, um Wörter in einem Dokument zu Topics zuzuweisen. Es wird also davon ausgegangen, dass ein Dokument auf die oben beschriebene Art und Weise zustande gekommen ist, und ausgehend von den Dokumenten als Ergebnissen des generativen Prozesses sollen zwei versteckte Variablen ermittelt werden: ihre Topic-Verteilung und die Topic-Zusammensetzung der gesamten Dokumentsammlung.[78] Dies geschieht in einem iterativen Prozess. Zu dessen Beginn stehen eine vorgegebene Zahl von Topics und eine Menge von Dokumenten. Jedem Wort in den Dokumenten wird zufällig ein Topic zugewiesen. Danach wird für jedes Wort *w* in jedem Dokument *d* nacheinander errechnet, in welches Topic *t* es vermutlich gehört:

$$P(t|w,d) = \frac{Häufigkeit\ von\ w\ in\ t}{Anzahl\ der\ tokens\ in\ t} \times (Anzahl\ der\ zu\ t\ gehörigen\ Worte\ in\ d)$$

Die Wahrscheinlichkeit für Topic *t* in Abhängigkeit von Wort *w* und Dokument *d* berechnet sich also aus der Wahrscheinlichkeit, dass *w* zu *t* gehört und der Anzahl der schon zu *t* gehörigen Worte in *d*.[79] Dabei gilt die Annahme, dass alle Zuweisungen außer der des aktuell betrachteten Worts korrekt sind. Diese Berechnung wird für alle Worte in allen Dokumenten so oft wiederholt, bis die Zuweisungen stabil sind.

Topic Modelling funktioniert ohne jegliche manuelle Annotation oder Vorbereitung. Um die Ergebnisse zu verbessern, können im Vorfeld Stoppwörter gefiltert werden, außerdem muss vom Bearbeiter eine passende Zahl an Topics festgelegt werden. Die Topics werden über die für sie signifikantesten Worte beschrieben, ihre Interpretation muss der Bearbeiter selbst vornehmen. Es ist also zu erwarten, dass die Benutzung von Topic Models Erfahrungswerte bezüglich ihrer Konfiguration und bei der Interpretation ihrer Ergebnisse benötigen wird.

3.9 Häufigkeitsbasierte Cluster mit Tf-idf

Eine weitere Methode, um die in einem Korpus enthaltenen Dokumente thematisch zu gruppieren, ist Clustering. Dabei handelt es sich um ein Verfahren des unüberwachten

[78] Vgl. Blei, S. 79–81.
[79] Vgl. Underwood, Ted: Topic modeling made just simple enough. Die Formel soll vorrangig Logik des Prozesses abzubilden und verzichtet daher auf die Hyperparameter.

maschinellen Lernens, welches Daten nach einem gewählten Ähnlichkeitsmaß gruppiert.

3.9.1 Maschinelles Lernen mit Dokumentvektoren

Im Gegensatz zum in Kapitel 3.7 angerissenen überwachten maschinellen Lernen wird beim unüberwachten Lernen ohne manuell annotierte Trainingsdaten gearbeitet. Die Bearbeiterin wählt einen Algorithmus und legt dessen Parameter fest, aber die Verteilung der Datensätze auf Gruppen erfolgt ohne ihr Zutun. Bei diesem Vorgehen werden zum Erzielen guter Ergebnisse meist sehr viele Trainingsdaten benötigt, dementsprechend hoch ist die Komplexität der Verfahren.[80] Dafür fällt die arbeitsintensive Annotation der Korpora weg, und dies kann für die Durchführbarkeit einer Methode den entscheidenden Unterschied machen. Dies gilt im vorliegenden Anwendungsfall: Es wäre im archivischen Arbeitsalltag kaum möglich, den gesamten Bestand einer Dateiablage zu sichten, Kategorien zu finden und ausreichend Dokumente damit zu annotieren, um eine Klassifikation durchzuführen.

Da die manuelle Annotation aller Dokumente keine Option ist, müssen Kriterien gefunden werden, anhand derer sie automatisiert gruppiert werden können. Der bloße Dokumenttext taugt dafür nicht, denn er besteht aus einer willkürlichen Abfolge von Zeichen, die keine semantischen Eigenschaften mitbringen. Auf Basis der Gesamtmenge der Texte aber können Repräsentationen der Tokens erzeugt werden, die eine semantische Dimension haben. Diese ergibt sich dadurch, dass die Tokens nicht isoliert, sondern im Kontext aller sie umgebenden Tokens betrachtet und miteinander verglichen werden. Ausgewählte Eigenschaften der Tokens werden in Werte überführt, und aus den Werten aller Tokens im Text ein Vektor erzeugt. Dieser kann über Distanzmaße wie die Euklidische Distanz mit den Vektoren anderer Dokumente verglichen und so der Grad der Ähnlichkeit zwischen den Vektoren (und demzufolge auch den Dokumenten) bestimmt werden.

Eine der einfachsten Methoden für die Zuweisung von Werten zu Tokens ist das Bag-of-Words-Verfahren, das eine Kookkurenz-Matrix der im Dokument enthaltenen

[80] Als Komplexität wird in der Informatik die Anzahl der notwendigen Ausführungsschritte zur Lösung eines Problems bezeichnet. Je höher die Komplexität eines Problems ist, desto mehr Ressourcen (Zeit oder Speicher) benötigt seine Lösung. Häufig gehen Reduktionen in einer Dimension zulasten der anderen.

Tokens erstellt.[81] Komplexere Modelle gewichten die Tokens zusätzlich, zum Beispiel nach Tf-idf (siehe Kapitel 3.5). So ergibt sich für jedes Dokument ein eindeutiger Vektor, der die enthaltenen Tokens abstrakt repräsentiert und mit dem weitere Verarbeitungen vorgenommen werden können. In vorliegendem Fall wird eine Matrix aus den Identifikatoren aller Dokumente in der einen Dimension und den Td-idf-Repräsentationen der in ihnen vorkommenden Tokens in der anderen gebildet.

3.9.2 Clustering der hochdimensionalen Dokumentvektoren mit k-means

Um die Vektoren zu gruppieren, ohne mindestens eine Auswahl aus mehreren tausend Beispielen manuell zu ordnen, wird ein Clustering-Algorithmus benötigt. Clustering behandelt das Problem der automatischen Gruppierung großer Datenmengen anhand ihrer inhärenten Eigenschaften. Es soll zugleich eine möglichst große Ähnlichkeit der Objekte in einer Gruppe zueinander bestehen, und eine möglichst große Unähnlichkeit zu den Objekten aller anderen Gruppen.[82] Die Definition von Ähnlichkeit (und somit die Entscheidung darüber, wie ein Cluster zusammengesetzt wird) hängt vom Algorithmus ab; die Expertise einer menschlichen Fachperson ist immer notwendig, um einzuschätzen, ob die automatisch getroffene Einteilung sinnvoll ist. Die Herausforderungen beim Clustering liegen in der Wahl des Algorithmus, seiner Parameter und in der Überprüfung der Cluster-Ergebnisse.

Die Ähnlichkeit zweier Vektoren wird an der Distanz zwischen ihnen im Vektorraum festgemacht. Dafür wird meist die Euklidische Distanz als Distanzmaß verwendet. Dies führt zu Problemen, sobald die Daten sehr viele Dimensionen haben. Denn dann wächst zwar der Merkmalsraum sehr schnell, aber nur ein kleiner Teil des Raums ist tatsächlich mit Daten besetzt – der Großteil ist leer. Das führt dazu, dass der Unterschied in der Distanz zwischen den beiden einander nächsten und den am weitesten voneinander entfernten Dokumenten sehr viel kleiner ist als die im Vektorraum mögliche Distanz. Dadurch werden die Vektoren nicht als sehr unterschiedlich voneinander wahrgenommen.[83]

---

[81] Vgl. Harris.
[82] Vgl. Jain, Anil K.: Data Clustering, S. 3.
[83] Vgl. Domingos, Pedro: A few useful things to know about machine learning. In *Communications of the ACM*, S. 81.

Dokument-Vektoren beinhalten einen Wert pro Token im Dokument und sind dementsprechend hochdimensional. Die besten Ergebnisse für ihre automatische Gruppierung wurden mit k-means erzielt. Dabei handelt es sich um einen vielseitig einsetzbaren Clustering-Algorithmus, der für eine vorgegebene Zahl von Clustern versucht, Gruppen mit möglichst geringer Varianz zu bilden.[84] Die Zentren dieser Cluster werden zu Beginn des Clustering-Prozesses entweder zufällig initiiert oder möglichst weit voneinander entfernt verteilt.[85] Jeder Datenpunkt wird dann dem Cluster zugewiesen, dessen Zentrum ihm am nächsten ist. Anschließend wird das Zentrum neu berechnet und der vorherige Schritt wiederholt, bis die Cluster-Zentren sich kaum mehr ändern.

Für k-means essentiell ist die Bestimmung des Parameters *k*, der die Anzahl der Cluster festlegt. Sie erfolgt meist über eine Trial-and-Error-Strategie: Verschiedene Werte werden ausprobiert und von einer Expertin bewertet. Der am ehesten dem erwarteten oder gewünschten Ergebnis entsprechende *k*-Wert wird ausgewählt.[86] Eine weitere Schwierigkeit bei der Nutzung von k-means ist, dass der Algorithmus nicht deterministisch ist. Er konvergiert bei lokalen Minima, was dazu führt, dass er bei verschiedenen Durchläufen verschiedene Ergebnissen erzeugt.

Es ist zu erwarten, dass aussagekräftige Cluster mit dieser Methode auch bei einer kleineren Sammlung mittellanger Dokumente entstehen. Als probabilistisches Modell genügen Tf-idf im Gegensatz zu Verfahren, die neuronale Netze nutzen (siehe zum Beispiel Kapitel 3.10) vergleichsweise kleine Korpora, um sinnvolle Ergebnisse zu erzeugen.[87] Die Ermittlung einer sinnvollen Clustermenge wird Erfahrungswerte benötigen, das Konvergieren bei lokalen Minima führt zu einer zufälligen Komponente. Die Bildung von Tf-idf-Clustern wird also ein iterativer Prozess sein.

---

[84] Vgl. Jain, S. 6 f.
[85] Letzteres, kmeans++ genannt, ist die Standardkonfiguration von scikit-learn.
[86] Vgl. ebd., S. 8.
[87] Auch für die sinnvolle Anwendung von Tf-idf muss eine Mindestmenge von Text vorhanden sein. Denn es müssen Unterscheidungen zwischen allgemeinem und spezifischem Vokabular getroffen werden, was mit nur wenigen Dokumenten nicht möglich ist. Ohne konkrete Zahlen benennen zu können, wird hier aber die Hypothese aufgestellt, dass dafür deutlich weniger Text als für die im folgenden Kapitel vorgestellte Methode Doc2Vec benötigt wird. Die These wird in Kapitel 5 überprüft.

3.10 Semantische Cluster mit Doc2Vec

Ein weiterer Versuch, eine semantische Dimension in den Textdaten zu erfassen, wird mit den Verfahren Word2Vec[88] und Doc2Vec[89] unternommen. Im Folgenden werden zunächst die beiden Algorithmen erläutert. Danach werden die Besonderheiten beim Clustern der daraus resultierenden Vektoren diskutiert.

Wie im vorhergegangenen Kapitel liegt auch hier der Gedanke zugrunde, den Inhalt eines Textdokuments über einen Vektor von die Worte repräsentierenden Zahlenwerten darzustellen. Anders als Tf-idf arbeitet Word2Vec mit einem flachen neuronalen Netz, dem eine Liste tokenisierter Dokumente als Eingabe übergeben wird und das die Wahrscheinlichkeit maximiert, ein Wort basierend auf seinem Kontext vorherzusagen.[90]

3.10.1 Die Erzeugung von Wortvektoren mit Word2Vec

Das neuronale Netz besteht aus drei Ebenen: einer Input-Ebene, einer allen Worten gemeinsamen Projektions-Ebene und einer Ausgabeebene.[91] Es handelt sich also um ein flaches neuronales Netz ohne versteckte Ebene (im Gegensatz zu den Strukturen beim sogenannten Deep Learning). Diese einfache Struktur zielt auf eine Komplexitätsreduktion ab, denn die versteckten Ebenen sind für den Großteil der Komplexität eines neuronalen Netzes verantwortlich. Durch ihren Wegfall können bei gleichbleibender Qualität der Ergebnisse mehr Eingabedaten verwendet.[92] Die Input- und Output-Ebene haben jeweils die Größe des Gesamtvokabulars des Korpus‘. Im ersten Schritt wird eine Kookkurenz-Matrix erzeugt: Auf der Input-Ebene wird für jedes im betrachteten Textausschnitt vorhandene Wort eine Eins gesetzt, für alle anderen eine Null.[93] Auf der Projektionsebene wird eine bestimmte Zahl an Kontextworten vor und nach dem gesuchten Token in der zentralen Position betrachtet. Dieses wird

---

[88] Vgl. Mikolov, Tomas und Quoc Le: Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*.
[89] Vgl. Mikolov, Tomas, Kai Chen, Greg Corrado et al.: Efficient Estimation of Word Representations in Vector Space.
[90] Die oben erläuterte Architektur beschreibt Word2Vec mit einem Feedforward-Netz und dem Continuous-Bag-of-Words-Modell, da für vorliegendes Projekt mit dieser Konfiguration gearbeitet wird. Nicht genauer beschrieben werden die Alternativen, ein Rekurrentes Feedforward-Netz (vgl. Mikolov, Tomas, Kai Chen, Greg Corrado et al.: Efficient Estimation of Word Representations in Vector Space, S. 3) und das Continuous-Skip-Gram-Modell (vgl. ebd., S. 4).
[91] Vgl. ebd., S. 1 f.
[92] Vgl. ebd.
[93] Vgl. ebd., S. 3.

ausgeblendet und ermittelt, welches Token abhängig von den Kontextworten am wahrscheinlichsten das Gesuchte ist.[94] Die Abfolge der Worte im Kontext wird ignoriert, weshalb Mikolov et al. das Modell „Continuous Bag of Words“ (CBOW) nennen.

Mikolov und seine Kollegen haben ein Sprachmodell mit einem Google-News-Korpus aus sechs Milliarden Tokens trainiert. Es war unter anderem in der Lage, semantische Beziehungen wie die zwischen der männlichen und der weiblichen Form eines Wortes oder die eines Landes zu seiner Hauptstadt abzubilden.[95] Die Herstellung solcher Zusammenhänge anhand von Wortkontexten macht Word2Vec für die vorliegende Problemstellung interessant. Auch wenn die Wortvektoren hauptsächlich ein Nebenprodukt der Erzeugung von Dokumentvektoren sind (siehe Kapitel 3.10.2), können sie beispielsweise die Freitextsuche unterstützen, indem die semantisch ähnlichsten Worte aus dem Vokabular als ähnliche Suchbegriffe vorgeschlagen werden. Es kann außerdem überprüft werden, wie das Suchwort im vorliegenden Korpus verwendet wird; überraschende Bedeutungen werden dabei aber vermutlich selten auftreten.

Um gute Wortvektoren zu erhalten, sind große Trainingskorpora nötig. Es ist fraglich, wie viele Archivbestände die Ansprüche an die dafür nötigen Textmengen erfüllen werden. Word2Vec wird im begleitenden Projekt in der Implementation der Python-Bibliothek gensim[96] genutzt.

3.10.2 Dokumentvektoren als Cluster-Daten

Mikolov hat sein Konzept ausgebaut, um auch semantische Dokumentvektoren erzeugen zu können. Zunächst werden wie bei Word2Vec Wortvektoren für jedes in einem Dokument vorkommende Token erzeugt. Zusätzlich werden auch *paragraph vectors*[97] (im folgenden Dokumentvektoren genannt) erstellt, in denen der Kontext der Worte erhalten bleibt. Um den Dokumentvektor zu erzeugen, werden bei der Vorhersage des nächsten Worts die Wortvektoren des aktuellen Kontexts mit dem

[94] Vgl. Mikolov et al., S. 4.
[95] Vgl. ebd., S. 5–7.
[96] Vgl. Řehůřek, Radim und Petr Sojka: Software Framework for Topic Modelling with Large Corpora.
[97] Mikolov spricht in seinem Paper von Absätzen. Grundsätzlich kann ein *paragraph* eine Einheit von wenigen Sätzen bis hin zu einem ganzen Dokument umfassen (vgl. Mikolov, Tomas und Quoc Le: Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, S. 1).

bisherigen Dokumentvektor verkettet. Dadurch beeinflussen auch Tokens außerhalb des kleinen Kontextfensters die Vorhersage des nächsten Worts, es existiert eine Art Gedächtnis für das gesamte Dokument. Der Output ist eine Matrix von Dokumentvektoren, die als Input für weitere Anwendungen des maschinellen Lernens wie beispielsweise Clustering genutzt werden kann.[98]

Die im vorherigen Kapitel aufgelisteten Probleme beim Clustern hochdimensionaler Daten und der Verwendung von k-means gelten auch hier. Außerdem funktioniert Doc2Vec genau wie Word2Vec voraussichtlich nur bei großen bis sehr großen Korpora.[99] Die Ergebnisse haben das Potential, überraschendere Resulte als Tf-idf-Cluster zu liefern und auch weniger offensichtlich zusammenhängende Dokumente sinnvoll zu gruppieren, wenn genügend Daten vorhanden sind. Bei kleinen bis mittelgroßen Beständen werden vermutlich Tf-idf-Cluster zuverlässigere Ergebnisse erzeugen.

4 Die Erkundung von Dateisammlungen mit der Web-Anwendung „CollectionExplorer“

Um die These zu überprüfen, dass die im vorherigen Kapitel vorgestellten Methoden zur Lösung des Problems der archivischen Bewertung von Dateiablagen beitragen können, wird diese Arbeit von einem Softwareprojekt begleitet. Es handelt sich dabei um eine in Python geschriebene Django-Webanwendung, die im folgenden Kapitel vorgestellt werden soll. Dafür werden zunächst die im Vorfeld getroffenen technischen Entscheidungen erläutert und dann die Benutzung und Funktionalitäten der Anwendung beschrieben.[100]

4.1 Architektur

Zu Beginn des Entwicklungsprozesses mussten mehrere weitreichende Entscheidungen über die zu verwendenden technischen Mittel getroffen werden. Die

---

[98] Vgl. Mikolov und Le, S. 3.

[99] Tests im Vorfeld dieser Arbeit haben mit etwa 100.000 Wikipedia-Artikeln zu guten Ergebnissen bei Doc2Vec-Clustern geführt (siehe Anhang B). Die Ermittlung der Mindest-Bestandsgröße für den sinnvollen Einsatz von Doc2Vec bei realistischen Dateisammlungen ist eines der Ziele dieser Arbeit. Ergebnisse finden sich in Kapitel 5.2.

[100] Die Webanwendung ist zum Zeitpunkt des Abfassens der Arbeit erreichbar unter [gestrichen]. Dauerhafte Erreichbarkeit durch Aufsetzen einer Produktionsumgebung mit Apache und WSGI wurde für diese Arbeit nicht sichergestellt. Die Anwendung läuft in einer weniger robusten Testumgebung, die unter Umständen manuell gestartet werden muss. Anweisungen dafür siehe Anhang A.

Wahl der Programmiersprache fiel auf Python, da eine Vielzahl an Python-Bibliotheken für maschinelles Lernen und maschinelle Sprachverarbeitung existieren. Statt einer Desktopanwendung wurde eine Web App umgesetzt, da die Programmierung für Desktop-Benutzeroberflächen aufwändiger als für Webseiten und der Zugang zu letzteren einfacher ist – sie erfordern nur eine zentrale Installation auf einem von der Zielgruppe erreichbaren Server. Auch für Datenvisualisierungen, die zur Aufbereitung der Analyse-Ergebnisse wichtig sind, gibt es sowohl für Python als auch für das im Web-Frontend verwendete Javascript eine Vielzahl an frei verfügbaren Bibliotheken und Code-Snippets.

Die Umsetzung der Web App erfolgte mit dem Framework Django, welches die Programmierung hoch konfigurierbarer moderner Python-Webanwendungen ermöglicht. Anstelle der standardmäßig mitgelieferten SQLite-Datenbank wird PostgreSQL verwendet, da SQLite keine konkurrierenden Zugriffe auf die Datenbank erlaubt und nur Statements ausführen kann, die maximal tausend Datensätze verändern.[101] Die Nutzung von PostgreSQL ist auch insofern vorteilhaft, als dass dies die einzige Datenbank ist, in der mit einem Befehl massenhaft und schnell neue Django-Objekte unkompliziert und vollständig (inklusive Primärschlüssel) angelegt werden können.[102]

Viele der Verarbeitungs- und Analyseschritte haben eine lange (und mit wachsender Korpusgröße stark ansteigende) Laufzeit. Damit diese lang laufenden Prozesse die Benutzung der Webseite nicht blockieren, werden sie asynchron über die Aufgabenverwaltung celery ausgeführt.

Die Ausgabe erfolgt im Web Browser, entwickelt und getestet wurde die Seite mit Google Chrome. Es ist davon auszugehen, dass manche Funktionalitäten in anderen und vor allem älteren Browsern nicht zur Verfügung stehen werden.

Die Entwicklung sowie die Verarbeitung des ersten Testbestands (siehe Kapitel 5.2.1) wurden auf einem Server der Universität zu Köln vorgenommen. Dabei handelt es sich um eine Maschine mit einem Zehn-Kern-Prozessor mit einer Taktung von 2,3 GHz und 4 GB RAM. Für alle weiteren Tests (siehe Kapitel 5.2.2) wurde ein Server

[101] Vgl. SQLite: Limits In SQLite., Nr. 9.
[102] Vgl. Django Software Foundation: QuerySet API reference.

des Landes Hessen mit einem Acht-Kern-Prozessor, einer Taktung von 2,4 GHz und 16 GB RAM verwendet.

4.2 Datenmodell

Der Anwendung liegt ein simples Datenmodell zugrunde, welches hauptsächlich auf Praktikabilität ausgelegt ist. Ihr Kern sind sogenannte Sammlungen. Jede Sammlung besteht aus beliebig vielen Dokumenten, die jeweils nur einer Sammlung angehören. Dokumente können Duplikate haben; dabei handelt es sich um andere Dokumente, die Beziehung ist bidirektional. Eine Unterklasse von Dokumenten sind Versionen, zu denen ein Originaldokument, ein Zieldokument, ein Ähnlichkeitsmaß und ein Ähnlichkeitswert gehören.

Außerdem gibt es Entitäten, die zu beliebig vielen Dokumenten und Sammlungen gehören können. Entitäten haben eine Bezeichnung, einen Typ (Person, Ort, Stadt, Weiteres) und eine Anzahl. N-Gramme verhalten sich genauso, unterscheiden sich aber (statt in Personen et cetera) in Bi- und Trigramme und zusätzlich nach dem verwendeten Relevanzmaß.

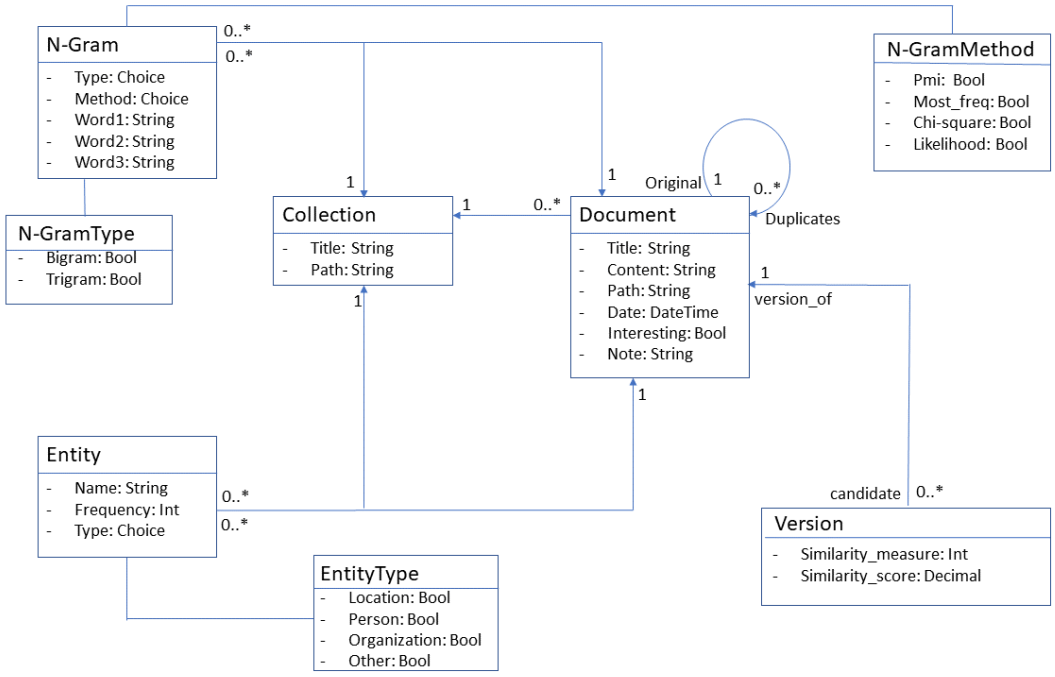


Abbildung 2: Klassendiagramm des CollectionExplorers

Das Datenmodell ergibt sich aus den Bedürfnissen der Anwendung, die sich vorrangig mit Sammlungen und sie beschreibenden Informationsobjekten befasst und nachrangig mit Einzeldokumenten und deren Verfasstheit. Von der Modellierung auf Grundlage eines Standards zur digitalen Archivierung wurde aus mehreren Gründen

abgesehen. Der wichtigste ist, dass diese Standards ab einem späteren Punkt im archivischen Arbeitsprozess einsetzen, nämlich frühestens beim Einlesen in ein digitales Archivierungssystem.[103] Hier wird aber eine Vorstufe dieses Prozesses behandelt; es geht um Daten, von denen der Großteil voraussichtlich kassiert wird. Beziehungen wie Versionen oder Duplikate sind in den Standards nicht vorgesehen, da davon ausgegangen wird, dass keine redundanten Dokumente übernommen werden.[104] Außerdem hätte Konformität zu diesen Standards das Datenmodell stark aufgebläht: Denn auch für möglichst viel Flexibilität angelegte Standards, die nur sehr wenige Pflichtfelder haben, erfordern das Anlegen von für den CollectionExplorer nicht relevanten Informationen.[105] Ein Grund, standardkonform zu arbeiten, hätte die Vorbereitung von Informationspaketen zum Einlesen in digitale Archivierungssysteme sein können. Allerdings besteht kein archivübergreifender Standard dafür. Es hätte also nur Kompatibilität zu einem ausgewählten System bestanden oder es hätten prophylaktisch möglichst viele Daten inkludiert werden müssen. Weiterhin ergibt die Anbindung an digitale Archivierungssysteme erst dann Sinn, wenn der CollectionExplorer tatsächlich für den Einsatz zur Bewertung digitaler Sammlungen reif ist und aus ihm heraus Datensätze in ein digitales Langzeitarchiv importiert werden sollen. Das ist bei diesem Prototyp noch nicht der Fall.

4.3 Workflow für Dateiupload und Textverarbeitung

Um die Anwendung (im Folgenden dem Namen der Django-App folgend CollectionExplorer genannt) zu benutzen, muss zunächst eine Sammlung angelegt werden.

---

[103] Der erste im verbreiteten Standard Open Archival Information System (OAIS) beschriebene Arbeitsschritt ist die Bildung eines Submission Information Package (SIP). Das SIP enthält für die Übernahme ins Archiv vorgesehene Dateien und sie beschreibende Informationen; die Erstellung eines SIP aber ist der auf die Bestandserkundung mit dem CollectionExplorer folgende Schritt. Vgl. CCSDS: Reference Model for an Open Archival Information System (OAIS).

[104] Ein Versuch, ein EU-weites OAIS-konformes SIP-Format zu definieren, erfolgte von 2014 bis 2017 im E-ARK-Projekt. Die dort erarbeitete Spezifikation wird in Ermangelung deutschlandweiter Standards als Referenz genutzt. Vgl. Kärberg, Tarvo, Karin Bredenberg, Björn Skog et al.: E-ARK SIP Pilot Specification (revision of D3.2, main part of the D3.3).

[105] Das E-ARK-SIP-Format beispielsweise fordert für jede Datei die Angabe des MIME Types, die Dateigröße sowie eine Prüfsumme mit Typ und Wert. Vgl. ebd., S. 29–32.

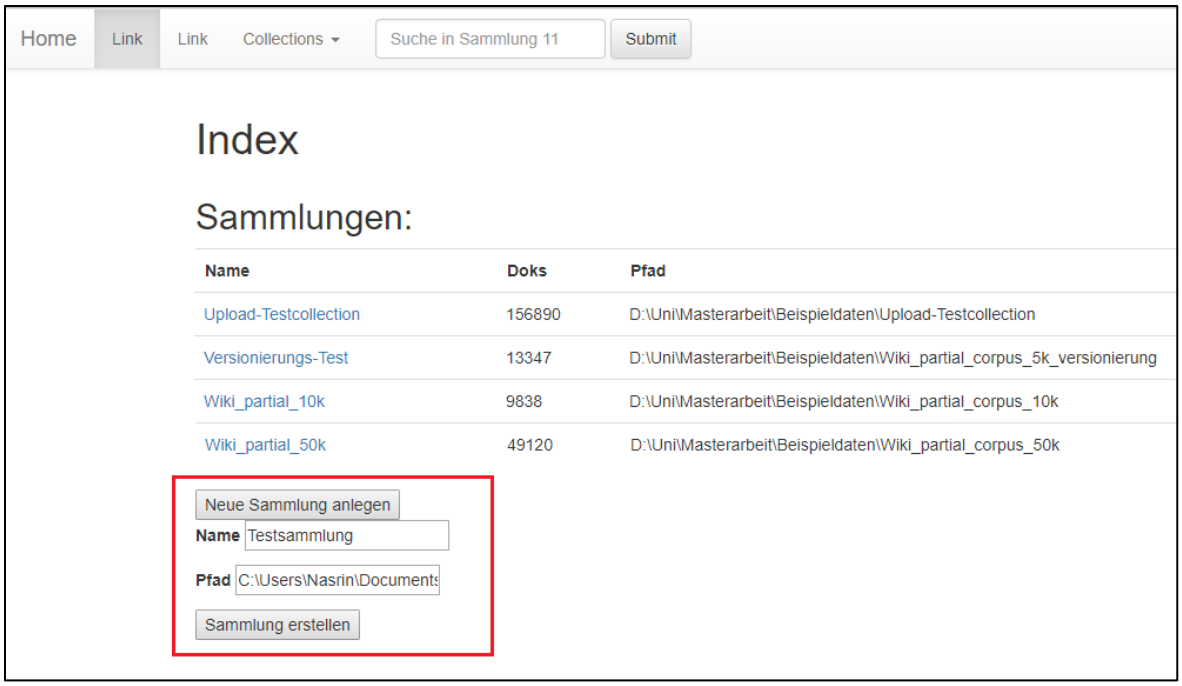


Abbildung 3: Neue Sammlung anlegen

Dieser Sammlung werden im nächsten Schritt Dokumente hinzugefügt, entweder durch das Hochladen lokaler Dateien oder das Lesen von Dateien auf dem Server. Es können die Dateiformate .csv, .doc, .docx, .epub, .htm, .html, .json, .pdf, .pptx, .tsv, .txt, .xls und .xlsx eingelesen werden (siehe Kapitel 3.1). Von diesen Dokumenten werden Dateiname, Text, Pfad und Datum erfasst. Nur Name und (vor allem) Volltext sind für die weitere Verarbeitung relevant.

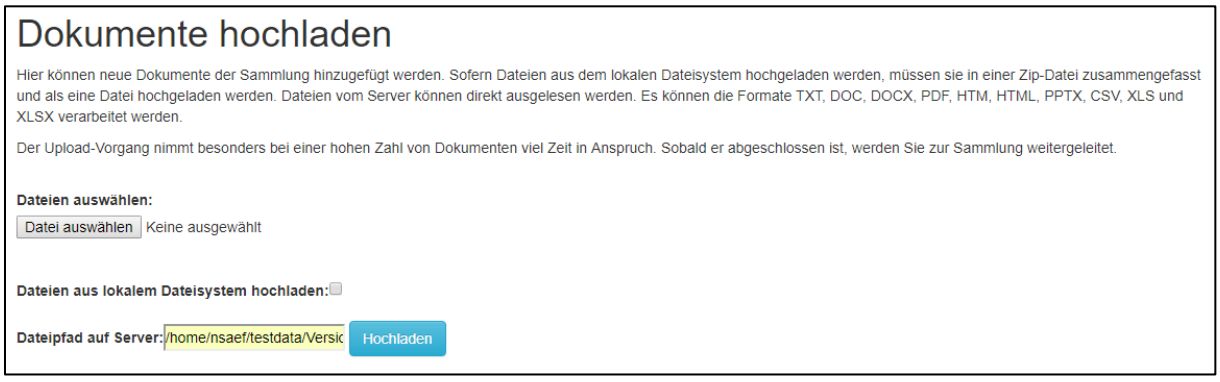


Abbildung 4: Dokumente hochladen

Da der CollectionExplorer für die Verarbeitung sehr großer Mengen von Dokumenten konzipiert wurde, die entsprechende Laufzeiten verursacht, müssen nach dem Hochladen verschiedene Schritte zur Vorverarbeitung durchgeführt werden (siehe Kapitel 3.2). Dabei handelt es sich vor allem, aber nicht ausschließlich, um die

Erzeugung anderer Repräsentationen der Texte, die algorithmisch verarbeitet werden können.

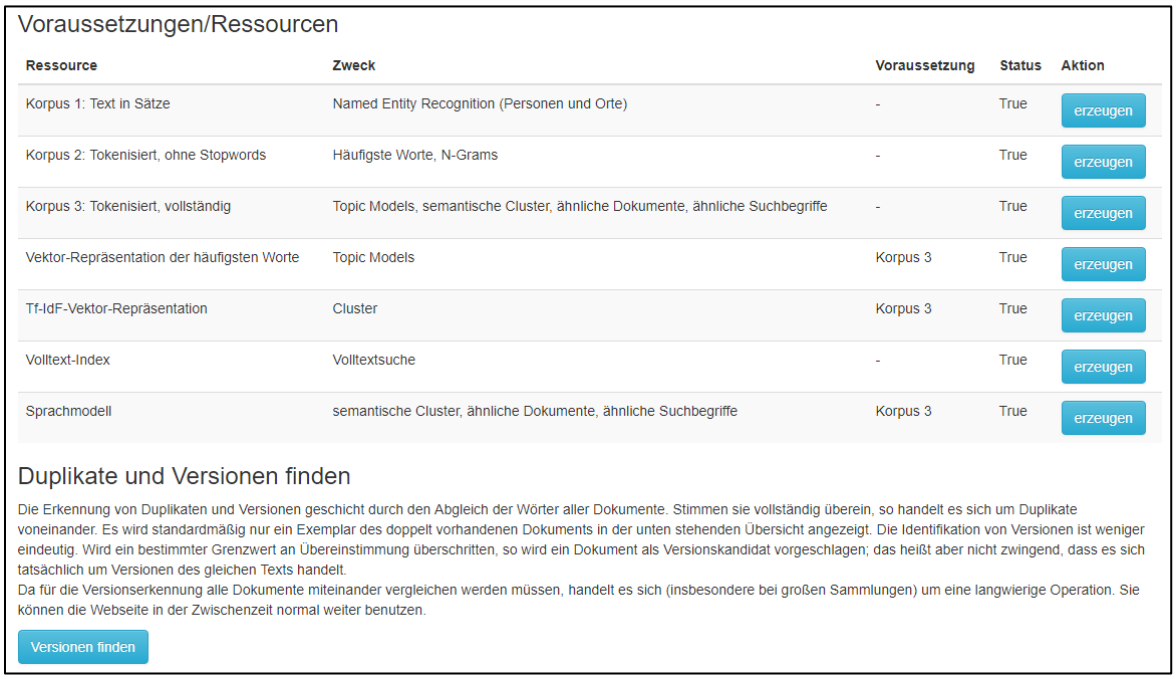


Abbildung 5: Vorverarbeitungsschritte durchführen

Für die NER müssen alle Dokumente in Sätze gespalten, für die häufigsten Worte und N-Gramme müssen sie tokenisiert ohne Stoppworte vorliegen.[106] Um Vektor-Repräsentationen für Topic Models, Tf-idf und das Doc2Vec-Sprachmodell sowie den Index für die Suche zu erzeugen, wird der vollständige tokenisierte Text (einschließlich der Stoppworte) benötigt. Die Ergebnisse all dieser Vorverarbeitungsschritte werden in je einer (oder, falls diese zu groß würde, mehreren) Datei(en) zwischengespeichert, so dass später schnell auf sie zugegriffen werden kann. Außerdem können nach Erzeugung des tokenisierten Korpus‘ (inklusive Stoppworte) der Volltext-Index für die Suche erstellt sowie Versionen und Duplikate identifiziert werden. Sie werden bei der Ansicht eines Dokuments entsprechend gekennzeichnet, Duplikate werden in der Übersicht über alle Dokumente außerdem standardmäßig ausgeblendet.

Für einzelne Dokumente sind diese Vorverarbeitungsschritte nicht nötig. Bei ihnen werden zwar beispielsweise auch Worthäufigkeiten ermittelt und Named Entitys

[106] Alternativ hätten die Stoppworte auch während der Laufzeit des Programms herausgefiltert werden können, um in der Vorverarbeitung nur ein tokenisiertes Korpus zu erzeugen. Allerdings ist insbesondere das Finden von N-Grammen eine ressourcenintensive Aufgabe, die von der Verkleinerung des Korpus‘ im Vorfeld stark profitieren sollte.

erkannt, aber aufgrund der (vergleichsweise) geringen Textmenge pro Dokument geschieht das on-the-fly beim Aufruf.

4.4 Methoden der Datenanalyse

Abbildung 6: Word Cloud der häufigsten Begriffe

Sobald die Vorverarbeitungsschritte abgeschlossen sind, stehen verschiedene Analysetools zur Verfügung, die je auf einer eigenen Seite angezeigt werden.

Die statistische Analyse beinhaltet die häufigsten Begriffe (dargestellt als Word Cloud und in Listenform) sowie nach verschiedenen Relevanzkriterien ausgewählte Bi- und Tri-Gramme. Named Entitys werden sortiert nach Typ (Organisation, Ort, Person, Weiteres) und Häufigkeit angezeigt und sind durchsuchbar.

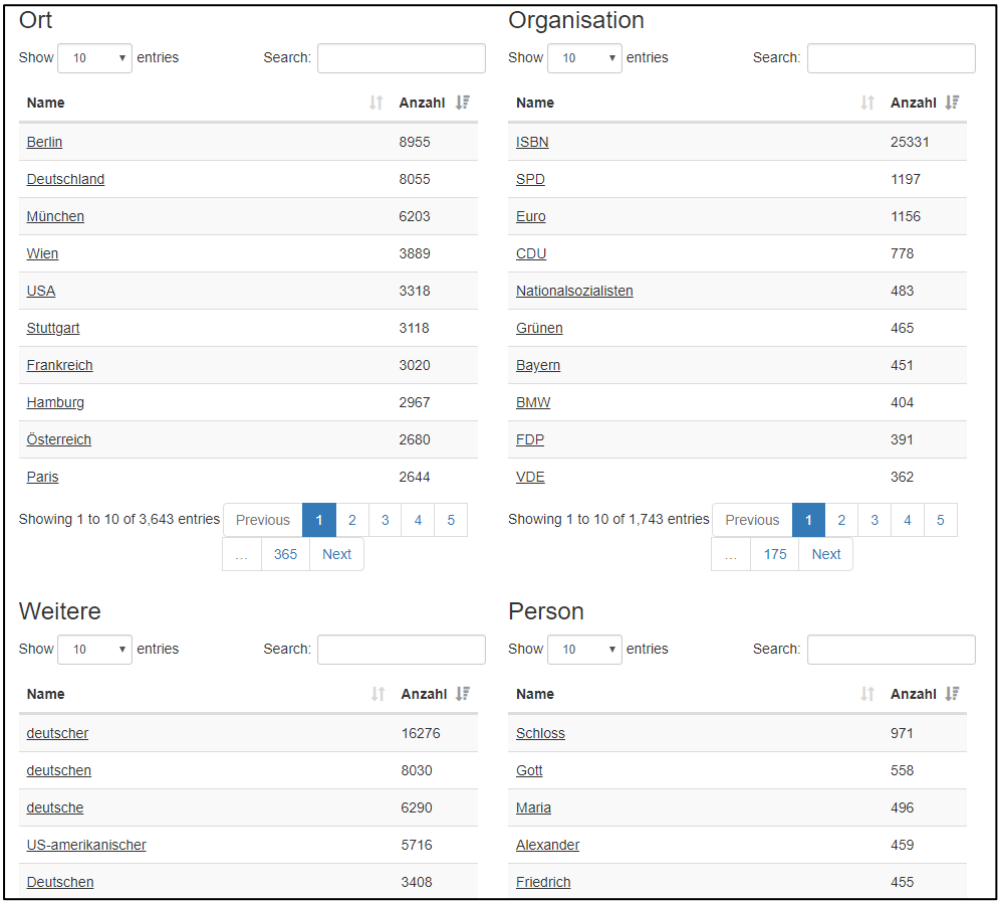


Abbildung 7: Erkannte Eigennamen

Für die semantische Analyse müssen zunächst die gewünschte Anzahl der Cluster oder Topics und die Methode (Doc2Vec-Cluster, Tf-idf-Cluster oder Topic Modelling) gewählt werden. Sobald die Verarbeitung abgeschlossen ist, werden die Cluster oder Topics und die zu ihnen gehörenden Dokumente angezeigt.

Abbildung 8: Clustererstellung

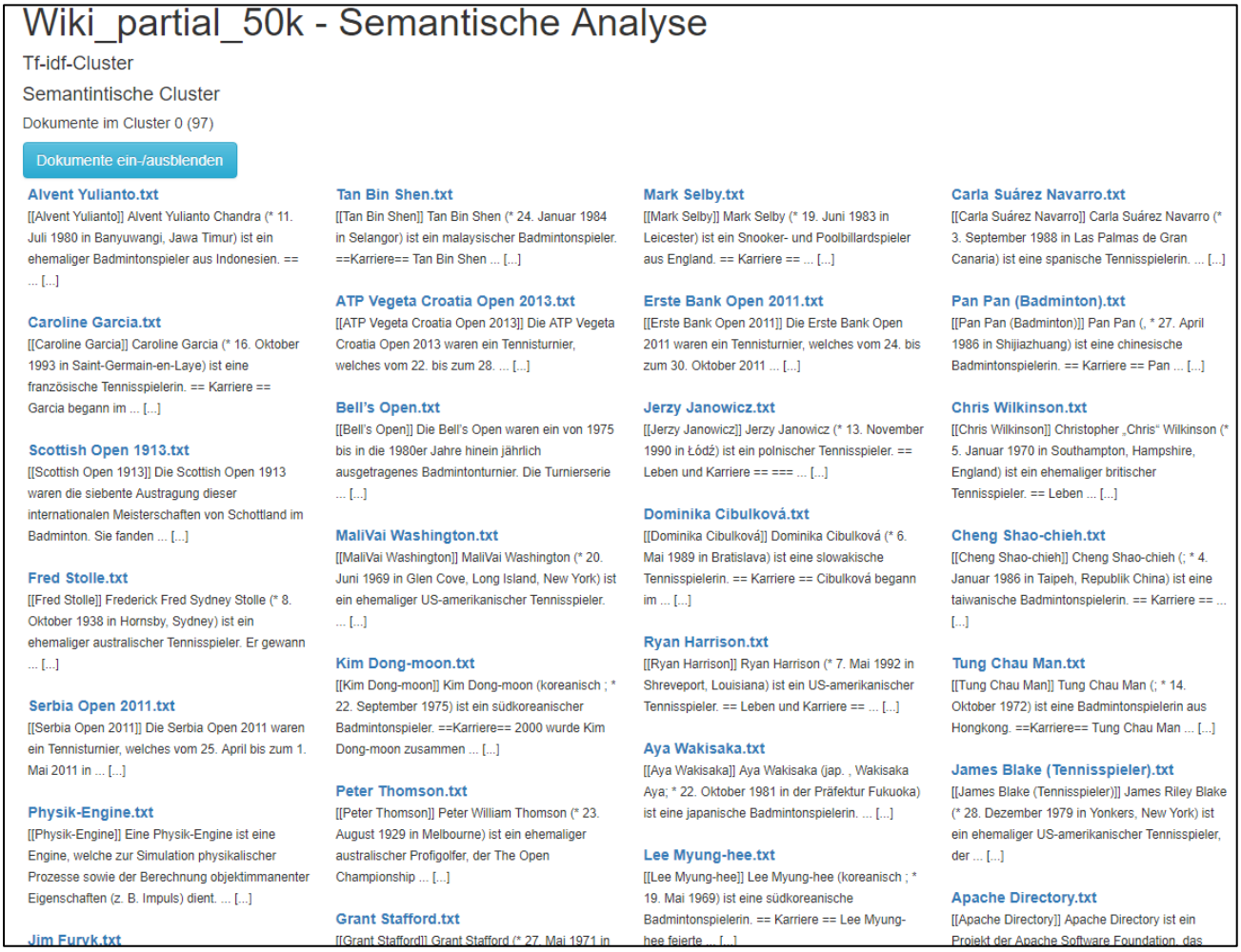


Abbildung 9: Clusterergebnisse

Die Analyse des Vokabulars erlaubt zwei Untersuchungen. Die erste stützt sich auf eine Eigenheit von Word2Vec: Mit den Ergebnis-Vektoren kann gerechnet werden, so gilt beispielsweise: $König - Mann + Frau = Königin$ (vorausgesetzt, die Worte waren in den Trainingsdaten enthalten und das Sprachmodell ist gut genug). Außerdem können die zehn ähnlichsten Begriffe zu einem Suchwort gefunden werden. Dieses Werkzeug kann nicht nur zu interessanten Ergebnissen führen, es kann auch zur Überprüfung der Doc2Vec-Ergebnisse genutzt werden. Wenn hierbei keine sinnvollen Ergebnisse auftauchen, ist nicht davon auszugehen, dass die semantischen Dokument-Cluster mehr als zufällig verteilt sind.

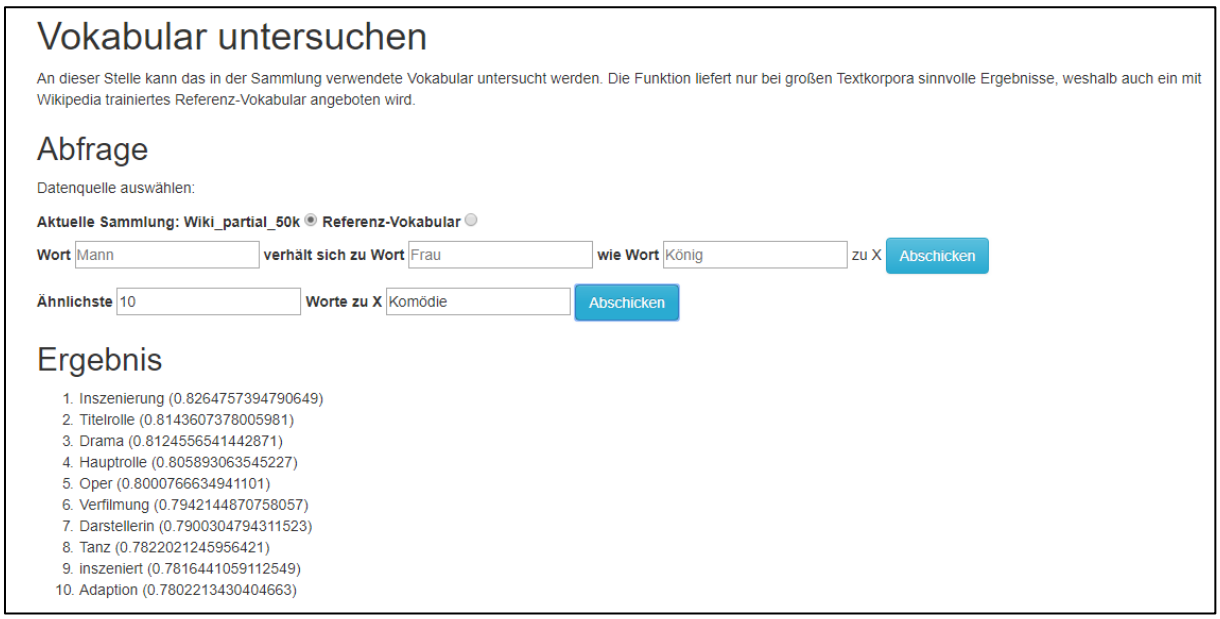
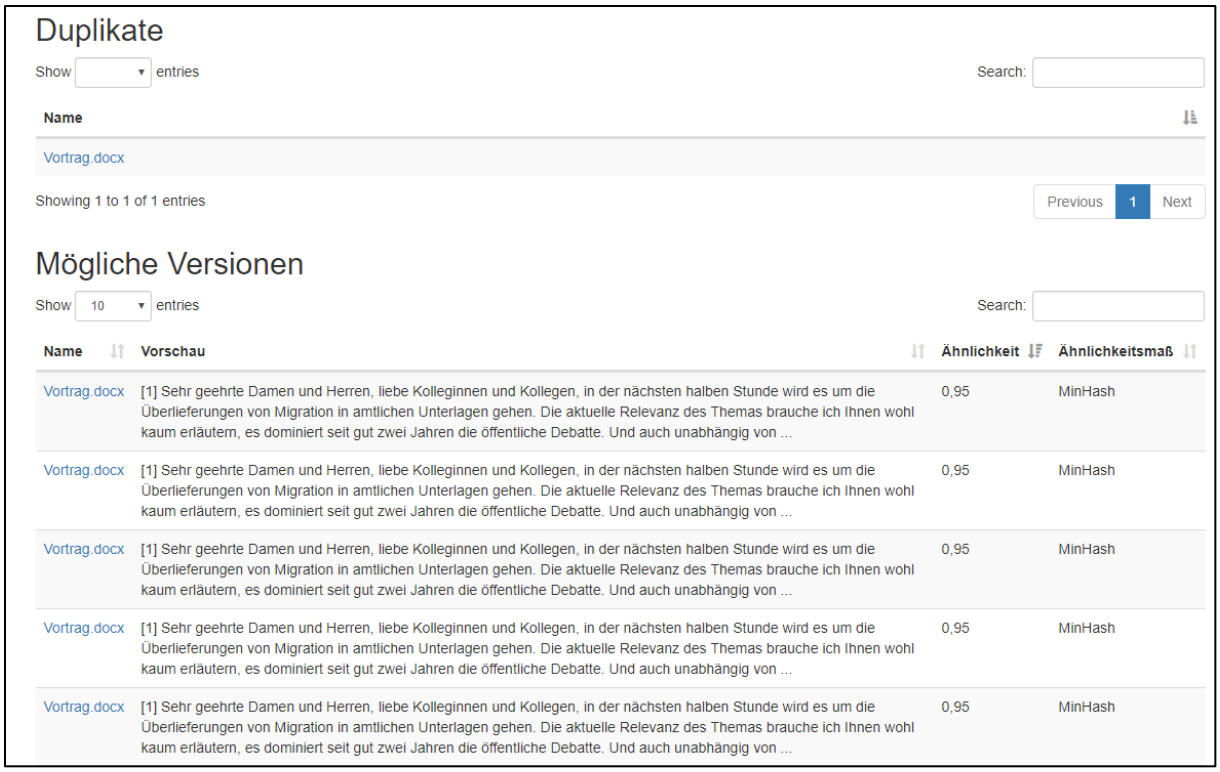


Abbildung 10: Anzeige ähnlicher Begriffe

Auf Dokumentebene gibt es zusätzlich zu den oben genannten Analysen noch die Anzeige ähnlicher Dokumente (auf Basis von Doc2Vec) sowie von Duplikaten und möglichen Versionen. Duplikate und Versionskandidaten werden von den ähnlichen Dokumenten ausgeschlossen, so dass alle drei Kategorien unterschiedliche Dokumente ausgeben.



Duplikate

Show entries Search:

Name
Vortrag.docx

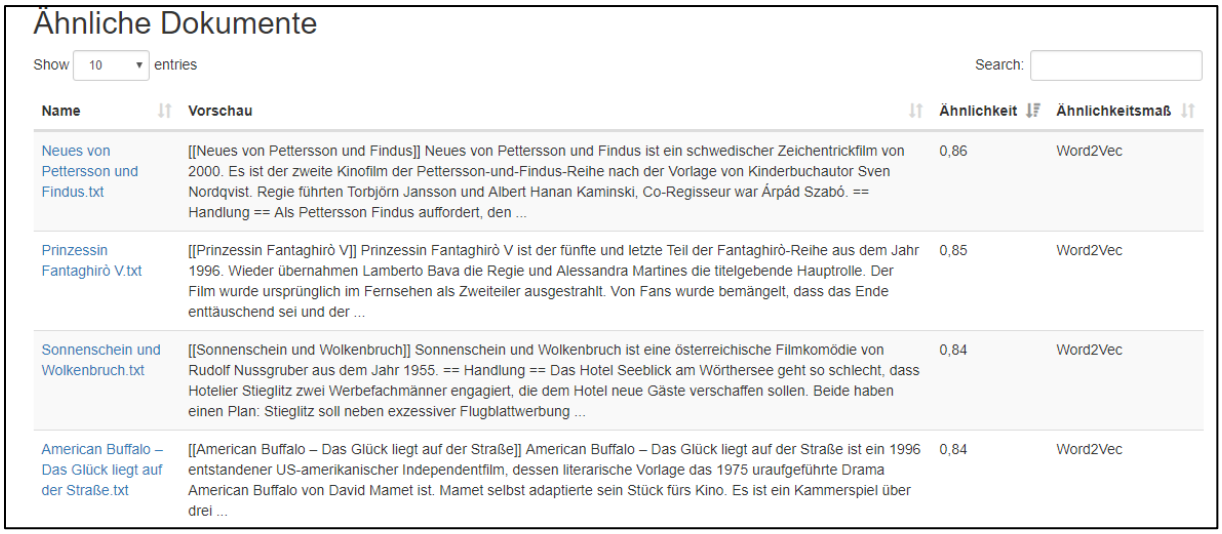
Showing 1 to 1 of 1 entries Previous 1 Next

Mögliche Versionen

Show 10 entries Search:

Name	Vorschau	Ähnlichkeit	Ähnlichkeitsmaß
Vortrag.docx	[1] Sehr geehrte Damen und Herren, liebe Kolleginnen und Kollegen, in der nächsten halben Stunde wird es um die Überlieferungen von Migration in amtlichen Unterlagen gehen. Die aktuelle Relevanz des Themas brauche ich Ihnen wohl kaum erläutern, es dominiert seit gut zwei Jahren die öffentliche Debatte. Und auch unabhängig von ...	0,95	MinHash
Vortrag.docx	[1] Sehr geehrte Damen und Herren, liebe Kolleginnen und Kollegen, in der nächsten halben Stunde wird es um die Überlieferungen von Migration in amtlichen Unterlagen gehen. Die aktuelle Relevanz des Themas brauche ich Ihnen wohl kaum erläutern, es dominiert seit gut zwei Jahren die öffentliche Debatte. Und auch unabhängig von ...	0,95	MinHash
Vortrag.docx	[1] Sehr geehrte Damen und Herren, liebe Kolleginnen und Kollegen, in der nächsten halben Stunde wird es um die Überlieferungen von Migration in amtlichen Unterlagen gehen. Die aktuelle Relevanz des Themas brauche ich Ihnen wohl kaum erläutern, es dominiert seit gut zwei Jahren die öffentliche Debatte. Und auch unabhängig von ...	0,95	MinHash
Vortrag.docx	[1] Sehr geehrte Damen und Herren, liebe Kolleginnen und Kollegen, in der nächsten halben Stunde wird es um die Überlieferungen von Migration in amtlichen Unterlagen gehen. Die aktuelle Relevanz des Themas brauche ich Ihnen wohl kaum erläutern, es dominiert seit gut zwei Jahren die öffentliche Debatte. Und auch unabhängig von ...	0,95	MinHash
Vortrag.docx	[1] Sehr geehrte Damen und Herren, liebe Kolleginnen und Kollegen, in der nächsten halben Stunde wird es um die Überlieferungen von Migration in amtlichen Unterlagen gehen. Die aktuelle Relevanz des Themas brauche ich Ihnen wohl kaum erläutern, es dominiert seit gut zwei Jahren die öffentliche Debatte. Und auch unabhängig von ...	0,95	MinHash

Abbildung 11: Erkannte Duplikate und Versionskandidaten



Ähnliche Dokumente

Show 10 entries Search:

Name	Vorschau	Ähnlichkeit	Ähnlichkeitsmaß
Neues von Pettersson und Findus.txt	[[Neues von Pettersson und Findus]] Neues von Pettersson und Findus ist ein schwedischer Zeichentrickfilm von 2000. Es ist der zweite Kinofilm der Pettersson-und-Findus-Reihe nach der Vorlage von Kinderbuchautor Sven Nordqvist. Regie führten Torbjörn Jansson und Albert Hanan Kaminski, Co-Regisseur war Árpád Szabó. == Handlung == Als Pettersson Findus auffordert, den ...	0,86	Word2Vec
Prinzessin Fantaghirò V.txt	[[Prinzessin Fantaghirò V]] Prinzessin Fantaghirò V ist der fünfte und letzte Teil der Fantaghirò-Reihe aus dem Jahr 1996. Wieder übernahmen Lamberto Bava die Regie und Alessandra Martines die titelgebende Hauptrolle. Der Film wurde ursprünglich im Fernsehen als Zweiteiler ausgestrahlt. Von Fans wurde bemängelt, dass das Ende enttäuschend sei und der ...	0,85	Word2Vec
Sonnenschein und Wolkenbruch.txt	[[Sonnenschein und Wolkenbruch]] Sonnenschein und Wolkenbruch ist eine österreichische Filmkomödie von Rudolf Nussgruber aus dem Jahr 1955. == Handlung == Das Hotel Seeblick am Wörthersee geht so schlecht, dass Hotelier Stieglitz zwei Werbefachmänner engagiert, die dem Hotel neue Gäste verschaffen sollen. Beide haben einen Plan: Stieglitz soll neben exzessiver Flugblattwerbung ...	0,84	Word2Vec
American Buffalo – Das Glück liegt auf der Straße.txt	[[American Buffalo – Das Glück liegt auf der Straße]] American Buffalo – Das Glück liegt auf der Straße ist ein 1996 entstandener US-amerikanischer Independentfilm, dessen literarische Vorlage das 1975 uraufgeführte Drama American Buffalo von David Mamet ist. Mamet selbst adaptierte sein Stück fürs Kino. Es ist ein Kammerspiel über drei ...	0,84	Word2Vec

Abbildung 12: Ähnliche Dokumente nach Doc2Vec

Eine Volltextsuche ermöglicht einfache Suchen über den Dokumentinhalt, mit den Suchergebnissen werden außerdem ähnliche Suchbegriffe auf Basis von Word2Vec vorgeschlagen.

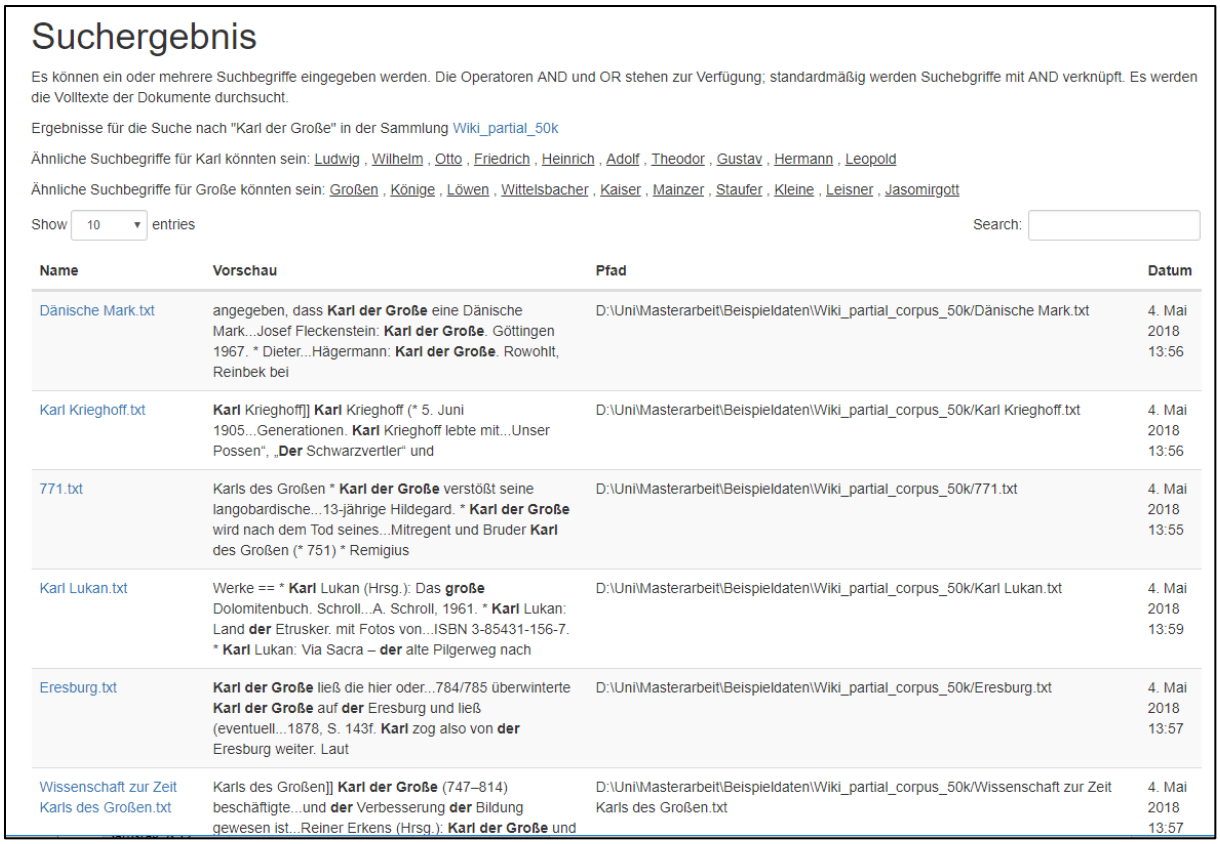


Abbildung 13: Volltextsuche

5 Evaluation des CollectionExplorers

Der CollectionExplorer hat zum Ziel, den Inhalt großer Textsammlungen greifbar zu machen. Mit seiner Hilfe soll die Nutzerin, zum Beispiel eine mit Bewertung befasste Archivarin, sich einen Überblick über die in den Dokumenten einer Sammlung behandelten Themen verschaffen und so einen Ansatz für den weiteren Umgang mit ihr finden. Ob der CollectionExplorer dazu geeignet ist, dieses Ziel zu erreichen, soll im Folgenden anhand mehrerer Testbestände mit unterschiedlichen Merkmalen überprüft werden. Begonnen wird die Überprüfung mit einer allgemeinen Bewertung der Benutzbarkeit der Anwendung, bevor auf die Testbestände eingegangen wird.

5.1 Benutzbarkeit der Anwendung

Eine Voraussetzung für die Nutzung jeglicher Softwareanwendung ist die allgemeine Benutzbarkeit hinsichtlich Themen wie Usability und Performance. Ob und inwiefern diese beim CollectionExplorer gegeben ist, wird nachfolgend überprüft.

5.1.1 Usability

Beim CollectionExplorer handelt es sich um einen Prototyp, der vor allem die Nützlichkeit ausgewählter computerlinguistischer Methoden für die archivische Bewertung überprüfen soll. Die Benutzung außerhalb dieses Forschungskontexts wäre zwar wünschenswert, ist aber nicht vorgesehen, da die Entwicklung einer für Laien komfortabel benutzbaren Anwendung über den den Rahmen einer Masterarbeit hinausgehen würde.

Grundlegende Usability wird über ein mit Bootstrap gestaltetes (und somit optisch vielen Nutzern vertrautes) User Interface gewährleistet. Dessen Übersichtlichkeit schwankt je nach Funktionalität: Die Darstellung von Dokumentclustern beispielsweise ist (naturgemäß, da hunderte bis tausende Dokumente gemeinsam präsentiert werden müssen) weniger übersichtlich als der Überblick über erkannte Named Entitys, die sortiert nach Anzahl in einer Tabelle pro Typ dargestellt werden (siehe Abbildung 9 und Abbildung 7). Alle Funktionalitäten werden von kurzen Erläuterungen zu ihrer Benutzung und der Aussagekraft ihrer Ergebnisse begleitet. Fehler (beispielsweise durch fehlende Vorverarbeitung) werden zwar vielerorts abgefangen, aber nicht immer im Detail erklärt.

Ausbaufähig sind insbesondere die Verwaltung der Daten und das Feedback zu asynchron laufenden Prozessen. Den Nutzern werden kaum Funktionalitäten zur Verwaltung von Sammlungen und Dokumenten angeboten; im Entwicklungsprozess erfolgte sie über Djangos Datenbankschnittstelle auf der Python-Konsole. Im Frontend können bloß Sammlungen erzeugt und alle in ihnen enthaltenen Dokumente gelöscht werden. Auch celery, welches asynchron laufende Prozesse verwaltet, wird komplett über die Konsole gesteuert. Es gibt außerhalb dieser kein Feedback, an welcher Stelle des Programms sich eine Aufgabe gerade befinde und ob sie gelungen oder gescheitert ist. Außerdem stürzt es nach Abschluss von Aufgaben, für die sehr große Datenmengen gehandhabt werden müssen, gelegentlich ab. Dies erfordert einen Neustart von celery mit dazwischen geschaltetem Bereinigen der Aufgabenliste. Für all das gibt es kein Interface im Frontend, es muss direkt auf der Konsole des ausführenden Servers erledigt werden.

Grundlegende Usability ist also gewährleistet, und ein unbedarfter Nutzer kann nach abgeschlossener Vorverarbeitung alle explorativen Funktionalitäten des CollectionExplorers nutzen. Um aber alle Arbeitsschritte auszuführen oder eine

Sammlung zu verändern, sind Python-Kenntnisse, Zugang zum Server und grundlegende Kommandos für celery nötig.

5.1.2 Performance

Anwendungen aus dem Bereich maschinelle Sprachverarbeitung haben meist eine hohe Komplexität. Selbst kleine Sammlungen von circa fünftausend Dokumenten benötigen für die verschiedenen Vorverarbeitungsschritte oft einige Minuten, bei zehntausend oder mehr Dokumenten können die Verarbeitungszeiten auf eine bis mehrere Stunden steigen. In dieser Zeit bleibt die Anwendung benutzbar, da die Aufgaben asynchron ausgeführt werden; außerdem können (abhängig von der Hardware) mehrere asynchrone Aufgaben parallel verarbeitet werden. Trotzdem müssen hohe Verarbeitungszeiten bei der Benutzung des CollectionExplorers eingeplant werden.

Auch die Ladezeiten der Web-Anwendung sind oft hoch, selbst, wenn nicht wie beim Laden eines Dokuments noch Analyseschritte (wie zum Beispiel Tokenisierung und Ermittlung von Worthäufigkeiten) ausgeführt werden müssen. Das liegt an den zu verarbeitenden Datenmengen: Die Erzeugung einer Tabelle mit Informationen zu über zehntausend Dokumenten benötigt Zeit. Es wäre möglich gewesen, den Ladezeiten der Web-Anwendung zum Beispiel durch synchrones Laden nur der ersten Tabellenelemente (sei es bei Dokumenten, Named Entitys oder anderen Daten) zu begegnen, und den Rest asynchron nachladen zu lassen. Solche Optimierung spielte aber ähnlich wie die im vorherigen Abschnitt genannten Usability-Probleme beim Bau eines Prototyps keine übergeordnete Rolle.

Als Herausforderung für die Analyse stellte sich die Balance zwischen Zeitkomplexität und Platzkomplexität (also der für die Ausführung nötigen Zeit beziehungsweise dem Speicherplatz) heraus. Während anfangs zugunsten der Laufzeit optimiert wurde, stellte sich bei Tests mit echten Datensätzen heraus, dass einige Funktionen mit dem auf der vorhandenen Hardware nutzbaren Arbeitsspeicher nicht ausführbar waren. Deshalb wurden insbesondere speicherplatzintensive Funktionalitäten wie die Erkennung von Versionen (für die jedes Dokument mit jedem anderen Dokument verglichen werden muss) oder die NER (die mit dem weniger

effizienten Java anstelle vom auf C basierenden Python durchgeführt wird) zugunsten geringerer Platzkomplexität umstrukturiert.[107]

Aufgrund der Vertraulichkeit der verarbeiteten Testdaten (siehe Kapitel 5.2.2) konnte das Komplexitätsproblem nicht durch die Nutzung von auf die Verarbeitung großer Datenmengen ausgerichteten Cloud-Diensten wie Amazon Web Services, Microsoft Azure oder Google Cloud Platform gelöst werden. Die Daten durften das Netzwerk des Lands Hessen nicht verlassen, ergo musste mit der vorhandenen Rechenkapazität gearbeitet werden.[108]

Zusammenfassend muss wie bei der Usability konstatiert werden, dass die Performance an vielen Stellen verbessert werden könnte. Aber auch hier gilt: Für die Machbarkeitsstudie waren Performance-Fragen nur insofern ausschlaggebend, dass die Verarbeitung möglichst großer Datenmengen in endlicher Zeit auf durchschnittlicher Hardware möglich sein musste. Hohe Ladezeiten im Browser oder Verarbeitungsschritte, die bei sehr großen Datenmengen über Nacht laufen müssen, sind unangenehm, stehen dem Forschungsziel aber nicht im Weg.

5.1.3 Datenmodell

Das in Kapitel 4.2 vorgestellte Datenmodell taugt weitgehend zur Problemlösung. Schwierig ist vor allem der Umgang mit Duplikaten: Logisch ist es korrekt, dass es sich dabei um eine wechselseitige Beziehung handelt und beide Dokumente, die Teil einer Duplikatbeziehung sind, als Duplikat hinterlegt sind. Praktisch aber soll eines der beiden Duplikate regulär verarbeitet werden, während das zweite als Duplikat beispielsweise ausgeblendet oder bei Verarbeitungsschritten ignoriert wird. Dies musste über – die Performance beeinträchtigende – Umwege erreicht werden.

---

[107] Strategien hierfür waren zum Beispiel die Verarbeitung in Paketen anstelle der gemeinsamen Verarbeitung einer gesamten Sammlung. Die Erzeugung vieler Django-Objekte in einem einzelnen Befehl ist deutlich performanter als die einzelne Bearbeitung, dafür müssen aber zunächst alle für die Erzeugung notwendigen Daten gleichzeitig in den Speicher geladen werden, was nicht immer möglich war.

[108] Diese Problematik stellt sich nicht nur in vorliegender Arbeit, sondern für öffentliche Archive allgemein. Die Verwendung externer Dienstleister für das Hosting ist nicht möglich, wenn die zu verarbeitenden Datensätze gesetzlich geschützte personenbezogene Daten beinhalten oder staatlichen Geheimhaltungsvorschriften unterliegen. Es muss also mit den innerhalb der eigenen Verwaltung verfügbaren Mitteln gearbeitet werden, und dort sind Hochleistungs-Computing-Cluster eher die Ausnahme als die Regel.

Umgekehrt verhält es sich mit Entitäten und N-Grammen. Dort wurde eine pragmatische Lösung umgesetzt, nämlich die separate Betrachtung von Dokument und Sammlung. Es wäre denkbar, logisch und in manchen Szenarien auch effizient gewesen, die Named Entitys und die häufigsten N-Gramme pro Dokument zu erfassen und bei der Sammlung die Summe aller Entitäten und N-Gramme aus den Dokumenten anzugeben. Dann müsste nur ein Erkennungsprozess laufen, und nicht nur die Sammlung, sondern auch alle Dokumente wären mit Informationen zu Named Entitys und den häufigsten N-Grammen angereichert. Auch logisch sind die N-Gramme und Entitäten Teil der Dokumente, für die die Sammlung nur ein Behälter ist. Aber erstens würde dies die Verarbeitung stark verlangsamen, weil je nur ein Dokument auf einmal verarbeitet würde. Zweitens müssten die N-Gramme nach anderen Relevanzmaßen als der Häufigkeit weiterhin für den Gesamtbestand gesucht werden, da hierfür der Gesamttext aller Dokumente der Sammlung benötigt wird. Und drittens wird bei großen Sammlungen vermutlich nur ein Bruchteil der Dokumente je aufgerufen, so dass der praktische Vorteil dieses Vorgehens – nämlich, dass Entitäten und N-Gramme auf Dokumentebene bereits erkannt sind – nicht sehr groß wäre.

5.2 Verarbeitung von Testbeständen

Um den Nutzen des CollectionExplorers für seinen eigentlichen Zweck zu testen, wurden fünf Testbestände herangezogen. Diese unterscheiden sich nicht nur in ihrem Inhalt, sondern auch in ihrer Größe und Zusammensetzung. Die genutzten Testbestände und ihre Charakteristika werden in den folgenden Abschnitten zunächst kurz beschrieben. Dann wird überprüft, inwiefern die angewandten Methoden zur Erreichung des anfangs gesetzten Ziels taugen.

5.2.1 Wikipedia-Artikel als Demonstrationsbestand

Bevor authentische Testdaten zur Verfügung standen, wurde die Entwicklung mit zufällig ausgewählten Wikipedia-Artikeln begonnen. Es wurden Korpora in den Größen 6.000, 10.000 und 50.000 Dokumente gebildet, die jeweils nur die Überschrift und den Artikeltext, aber keine Metadaten, Kategorien, Inhaltsverzeichnisse oder ähnliches enthalten.

Die Wikipedia-Korpora sind aus zwei Gründen über die Testphase hinaus Teil vorliegender Arbeit geblieben. Zum einen unterliegen authentische Daten für den behandelten Anwendungsfall strengen Datenschutzregelungen und können somit

nicht ohne Weiteres veröffentlicht oder zitiert werden. Um trotzdem die Funktionalitäten des CollectionExplorers anhand von zitierfähigen Beispielen evaluieren zu können, müssen andere, frei verfügbare Daten genutzt werden. Die Wikipedia-Daten können nicht nur in vorliegender Arbeit zitiert, sondern auch in der Web-Anwendung betrachtet und selbstständig analysiert werden.

Weiterhin soll die Aussagekraft der Ergebnisse anhand eines einfach einzuschätzenden Bestands überprüft werden können. Auch wenn die genauen enthaltenen Artikel nicht bekannt sind, wird schnell klar, ob der allgemeine Inhalt der Sammlung gut reflektiert wird, denn ihre Struktur und Natur sind vertraut: Es handelt sich um eine Enzyklopädie, die Einträge beispielsweise zu Personen, Orten, historischen Ereignissen, (Populär-)Kultur oder Technik und Wissenschaft enthält. Mit authentischen Daten wird dies ungleich schwerer, da die Bestandsstruktur in der Regel nicht bekannt sein wird.

Es gibt aber auch Nachteile bei der Nutzung von Wikipedia als Testdaten. Die Artikel sind großteils ähnlich in Länge und Stil, sie sind vergleichsweise kurz, und sie liegen in einem einheitlichen Format (nämlich als UTF-8 kodierte txt-Dateien) vor. Aufgrund der Kürze der Ergebnisse werden sehr viele Artikel benötigt, um zufriedenstellende – oder zumindest mit realistischen Anwendungsfällen vergleichbare – Ergebnisse zu erzielen. Im Folgenden wird daher eine Testsammlung mit 50.000 Artikeln herangezogen, die von der Gesamtzahl der Zeichen einem kleinen Archivbestand entspricht (siehe den in Kapitel 5.2.2.2 behandelten Bestand).

Die Sammlung setzt sich folgendermaßen zusammen:

- Gesamtzahl Dokumente: 49.120
- Textdokumente (DOC, DOCX, TXT): 49.025
- Durchschnittliche Dokumentlänge: 3362.4 Zeichen inkl. Leerzeichen

Worthäufigkeiten

Die (in Abbildung 6 dargestellten) häufigsten Worte sind für die Wikipedia typische Begriffe, beispielsweise (Jahres-)Zahlen, Monatsnamen und häufige Überschriften wie „Weblinks“, „Geschichte“ oder „Literatur“. Das spiegelt zwar den Inhalt der Sammlung gut wieder, ist aber fürs Erfassen ihres Inhalts nicht hilfreich, da es keine neuen Informationen beiträgt. Sofern sich dies bei anderen Beständen bestätigt, ist nicht davon auszugehen, dass die häufigsten Worte viel zum besseren Verständnis einer

Sammlung beitragen. Sie könnten dann eine angedachte Strukturierung bestätigen, würden aber selten einen Informationsgewinn erzeugen.

N-Gramme

Die häufigsten Bi- und Trigramme ähneln den häufigsten Worten insofern, dass durch sie wenig Erkenntnisgewinn zu erwarten scheint. Sie enthalten Elemente, die unter den häufigsten Named Entitys zu erwarten sind (z.B. „New York“ oder „Vereinigten Staaten“) und häufige Phrasen („v. Chr.“ oder „folgender Personen“). Einzig „Zweiten Weltkrieg“ sagt direkt etwas über einen großen Themenkomplex innerhalb der Wikipedia aus.[109]

Auch die nach anderen Relevanzmaßen ausgewählten N-Gramme sind wenig hilfreich: Dort enthalten sind Worte, die in einer besonders hohen Wahrscheinlichkeit zusammen auftreten. Dadurch sind viele fremdsprachige („Secretariado Técnico Administração“) oder sehr fachspezifische Begriffe („Affodillgewächse Asphodeloideae“) in den Listen enthalten. Diese haben wenig Aussagekraft über den Gesamtbestand und helfen zwar beim Finden von Kuriositäten, nicht aber für die Strukturierung der Sammlung oder beim Identifizieren typischer Inhalte.

Named Entitys

Die Wikipedia ist reich an Personen, Orten und Institutionen. Die NER kann dabei helfen, zu erkennen, auf welchen Personen und Orten der Fokus der Sammlung liegt. Dabei ist die eingangs erwähnte Einschränkung zu beachten: Mehrere beieinanderstehende Entitäten werden zu einer zusammengefasst, über Satzzeichen voneinander getrennte Bestandteile des gleichen Namens werden als mehrere Entitäten gezählt.

Die häufigsten Orte in der Sammlung sind in absteigender Häufigkeit Berlin, Deutschland, München, Wien, die USA, Stuttgart, Frankreich, Hamburg, Österreich und Paris (für einen Auszug aus den erkannten Entitäten inklusive der zehn häufigsten Orte siehe Abbildung 7). Zunächst ist festzustellen, dass es sich bei all diesen Eigennamen tatsächlich um Orte handelt; der (in der nach Häufigkeit sortierten

[109] Von einem Bigramm wie “v. Chr.“ können natürlich Informationen abgeleitet werden: Es kommen in den Texten offenbar viele Daten vor, es werden Themen aus der Antike behandelt, es wird eine westliche Perspektive eingenommen beziehungsweise der entsprechende Kalender genutzt. Das ist aber erstens schon aus den häufigsten Wörtern ersichtlich und zweitens bekannt, wenn der Nutzer ein Minimum an Vorkenntnissen über die Sammlung hat. Insofern wird es als wenig hilfreich erachtet.

Aufzählung) erste Treffer, der keinen Ort bezeichnet, ist „Bronze“ auf Platz 95. Die Rangfolge leidet unter dem oben beschriebenen Problem: „Deutschland“ zählt 8.055 Erwähnungen. Werden auch die Genitivformen und Varianten mit Satzzeichen am Wortende dazugezählt, erhöht sich die Zahl auf 9.946. Für eine genaue Zählung oder Rangfolge taugt die NER also wie erwartet nicht, ebenso kann sie logisch zusammenhängende Entitäten wie „Deutschland“ und „Bundesrepublik Deutschland“ oder „USA“ und „Vereinigte Staaten“ nicht zusammenführen. Der Eindruck, dass die deutsche Wikipedia sich vor allem mit Orten in Deutschland (Berlin, Deutschland, München, Stuttgart, Frankfurt, Hamburg) oder seinen Nachbarländern (Wien, Österreich, Paris) und wichtigen politischen Partnern (USA) befasst, scheint aber plausibel.

Ähnlich verhalten sich die Personen. Die zehn häufigsten erkannten Entitäten heißen Schloss, Gott, Maria, Alexander, Friedrich, Gottes, Herzog, Heinrich, Fischer und Wilhelm. Hier wird zum einen das Genitiv-Problem ersichtlich („Gott“ und „Gottes“); zum anderen ist ein offensichtlich falsch klassifiziertes Wort („Schloss“) die häufigste Entität. Auch die Probleme mit der fehlenden Zusammenführung oder Disambiguierung von Namen wiederholen sich: „Friedrich“, „Friedrich I.“, „Barbarossa“, „Friedrich Barbarossa“ und „Friedrich I. Barbarossa“ (sowie Genitivformen all dieser Namen) werden getrennt voneinander behandelt, obwohl sie die gleiche Person behandeln (können); zugleich können „Friedrich“ und „Friedrich I.“ viele andere Personen als den römisch-deutschen Kaiser bezeichnen, weshalb eine automatische Zusammenführung nicht ohne weiteres möglich wäre, selbst wenn der Zusammenhang bekannt wäre.[110] Von diesen Problemen abgesehen ist die häufige Nennung religiöser oder historischer Figuren (Gott, Maria, Alexander, Friedrich, Herzog, Heinrich, Wilhelm) für die Wikipedia typisch.

Unter den zehn häufigsten Organisationen finden sich politische Parteien und Gruppierungen (SPD, CDU, Nationalsozialisten, Grüne und FDP) und eine große Firma (BMW). Wichtigste „Organisation“ ist die ISBN – die nicht tatsächlich eine

[110] Der Versuch der Identifikation der mit einer Bezeichnung verknüpften Entität wird als Named Entity Disambiguation (NED) bezeichnet. Dafür wird zum Beispiel eine Wissensbasis wie Wikipedia verwendet, anhand derer die Entitäten mit Eigenschaften verknüpft werden können, um die Zuordnung zu erleichtern. Es wurde nicht versucht, NED zu implementieren, da zu erwarten ist, dass der Großteil der in den archivischen Quellen vorkommenden Personen nicht in derartigen Wissensbasen behandelt wird.

Organisation bezeichnet, aber zumindest in einem Zusammenhang mit Institutionen steht. Außerdem wurde Bayern in 451 Fällen als Organisation und nicht als Ort gezählt.[111] Die Kategorie „Organisation“ enthält einige falsch klassifizierte Abkürzungen oder Eigennamen (zum Beispiel DVD, WM oder Windows).

Als „Weitere“ werden vor allem Adjektive zu Nationalitäten (u.a. „deutscher“, „US-amerikanischer“, „französischen“) geführt, außerdem zum Beispiel die Weltkriege („Zweiten Weltkrieg“, „Ersten Weltkrieg“) oder Dekaden („1990er“). Diese Kategorie ist vermutlich die am wenigsten hilfreiche.

Auf Dokumentebene ist die NER auch dafür geeignet, sich einen Überblick über alle vorkommenden Entitäten zu verschaffen – auf Sammlungsebene wären das in diesem Bestand zu viele, als dass der Versuch sinnvoll wäre.[112]

Doc2Vec-Cluster

Mit Doc2Vec konnte bei der Datengrundlage von 50.000 Wikipedia-Artikeln keine sinnvolle Verteilung festgestellt werden.

Topic Models

Es wurden 20 Topics gebildet (siehe Anhang C). Sie entsprechen Mustern von Dokument-Clustern, die in Vorfeldtests in ähnlicher Form ermittelt wurden[113], beispielsweise mit einem Sport-Topic (unter anderem beschrieben durch „*platz saison gewann*“), einem Personen-Topic („*personen name fußballspieler*“), einem Kunst-Topic („*film band album*“), Verkehrs- und Ortstopics („*km liegt county*“, „*gemeinde landkreis st*“, „*stadt jahrhundert kirche*“) und einem Geschichts-Topic („*ii friedrich könig*“). Mehrere länderspezifischen Topics legen nahe, dass die deutschsprachige Wikipedia ihren Schwerpunkt auf Inhalten zu Deutschland („*berlin deutschen deutsche*“), dem angelsächsischen Raum („*and new us*“), Frankreich und Italien

[111] Möglicherweise hängt dies mit Erwähnungen des Landes Bayern beziehungsweise seiner Regierung als Akteur zusammen. Die Stellen, an denen Bayern als Organisation statt als Ort klassifiziert wurde, lassen sich nicht nachträglich nachvollziehen, weshalb hier nur spekuliert werden kann. Der Versuch, bei einem weiteren Durchlauf die Klassifikation nachzuvollziehen, wurde nicht unternommen.

[112] In vorliegendem Bestand wurden insgesamt 689.553 unterschiedliche Entitäten erkannt. Davon werden nur 10.000 ausgegeben, da davon ausgegangen wird, dass die sehr selten vorkommenden Entitäten nicht viel über die Sammlung aussagen und das Laden zehntausender bis hunderttausender Entitäten die Performance stark beeinträchtigt. 551.031 der 689.553 Entitäten kommen nur ein Mal vor, was die These stützt, dass die häufigsten zehntausend Elemente für einen Eindruck der Sammlung ausreichen; um darin enthalten zu sein, muss eine Entität in dieser Sammlung mindestens 12 Mal vorkommen.

[113] Siehe die in Anhang B referenzierte Arbeit, insbesondere die beigefügten Ergebnisse.

(„*französischen jean del*“) sowie Österreich („*wien min österreich*“) hat. Auch zeigen die Topics auf, dass die Wikipedia häufig mit Jahreszahlen verknüpfte Ereignisse („*2004 2000 2006*“, „*1945 1933 1939*“) behandelt und Belege in den Artikeln eine wichtige Rolle spielen (Topics 6, 7, 8 und 18, zum Beispiel „*isbn verlag universität*“ oder „*abgerufen einzelnachweise mai*“).

Ein Blick auf die für die Topics signifikanten Dokumente offenbart allerdings, dass diese sehr durchmischt sind und der Beschreibung des Topics nur selten entsprechen. Das Topic Modelling ist also eine gute Orientierung für die Struktur der Sammlung, die Zuordnung der Dokumente aber ist bei diesem Testfall wenig hilfreich.

Tf-idf-Cluster

Bei der Bildung von 20 Tf-idf-Clustern wurden unter anderem Cluster zu Kunst und Kultur, Personen, Orten, Sport und Cluster voller Listen gebildet (siehe Anhang D). Dies entspricht ungefähr den Themen, die in den bereits referenzierten Vorfeldtests ermittelt wurden. Bis auf die beiden Cluster mit Listen orientieren sie sich rein an inhaltlichen Gesichtspunkten – was insofern plausibel ist, dass die Wikipedia-Artikel strukturell sehr homogen sind und die beim Topic Modelling gefundenen Merkmale wie Belege oder Jahreszahlen in allen Artikeln vorkommen (und somit für keinen spezifisch sind). Dem Großteil der Cluster (14 von 20 Cluster, circa 22.000 von 50.000 Dokumenten umfassend) konnten sinnvolle Überschriften gegeben werden; das bedeutet aber auch, das knapp die Hälfte der Dokumente nicht in ein Cluster einsortiert werden konnte. Für diese sollte es in einer ausgereiften Anwendung eine Backup-Strategie geben.[114] Je mehr Dokumente die Cluster umfassen, desto geringer ist die Wahrscheinlichkeit, dass sie thematisch kohärent sind (allein die Cluster 1 und 3 umfassen gemeinsam knapp 17.000 Dokumente).

Ähnliche Dokumente

Der Verweis auf ähnliche Dokumente auf Basis von Doc2Vec konnte nicht systematisch, sondern nur stichprobenartig getestet werden.[115] Unter den 50 dem

[114] Beispielsweise könnten als sinnvoll erachtete Cluster gespeichert und mit einem Namen versehen werden. Die darin enthaltenen Dokumente könnten von der weiteren Verarbeitung ausgeschlossen und die übrigen Dokumente neu geclustert werden. Die so entstandenen Cluster könnten entweder mit bestehenden verschmolzen oder als eigene Cluster angelegt werden.

[115] Alle im Folgenden referenzierten Texte können auch in der Wikipedia abgerufen werden; im CollectionExplorer wird die Fassung vom 18.06.2014 genutzt.

Artikel zum Film „…men Olsenbanden var ikke død!“[116] ähnlichsten Dokumenten finden sich 48 weitere Filme und zwei Romane (für einen Auszug aus der Liste siehe Abbildung 12). Mit 1920 Wörtern ist dies der längste Artikel in der Stichprobe; die Ermittlung ähnlicher Dokumente scheint für ihn gut zu funktionieren.[117]

Ebenfalls gute Ergebnisse liefert die Ähnlichkeit beim Artikel „Alter Bahnhof (Heilbronn)“[118] (736 Wörter). Von den 50 ähnlichsten Artikeln haben 23 direkten Bezug zum Bahnverkehr, bei weiteren sieben geht es um Straßen oder Wege, es besteht also ein Bezug zum Themenkomplex „Verkehr“. Die übrigen 20 Artikel behandeln Orte (im weiteren Sinne, von Ortschaften bis hin zu Seen oder römischen Kastellen), die teils auch unter Bezugnahme auf die örtliche Eisenbahn beschrieben werden.[119]

Der Artikel zum Karolinger-König Karl das Kind verlinkt hauptsächlich (spät-)mittelalterliche Herrscherlisten[120], was insofern plausibel ist, dass der sehr kurze Artikel (255 Wörter) sieben verschiedene Herrschernamen und drei weitere Namensformen enthält.

Noch kürzer (165 Wörter) ist der Artikel zur islamischen Persönlichkeit Alī ibn Husain Zain al-ʿĀbidīn.[121] Unter den verlinkten Artikeln befinden sich 20 mit Bezug zu entweder religiösen oder historischen Themen und Persönlichkeiten, darunter ein weiterer Artikel zu einem mittelalterlichen schiitischen Gelehrten[122]. Acht weitere Artikel behandeln ebenfalls eine spezifische Person und sind somit in dieser Eigenschaft dem Ursprungsartikel ähnlich. Bei den übrigen 21 Artikeln kann keine Ähnlichkeit ausgemacht werden. Es handelt sich um 17 Orte und vier Pflanzen.

Der Artikel zu den Österreichische[n] Alpine[n] Skimeisterschaften 1963[123] (556 Wörter) führt als ähnliche Dokumente vorrangig weitere Artikel zu Skiwettbewerben (sieben) oder anderen Sport-Themen (acht) und Listen (14) auf. Letzteres erklärt sich vermutlich daraus, dass der Artikel aus listenähnlichen Daten besteht: Er führt jeweils

---

[116] Abrufbar unter [gestrichen]:8000/explorer/collections/11/docs/70791/.
[117] Nachfolgend wird auf die Länge der Artikel Bezug genommen, da davon ausgegangen wird, dass mehr Vergleichstext zu besseren Ergebnissen führt.
[118] Abrufbar unter [gestrichen]:8000/explorer/collections/11/docs/23504/.
[119] Zum Beispiel „Oertzenhof (Woldegk)“, abrufbar unter ([gestrichen]:8000/explorer/collections/11/docs/54991/).
[120] Einschließlich Listen geistlicher Würdenträger.
[121] Abrufbar unter [gestrichen]:8000/explorer/collections/11/docs/70772/.
[122] Al-Kulainī, abrufbar unter [gestrichen]:8000/explorer/collections/11/docs/22854/.
[123] Abrufbar unter [gestrichen]:8000/explorer/collections/11/docs/70636/.

die zehn Bestplatzierten pro Disziplin und ihre Zeiten auf. Sechs verlinkte Artikel behandeln Personen, 15 sonstige Themen. Unter den auf den ersten Blick nicht mit den Skimeisterschaften zusammenhängenden Artikeln finden sich auch solche, die zwar streng genommen keine Liste sind, aber einen aufzählungsartigen Charakter haben.[124]

Die ähnlichen Dokumente sind zusammenfassend nicht immer passgenau, aber bei ausreichender Trainingsdatenmenge für das Sprachmodell werden viele miteinander in einem inhaltlichen oder strukturellen Zusammenhang stehende Dokumente gefunden – insbesondere, wenn das Ausgangsdokument lang genug ist.

Versionen und Duplikate

Auf das Testen von Versionen und Duplikaten wurde verzichtet, da alle Artikel im Bestand einmalig sein sollten.

5.2.2 Dateiablagen des Hessischen Hauptstaatsarchivs Wiesbaden

Um den CollectionExplorer mit authentischen Daten zu testen, fand eine Kooperation mit dem Hessischen Hauptstaatsarchiv Wiesbaden statt. Dem Staatsarchiv angebotene Dateiablagen wurden mit Hilfe des CollectionExplorers verarbeitet, um zu ermitteln, ob und unter welchen Bedingungen die genutzten Techniken hilfreich für die Bewertung von Dateisammlungen sind. Die gesamte Verarbeitung fand im Hauptstaatsarchiv auf Servern des Landes Hessen statt.

Dieses Kapitel bildet das Herz vorliegender Arbeit. Da die Bestände aber strengen Datenschutzregelungen unterliegen, können sie im Folgenden nicht genau benannt werden; ihre Inhalte werden nur insofern zitiert, wie sie komplett generisch sind und keinen Aufschluss auf ihren Urheber geben. Ansonsten wird mit Zusammenfassungen gearbeitet. Das ist insofern ungünstig, dass (ohne einen Besuch im Hauptstaatsarchiv) wenig Evidenz für die Ergebnisse geliefert werden kann. Allerdings hat das vorherige Kapitel bereits nachvollziehbar etabliert, ob und wie die Verfahren grundsätzlich funktionieren und welche Ergebnisse von ihnen zu erwarten sind. Dieses Kapitel zielt also nicht auf ab, die Funktionsfähigkeit des CollectionExplorers zu überprüfen. Es soll

[124] Zum Beispiel der Artikel „Wahlkreis Erfurt I.“, der Stadtteile und Wahlergebnisse aufzählt ([gestrichen]:8000/explorer/collections/11/docs/68131/), und „Verwaltungsgemeinschaft Dasing“, welcher die zur Verwaltungsgemeinschaft gehörigen Gemeinden mit Fläche und Einwohnerzahl aufführt ([gestrichen]:8000/explorer/collections/11/docs/67479/).

stattdessen den über den CollectionExplorer erzielbaren Informationsgewinn für Archivbestände evaluieren, unter der Annahme, dass die eingesetzten Methoden wie im vorangegangenen Abschnitt funktionieren.

5.2.2.1 Datenbestand einer Schule (A)

Bei dem Bestand handelt es sich um die Abgabe einer hessischen Schule. Enthalten ist die Dateiablage, mit der das Personal im Schulalltag gearbeitet hat, es ist also eine Ordnerstruktur vorhanden und es sind Vermutungen über den Inhalt der Ablage möglich. Die Bearbeitung eines so strukturierten und von der Größe her überschaubaren (siehe unten) Bestands ist auch ohne eine Anwendung wie den CollectionExplorer ein lösbares Problem. Er taugt darum vor allem zur Überprüfung der Qualität der Ergebnisse. Die Sammlung setzt sich folgendermaßen zusammen:

- Gesamtzahl Dokumente: 13.524
- Textdokumente (DOC, DOCX, TXT): 9.028
- Publikationen (PDF/EPUB): 2598
- Webdokumente (HTML, HTM): 148
- Tabellen (XLS, XLSX): 1471
- Durchschnittliche Dokumentlänge: 26.539,5 Zeichen inklusive Leerzeichen

Worthäufigkeiten

Die Worthäufigkeiten enthalten ähnlich dem Wikipedia-Bestand keine überraschenden oder hilfreichen Informationen. Sie bestätigen naheliegende Annahmen über die Inhalte der Dokumente, bringen aber keine neuen Einsichten hervor. Enthalten ist zum Beispiel der Ort, an dem die Schule sich befindet, sowie dessen Postleitzahl, außerdem typische Worte aus dem Schulalltag (zum Beispiel vermutlich Noten repräsentierende Zahlenwerte).

N-Gramme

Die häufigsten N-Gramme zeigen hauptsächlich Floskeln aus an Schulen häufigen Schreiben, wie zum Beispiel „Übereinstimmung [des] Duplikates“ (Beglaubigung), „unentschuldigt 0“ (Fehlstunden auf Zeugnissen) oder Name und Adresse der Schule.

Die nach anderen Relevanzmaßen ausgewählten N-Gramme taugen ebenfalls nur bedingt zur Analyse des Gesamt-Korpus‘. Sie enthalten Begriffe wie (das in der Schulmensa angebotene) „Chili con Carne“ oder den Namen eines in die Schule eingeladenen Musik-Kabaretts. Solche Kuriositäten können in Sonderfällen, bei

passenden Forschungsfragen, interessant sein. Für einen Überblick über den Bestand aber sind sie zu uncharakteristisch.

Named Entitys

Die Erkennung von Named Entitys funktioniert wie erwartet. Häufige Entitäten beinhalten unter anderem die Schule selbst, den Ort, in dem sie sich befindet, und Lehrpersonen. Die Zuordnung zu den Kategorien ist nicht immer korrekt, was die Zählung beeinträchtigt – denn so findet sich die gleiche Entität manchmal in zwei verschiedenen Kategorien, jeweils mit einer eigenen Vorkommenshäufigkeit. Da die Anzahl der Falschzuordnungen aber nicht sehr hoch ist, bleiben die Tendenzen gleich. Und für mehr als Tendenzen kann die Funktion aufgrund der bekannten Probleme mit zum Beispiel Deklinationen, Pluralformen oder aneinandergereihten Namen nicht genutzt werden. Die tatsächliche Häufigkeit ermittelt die Suche besser als die NER, insbesondere, da sie auch Teile von Wörtern oder Wortfolgen findet. Um aber die wichtigsten Akteure und Orte im Bestand zu identifizieren, sind die Ergebnisse hilfreich.

Doc2Vec-Cluster

Beim Test von 12 Doc2Vec-Clustern konnten keine sinnvollen Cluster in ausreichender Zahl festgestellt werden. Bei nur 13.500 Dokumenten von sehr variabler Länge liegt der Schluss nahe, dass die Menge an Trainingsdaten nicht groß genug war, um Dokumentvektoren von ausreichender Qualität zu erzeugen.

Topic Models

Aus den Dokumenten wurden zwölf Topics gebildet. Sie gruppieren zum Beispiel Stundenpläne, Aufsichtsangelegenheiten, Budgetangelegenheiten, Unterlagen zu Zeugnissen oder Unterrichtsplanung (siehe Anhang E). Mehrere der Topics wirken sinnvoll. Häufig könnten sie noch weiter untergliedert werden, die Sichtung erlaubt aber bereits das Bilden vorläufiger Kategorien. Mit Hilfe der Topics konnten typische Dokumente und Inhalte identifiziert werden. Topic Modelling wäre hier, wenn der Bestand nicht von vorneherein sehr strukturiert gewesen wäre, also ein hilfreiches Mittel gewesen, um sich einen Eindruck von ihm zu verschaffen. Dank der vorhandenen Gliederung kann konstatiert werden, dass die Topics viele Kategorien aus der Dateiablage abdecken, wenn auch nicht im gleichen Strukturierungsgrad.

Tf-idf-Cluster

Das Ergebnis der Bildung von zwölf Tf-idf-Clustern ist qualitativ gemischt. Es wurden einige sinnvolle, aber auch mehrere nutzlose Cluster gebildet. Erstere beinhalteten zum Beispiel Stunden- und Vertretungspläne, Unterrichts- oder Abituraufgaben, Budgetangelegenheiten oder Krankenblätter (siehe Anhang F). Clustergrößen von circa 500 Dokumenten erscheinen vielversprechend, mehr selten zielführend. Die Tf-idf-Cluster ermöglichen die Identifikation gleichartig strukturierter Quellen (siehe Cluster 7: *Budget* oder Cluster 10: *Krankenblätter*), aber auch Unterrichtsinhalte werden überraschend zuverlässig gemeinsam eingruppiert (insbesondere angesichts der Clustergröße; siehe Cluster 11: *Unterrichtsinhalte*). Tf-idf-Cluster erscheinen insofern als geeignetes Mittel, um dem Bestand explorativ zu begegnen.

Ähnliche Dokumente

Dokumente wurden stichprobenartig darauf überprüft, welche ähnlichen Dokumente von Doc2Vec gefunden wurden. Die Ergebnisse waren nicht verlässlich, es wurden also nicht immer passende Dokumente vorgeschlagen, aber die Vorschläge mit hoher Ähnlichkeit (>0.8) standen häufig in einem Zusammenhang mit dem Ursprungsdokument. So wurden beispielsweise verschiedene Dokumente der gleichen Referendarin identifiziert, in denen sie unter anderem Unterricht vorbereitet und Literatur bestellt hat.

Duplikate und Versionen

Die Versionserkennung ist zunächst am verfügbaren Arbeitsspeicher gescheitert. Nach Restrukturierung der Funktion ist sie erfolgreich durchgelaufen, die Ergebnisse der Versionserkennung wurden (nach erfolgreichen Tests zum Beispiel in Bestand B) aber nicht mehr überprüft.

5.2.2.2 Beschlagnahmte Festplatte eines Rechtsextremisten (B)

Der Bestand B stammt von einer durch die Staatsanwaltschaft beschlagnahmten Festplatte eines Rechtsextremisten. Er ist strukturiert, aber erst auf den unteren Order-Ebenen. Ein Überblick über die Dateien nur mit Nutzung des Dateisystems wäre schwer zu erlangen. Er ist folgendermaßen zusammengesetzt:

- Gesamtzahl Dokumente: 6.133, davon 4.699 Unikate[125]

[125] Die Bezeichnung als Unikate ist nicht ganz korrekt. In einer fiktiven Sammlung mit 100 Dokumenten, von denen 90 als Unikate beschrieben werden, existieren tatsächlich 80 einzigartige

- Textdokumente (DOC, DOCX, TXT): 2.875
- Publikationen (PDF/EPUB): 26
- Webdokumente (HTML, HTM): 2.840
- Durchschnittliche Dokumentlänge: 17.661,7 Zeichen inkl. Leerzeichen

Worthäufigkeiten

Die häufigsten Worte spiegeln den Inhalt der Sammlung thematisch gut wieder. Es finden sich viele Begriffe im Zusammenhang mit Deutschland, außerdem beispielsweise „Hitler“, „Juden“, „Krieg“, „USA“, „KDS“ oder „Politik“. Auch die Namen prominenter Neonazis sind in der Liste der häufigsten Wörter enthalten. Auf die Struktur der Sammlung, zum Beispiel die viele enthaltene Korrespondenz, die abgespeicherten Zeitungsartikel oder Leserbriefe, lassen die Worthäufigkeiten nicht schließen.

N-Gramme

Die häufigsten N-Gramme haben einen ähnlichen Informationsgehalt wie die häufigsten Worte: Mit Begriffspaaren wie „deutsches Volk“, „Adolf Hitler“, „Vereinigte Staaten“, „Holo-Industrie [||] Multikultur“ (als Verschlagwortung) oder den Namen prominenter Neonazis erfüllen sie die Erwartungen an den Sammlungsinhalt.

Die mit anderen Relevanzmaßen gewichteten N-Gramme geben kleine Hinweise auf die Struktur der Sammlung, sie beinhalten beispielsweise einige Postleitzahl-Stadt-Kombinationen aus Anschriften. Aber auch uninteressante N-Gramme (z.B. „Walt Disney“, „Greatest Hits“) oder häufige Tippfehler und ungewöhnliche Schreibweisen („Burscen“, „“hot unz getrofen“, „Kameradscaft“) werden mit der voreingestellten Mindest-Häufigkeit pro N-Gramm (100) durch sie erfasst.

Einzig die Trigramme nach Likelihood-Ratio beinhalten unerwartete Elemente. In den Dokumenten werden vom Stoppwort-Filter nicht erkannte Anführungszeichen verwendet. Das mündet in Trigrammen wie „`` 6-Millionen '“ , „`` Propagandadelikte '“ , „`` Ungläubigen '“ , „`` Befreiung '“ , „`` Holocaust-Leugnung '“ , „`` Protokolle '“ , „`` Holocaust-Leugner '“ oder „`` Volksverhetzung '“.' Es entsteht also eine Übersicht über durch den Autor in Frage gestellte Themenkomplexe, die sich nahtlos an den durch

Dokumente. Dazu kommen 20 mehrfach vorhandene Texte, die je einmal als Unikat und in allen weiteren Fassungen als Duplikat gezählt werden.

die häufigsten Wörter suggerierten Inhalt anschließt. (Und insofern, auch wenn sie überraschend ist, inhaltlich wenig Neues beiträgt.)

Named Entitys

Es wurden viele Named Entitys mit Bezug zu „Deutschland“ gefunden, mehrere rechtsextreme Organisationen („KDS“, „NPD“), die Bundesrepublik Deutschland als „BRD“, außerdem die Namen prominenter (Neo-)Nazis und häufiger Korrespondenten. Die sich deutlich unterscheidenden Häufigkeiten erlauben dabei die Identifizierung von im Bestand wichtigen Personen und Organisationen.

Doc2Vec-Cluster

Mit Doc2Vec wurden keine sinnvollen Ergebnisse erzielt.

Topic Models

Beim Topic Modelling mit zehn Topics wurden Dokument-Gruppen gebildet, die beispielsweise Leserbriefe und Zeitungsartikel, Briefe, technische Dokumente oder antisemitische Dokumente enthielten. Ein Abgleich mit vorhandenen Strukturen ist hier nicht möglich, die Einteilung wirkt aber sinnvoll. Manche Topics (wie 1: *Religion, Nationalsozialismus, deutsches Volk*, 4: *Zweiter Weltkrieg, Deutschland, Amerika, Krieg, historische Themen* oder 9: *Nationalsozialistische Ideologie, „Kampfbund deutscher Sozialisten“*) waren vergleichsweise unspezifisch, andere dagegen konnten entweder thematisch (5: *Holocaust, Juden, Auschwitz*, 7: *Juden, Banken, Holocaust, Israel*, 8: *Erinnerungen an einen verstorbenen Neonazi, Korrespondenz*) oder strukturell (2: *Leserbriefe, Zeitungsartikel*, 3: *Briefe*, 6: *Technische Dokumente*, 10: *Inhaltsangaben von Fernsehsendungen*) klar zugeordnet werden.

Tf-idf-Cluster

Die Bildung von zehn Clustern resultierte in ähnlichen Ergebnissen wie das Topic Modelling (siehe Anhang H). Es gab unter anderem Cluster mit Zeitungsartikeln und Leserbriefen, Korrespondenz oder antisemitischen Texten. Die Dokumente werden hier etwas stärker nach ihrer Struktur gruppiert (1: *Zeitungsartikel, Leserbriefe*, 2: *Korrespondenz*, 4: *Adressen/Verteiler*, 5: *Zeitungsartikel, Leserbriefe*, 8: *Korrespondenz*, 10: *Leserbriefe*) als nach Inhalten (3: *Holocaust, Juden*, 7: *Juden, Israel, Holocaust*, 9: *Deutschland, historische Themen, Amerika, Krieg, Globalisierung*). Zwei der Cluster sind inhaltlich leicht (9) bis komplett (6: -) vermischt.

Ähnliche Dokumente

Die Verknüpfung ähnlicher Dokumente produziert stellenweise sinnvolle Ergebnisse. So werden beispielsweise (einander sprachlich oder strukturell ähnliche) Gedichte oder (einander inhaltlich ähnelnde) antisemitische Dokumente als ähnlich erkannt.

Duplikate und Versionen

Die Identifikation von Versionen wirkt vielversprechend: Bei Tests wurden beispielsweise sieben Fassungen des gleichen Berichts korrekt als Versionen eines Dokuments identifiziert. Bei nur aus dem Briefkopf des Verfassers und einer Empfänger-Adresse bestehenden Dokumenten wurden Versionskandidaten fälschlicherweise vorgeschlagen – allerdings waren die Ähnlichkeiten nur knapp über der Schwelle von 0.5, ab der Dokumente als Versionen voneinander vorgeschlagen werden, und die sehr kurzen Dokumente zu großen Teilen (mindestens dem Briefkopf des Absenders, gelegentlich auch Teile der Empfängeradresse wie ein Name oder eine Stadt und Postleitzahl) deckungsgleich.

5.2.2.3 Beschlagnahmte Festplatte eines Rechtsextremisten (C)

Die Dateisammlung stammt von einer durch die Staatsanwaltschaft beschlagnahmten Festplatte einer Privatperson aus dem rechtsextremen Spektrum. Die Dateien darauf sind zum größten Teil in Archiven ohne aussagekräftige Titel zusammengefasst. Eine Struktur findet sich nur sehr tief im Dateibaum.

- Gesamtzahl Dokumente: 635, davon 233 Unikate
- Textdokumente (DOC, DOCX, TXT): 541
- Publikationen (PDF/EPUB): 73
- Tabellen (XLS, XLSX): 13
- Durchschnittliche Dokumentlänge: 3876.8 Zeichen inklusive Leerzeichen

Der Bestand (an Textdokumenten; die Ablage enthält hauptsächlich audiovisuelles Material) ist also sehr klein, außerdem ist der Großteil der Dokumente auf Englisch. Eine Konfiguration der Anwendung zum Parsen englischer Dokumente wäre möglich gewesen, wurde aber nicht vorgenommen.[126]

[126] Die verfügbare Zeit für Forschungsaufenthalte im Hauptstaatsarchiv war begrenzt, und die Anwendung ist für die Arbeit mit deutschen Texten vorgesehen, die den allergrößten Teil der relevanten Daten ausmachen. Automatische Spracherkennung und die Nutzung passender Ressourcen für die Tokenisierung und die Erkennung von Named Entitys wären grundsätzlich möglich, aus den oben genannten Gründen wurde aber auf eine Implementierung verzichtet.

Als einzig nützliches Verfahren stellte sich die Duplikaterkennung heraus: Mit ihr konnte der Bestand auf etwa ein Drittel reduziert werden und schrumpft auf eine Größe, die ein menschlicher Bearbeiter problemlos vollständig durchsehen kann.

Die im CollectionExplorer unter „Statistische Analyse“ zusammengefassten Verfahren scheitern an der Sprache des Bestands. Englische Stoppwörter werden nicht gefiltert und dominieren die Worthäufigkeiten und N-Gramme. Für erfolgreiche NER müssen mit der jeweiligen Sprache trainierte Modelle für Satzerkennung und NER vorliegen, was hier nicht der Fall ist. Topic Modelling, Tf-idf-Cluster und Doc2Vec erfordern mehr Trainingsdaten als hier gegeben und liefern somit ebenfalls keine sinnvollen Ergebnisse.

5.2.2.4 Datenbestand einer Schule (D)

Bei Bestand D handelt es sich wie bei Bestand A um die Dateiablage einer Schule. Dieser Bestand ist aber ungleich größer: Die Gesamt-Datenmenge ist im dreistelligen Gigabyte-Bereich, enthalten sind etwa 700.000 Dateien. Viele davon sind in Archiven verpackt, so dass die Gesamtmenge ohne Hilfsmittel nicht erkennbar ist. Die Dateien sind auf mehr als zehn Ebenen im Dateibaum verschachtelt. Stellenweise existiert eine transparente Ordnung oder Struktur, aber die starke Verschachtelung und die von mehreren Backups an unterschiedlichen Zeitpunkten erzeugten Parallelstrukturen führen trotzdem zu einem enorm unübersichtlichen Bestand. Mit herkömmlichen Methoden ist über ihn kein Überblick zu gewinnen. Er setzt sich zusammen aus:

- Gesamtzahl Dokumente: 150.062
- Textdokumente (DOC, DOCX, TXT): 127.140
- Publikationen (PDF/EPUB): 2.790
- Webdokumente (HTML, HTM): 11.683
- Tabellen (XLS, XLSX): 7.112
- Durchschnittliche Dokumentlänge: 9.782,7 Zeichen inkl. Leerzeichen

Die Größe dieses Bestandes kombiniert mit den begrenzten zeitlichen Ressourcen im Hauptstaatsarchiv Wiesbaden machten bei der Analyse des Bestands D einen Fokus auf ausgewählte Methoden notwendig. Da Doc2Vec zum Training die größten Datenmengen benötigt, sollten vor allem die Qualität des Doc2Vec-Sprachmodells und der Doc2Vec-Cluster untersucht werden. Auf die Untersuchung der schon erfolgreich eingesetzten Tf-idf-Cluster und Topic Models wurde aus Zeitgründen ebenso

verzichtet wie auf die zeitintensive (und ausreichend getestete) Versionserkennung und NER.

Worthäufigkeiten

Die Liste der häufigsten Worte umfasst vermutlich Noten repräsentierende Zahlenwerte („1.0“) und Wörter aus Korrespondenzfloskeln sowie schultypische Worte und für den pädagogischen Ansatz der Schulform spezifische Begriffe. Wie bei den bisher betrachteten Beständen wäre diese Liste vor allem dann hilfreich, wenn gar keine Informationen über den Bestand vorlägen.

N-Gramme

Bei den N-Grammen wurde aufgrund der Bestandsgröße eine Mindesthäufigkeit von 1000 gesetzt (im Gegensatz zu 100 bei den übrigen Beständen). Wegen der Menge und der Ähnlichkeit der Bigramm- und Trigramm-Ergebnisse bei den anderen Beständen wurden hier nur Bigramme analysiert. Die häufigsten Bigramme enthalten wie bei Bestand A Phrasen aus Korrespondenz sowie Zahlenwerte, bei denen es sich vermutlich um Noten handelt. Aus den Bigrammen „Lingua Latina“ und „kommunikationstechnische Grundbildung” lassen sich an der Schule unterrichtete Fächer oder Unterrichtsziele erahnen; diese Informationen ließen sich aber auch ohne den CollectionExplorer gewinnen, da der Name der Schule bekannt ist. Ansonsten sind einige Bigramme aus technischen Dokumenten vorhanden, die zwar auf (im Bestand tatsächlich häufig vorkommende) Anleitungen und technische Dokumentation schließen lassen. Es handelt sich dabei aber nicht um für den Bestand wichtige oder interessante Dokumente, da sie mit dem eigentlichen Schulbetrieb nichts zu tun haben.

Clustering-Verfahren

Vorrangiges Ziel der Verarbeitung von Bestand D war ein Test der Doc2Vec-Cluster. Es wurden verschiedene Optionen von 30 bis 150 Clustern getestet. Bei den Tests hat sich schnell eine Schwäche der Clustering-Methoden bei Sammlungen in der Größe des Bestands D offenbart: Alle auf die Cluster verteilten Dokumente müssen zumindest grob von einem Menschen durchgesehen werden. Bei den anderen Beständen stellte dies kein Problem dar; circa zehn Cluster mit wenigen hundert bis tausend Dokumenten zu überfliegen ist leistbar. Hier hingegen mussten dutzende Cluster mit teils einigen tausend Dokumenten angesehen und auf ein gemeinsames Oberthema überprüft werden. Dies stellte sich als nicht praktikabel heraus.

Alternative Vorgehensweisen

Die Erkenntnis aus dem vorhergegangenen Kapitel führt zu einem Dilemma: Gerade für die Bestände, bei denen ein menschlicher Bearbeiter ob der Mengen und (mangelnden) Struktur vollkommen überfordert ist, sind auch die Methoden des CollectionExplorers nicht mehr hilfreich. Im Folgenden sollen Überlegungen angestellt werden, wie Bestände dieser Größe so verarbeitet werden können, dass sie die Bewertung durch einen Menschen unterstützen. Auf die Umsetzung dieser Methoden wird verzichtet; sie würde über die Grenzen dieser Arbeit hinausgehen.

Ein wichtiger Teil der Antwort auf das Problem könnte in Anpassungen an den Bestand liegen. Kuratierte Stoppwortlisten und anpassbare Mindesthäufigkeiten für N-Gramme könnten die Nützlichkeit der statistischen Auswertungen erhöhen. Außerdem könnte eine oberflächliche Analyse des Bestands Ordnerpfade hervortun, die definitiv uninteressant sind und von der weiteren Verarbeitung ausgeschlossen werden können.[127] Unter der Voraussetzung, dass eine Durchlaufzeit von Tagen bis Wochen in Kauf genommen wird, könnten Duplikate aus dem Bestand gefiltert und von der weiteren Verarbeitung ausgeschlossen werden.

Sofern die so erreichten Reduktionen nicht ausreichten, um den Bestand auf eine handhabbare Größe herunterzubrechen, wäre die Einteilung in Teilsammlungen denkbar. Eine oberflächliche Analyse zeigt, dass es sich an verschiedenen Stellen wiederholende Ordner-(Teil-)Pfade gibt, die vermutlich durch Backups entstanden sind und zusammengeführt werden könnten. So könnten beispielsweise Untersammlungen mit Protokollen, Personalsachen oder Korrespondenz gebildet und jeweils für sich stehend analysiert werden.

Falls all diese Ansätze nicht erfolgreich sind, gäbe es noch eine weitere Option: Die Nutzung eines überwachten Verfahrens wie Klassifikation. Dies hätte den Nachteil, dass ein Mitarbeiter Zeit darin investieren müsste, sich mit dem Bestand vertraut zu machen und eine Teilmenge von ihm zu annotieren. Dafür wären die Ergebnisse besser zu überprüfen und – entsprechend gute Scores vorausgesetzt (siehe Kapitel 5.3.1) – verlässlicher als die unüberwachten Verfahren. Der Arbeitsaufwand für eine

[127] Bei Bestand D wären dies beispielsweise größere Mengen technischer Dokumente, wie zum Beispiel Softwarelizenzen oder Drucker-Bedienungsanleitungen, gewesen.

Teilannotation wäre zudem noch immer deutlich geringer als eine umfassende Mikrobewertung des Bestands (also eine Bewertung auf Aktenebene).

5.3 Zusammenfassung der Ergebnisse

Die Ergebnisse der Tests vor allem mit den Datenbeständen des Hauptstaatsarchivs Wiesbaden werden im Folgenden zusammengefasst und diskutiert, inwiefern und unter welchen Bedingungen der CollectionExplorer den Umgang mit unübersichtlichen Dateisammlungen erleichtert.

5.3.1 Rahmenbedingungen der Tests

Die Diskussion der Ergebnisse erfolgt vor dem Hintergrund der verfügbaren Hardware und Datenmengen. Diese beiden Faktoren bereiten insofern Schwierigkeiten, als dass sie einander beeinträchtigen – die eingesetzten Verfahren benötigen möglichst große Datenmengen, um sinnvolle Ergebnisse zu liefern, aber größere Mengen an Daten machen zugleich den Einsatz speicherintensiver Methoden schwierig und vom Bearbeiter mentale Kapazitäten fordernde Verfahren unmöglich. In diesem Spannungsfeld mussten Kompromisse gefunden werden, die meist zulasten der Laufzeit der Algorithmen gingen. Denn eine Verarbeitung notfalls über Nacht laufen zu lassen, war eine (aufgrund der knappen Zeit im Hauptstaatsarchiv unliebsame, aber durchführbare) Möglichkeit. Die Hardware (insbesondere RAM) des Servers so zu verbessern, dass die Datenmengen in der schnellstmöglichen Variante verarbeitet werden konnten, nicht.

Ein weiterer begrenzender Faktor waren die zu Verfügung stehenden Testdaten. Diese haben in ihren Zusammensetzungen glücklicherweise eine Vielzahl an Szenarien abgedeckt: Es standen ein sehr kleiner, ein kleiner, ein mittelgroßer und ein sehr großer Bestand zur Verfügung, einer von ihnen enthielt großteils englische Dokumente. Damit sollten sich die Bestände hinsichtlich der Effektivität der eingesetzten Methoden unterscheiden.

Neben der Zusammensetzung muss auch die Vollständigkeit der Testdaten betrachtet werden. Aus bereits erläuterten Gründen erfolgt eine Beschränkung auf die Formate .csv, .doc, .docx, .epub, .htm, .html, .json, .pdf, .pptx, .tsv, .txt, .xls und .xlsx. Eigentlich relevante Dateien, wie zum Beispiel Fotos von Dokumenten, werden ignoriert, da keine Qualitätssicherung für eine automatische Texterkennung vorgenommen werden kann. Es ist außerdem nicht garantiert, dass alle Dateien dieser

Typen vom CollectionExplorer tatsächlich eingelesen werden. Verschlüsselte Dokumente werden übersprungen, Kodierungsfehler können die Verarbeitung verhindern, die Dateien können korrumpiert oder aus anderen Gründen nicht lesbar sein. Übrig bleibt also nicht der vollständige Textbestand, sondern ein sich aus der Lesbarkeit der Daten ergebender Ausschnitt.[128]

Zuletzt muss eine in der Natur der Problemstellung liegende Schwierigkeit bei der Auswertung berücksichtigt werden. Es handelt sich bei den untersuchten Daten um unübersichtliche Dateisysteme, für die es keine Inhaltsverzeichnisse oder Themenlisten gibt. Die Bestände sind (sofern sie groß genug für eine sinnvolle Verarbeitung sind) zu groß, um sie zu annotieren. Es können also keine Messwerte wie Accuracy oder Precision und Recall erhoben werden – denn dafür wäre eine Ground Truth notwendig, die nicht existiert.[129] Die vollständige Sichtung und anschließende Annotation mehrerer Bestände zu Auswertungszwecken ist im Rahmen vorliegender Arbeit nicht durchführbar. Aus diesem Grund werden die Ergebnisse auf Plausibilität geprüft, nicht auf messbare Werte. Entsprechen sie den in einer Sammlung mit den bekannten Eigenschaften erwarteten Ergebnissen? Gibt es offensichtliche Fehler? Decken sich die durch den CollectionExplorer gewonnenen Erkenntnisse mit den bekannten Hintergrundinformationen zum Bestand?

5.3.2 Anwendbarkeit der computerlinguistischen Methoden auf die Testbestände

Die Vorverarbeitung und die Suche funktionieren wie erwartet problemlos und unterscheiden sich nicht nach Bestandsgröße oder -sprache. Die Anzeige ähnlicher

---

[128] Dies ist auch bei analogem Schriftgut eine bekannte Situation. Auch dessen Ordnungssystem kann verloren gehen, mit ähnlichen Konsequenzen für die Bewertung wie bei Dateisammlungen. Was überliefert wird, wird weiterhin von Katastrophen wie Bränden oder Kriegen beeinflusst – und in weniger dramatischen Fällen nehmen Behördenmitarbeiter mit dem Ausscheiden aus dem Dienst ihre Unterlagen mit und entziehen sie so dem Archiv, oder kassieren den gesamten Altbestand beim Umzug in ein neues Gebäude. Auch die Lagerungsbedingungen und der sich aus ihnen ergebenden Zustand der Archivalien beeinflusst, was erhalten bleibt. Der Einfluss von Ereignissen außerhalb der Kontrolle der Archivare ist also kein neues Problem digitaler Unterlagen.

[129] Wenn ein annotiertes Testkorpus vorliegt, können die darin enthaltenen Dokumente klassifiziert und die Ergebnisse mit der manuellen Zuordnung verglichen werden. Accuracy bezeichnet den Anteil an korrekt klassifizierten Dokumenten. Precision beschreibt den Anteil der richtigerweise einer Kategorie zugeordneten Dokumente an allen zugeordneten Dokumenten. Als Recall bezeichnet man die Vollständigkeit der Ergebnismenge, also wie viele der zu einer Kategorie gehörigen Ergebnisse gefunden wurden. Vgl. Manning et al. 2009, S. 154 f.

Suchbegriffe auf Basis von Word2Vec liefert ab der Größe von Bestand B (circa 6.000 Dokumente) sinnvolle Vorschläge.

	Wikipedia	Schule A	Festplatte B	Festplatte C	Schule D
Sammlungseigenschaften					
Dokumente	49.120	13.524	6.133	635	150.062
Gesamtzahl Zeichen	165.161.088	358.920.198	108.319.206	2.461.768	1.468.011.527
Sprache	Deutsch	Deutsch	Deutsch	Englisch	Deutsch
Strukturierungsgrad	-	hoch	mittel	niedrig	niedrig
Ergebnisse der maschinellen Verfahren					
Vorverarbeitung	erfolgreich	erfolgreich	erfolgreich	fehlerhaft	erfolgreich
Volltextsuche	erfolgreich	erfolgreich	erfolgreich	erfolgreich	erfolgreich
Worthäufigkeiten	Geringer Informations-gehalt	Geringer Informations-gehalt	Geringer Informations-gehalt	nutzlos	Geringer Informations-gehalt
N-Gramme	Geringer Informations-gehalt	Geringer Informations-gehalt	Geringer Informations-gehalt	nutzlos	Geringer Informations-gehalt
Named Entity Recognition	Hoher Informations-gehalt	Hoher Informations-gehalt	Hoher Informations-gehalt	nutzlos	Hoher Informations-gehalt
Doc2Vec-Cluster	Keine Struktur	Keine Struktur	Keine Struktur	Keine Struktur	Nicht anwendbar
Topic Models	Hilfreiche Struktur	Hilfreiche Struktur	Hilfreiche Struktur	Keine Struktur	/
Tf-idf-Cluster	Hilfreiche Struktur	Hilfreiche Struktur	Hilfreiche Struktur	Keine Struktur	/
Dokument-ähnlichkeit	Sinnvolle Vorschläge	Sinnvolle Vorschläge	Sinnvolle Vorschläge	Unsinnige Vorschläge	/
Duplikate und Versionen	erfolgreich	/	erfolgreich	erfolgreich	/

Tabelle 1: Testbestände und Ergebnisdokumentation

Die Methoden zur statistischen Auswertung eines Bestands – Worthäufigkeiten, N-Gramme und NER – funktionieren unabhängig von der Bestandsgröße, sie sind allerdings von der Eingabesprache abhängig. Auch wenn die Sprache stimmt und die

Verarbeitung funktioniert, ist der Erkenntnisgewinn durch Worthäufigkeiten und N-Gramme eingeschränkt. Denn in der Regel liegen genug Informationen über einen Bestand vor, um die wichtigsten Themen im Vorhinein einschätzen zu können – und zu darüber hinausgehenden Erkenntnissen führen die Worthäufigkeiten meist nicht.[130] Der geringe Erkenntnisgewinn durch N-Gramme erklärt sich aus der Art ihrer Ermittlung: Die häufigsten N-Gramme verhalten sich wie die häufigsten Worte. Die nach Signifikanz ermittelten N-Gramme finden Wortfolgen, die sehr viel wahrscheinlicher gemeinsam auftreten als mit anderen Worten. Dieses Vorgehen bevorteilt Fachbegriffe, fremdsprachige und unübliche Worte, die nicht Teil des normalen Sprachgebrauchs sind, sondern (fast) nur gemeinsam auftreten. Diese Methoden haben in allen Testfällen bestehende Annahmen über den Inhalt des Bestands gestützt, ohne signifikante neue Informationen beizutragen. Die Auflistung von Named Entitys dagegen scheint trotz aller bekannter Probleme hilfreich, um die wichtigsten oder prominentesten Akteure im Bestand zu identifizieren.

Verbesserungen bei den statistischen Analysen könnten vor allem durch Anpassung an den Bestand erzielt werden. Die Worthäufigkeiten und N-Gramme könnten mit themenspezifischen Stoppwort-Listen gefiltert werden. Für die Identifizierung interessanter N-Gramme könnte mit der Mindest-Vorkommenshäufigkeit experimentiert werden, um einen Wert zu finden, der Tippfehler und (selten genutzte) fremdsprachige Begriffe ausschließt. Wenn nichtdeutsche Bestände verarbeitet werden, müssen Tokenizer und NER dafür angepasst werden.

Die semantische Auswertung – Doc2Vec-Cluster, Topic Models und Tf-idf-Cluster – benötigt für die Erzeugung sinnvoller Ergebnisse eine Mindestmenge an Dokumenten. Ab der Größe von Bestand B hat dies in den Tests für Topic Modelling und Tf-idf-Cluster gut funktioniert, die Bestände werden in (größtenteils) sinnvolle Gruppen eingeteilt. Dabei fällt auf, dass Topic Models die Dokumente eher nach thematisch-inhaltlichen Gesichtspunkten gruppieren, Tf-idf-Cluster dagegen serielle

[130] Auf Dokumentebene sind sie interessanter – denn während der Inhalt der gesamten Sammlung ungefähr bekannt ist, ist das nicht zwingend für die einzelnen darin enthaltenen Dokumente der Fall. Dort kann eine Word Cloud mit den häufigsten Begriffen schnell Aufschluss über die grobe inhaltliche Einordnung des Texts geben.

Dokumente verlässlich gemeinsam einordnen.[131] Alles in allem sind die Parallelen zwischen den durch die beiden Methoden gebildeten Gruppen aber groß. Doc2Vec-Cluster dagegen waren in Vorfeldtests mit 100.000 Wikipedia-Artikeln nutzbar (siehe Anhang B), erzielten aber mit Archivbeständen keine sinnvollen Ergebnisse.

Das Auffinden ähnlicher Dokumente über Doc2Vec scheint schon bei Dokumentmengen, die für Cluster noch nicht nutzbar sind, nämlich ab der Größe von Bestand B, sinnvolle Vorschläge zu liefern. Die mit Word2Vec umgesetzte Untersuchung des Vokabulars liefert gemischte Ergebnisse: Über alle Bestände hinweg (außer dem allerkleinsten) werden bei der Anzeige ähnlicher Worte zu einem Suchwort plausible Ausgaben erzeugt.[132] Dagegen hat bei keinem Testbestand die Bewegung im Vektorraum funktioniert, die bei Mikolov Berechnungen wie „$König$“ – „$Mann$“ + „$Frau$“ = „$Königin$“ ermöglicht. Auch mit großen Sammlungen und nach mehrfachem Überprüfen der Implementierung konnte hiermit kein sinnvolles Ergebnis erzielt werden.

Die Dokumentähnlichkeit über MinHash identifiziert Versionen in Stichproben zuverlässig. Problematisch beim Auffinden von Versionen und Duplikaten ist die Performance – bei großen Beständen (in den Tests ab Größe von Bestand A) ist die Verarbeitungsdauer sehr hoch, für Bestand D wurde aus diesem Grund komplett auf eine Versionserkennung verzichtet.[133] Grundsätzlich kann die Versionserkennung aber sicherstellen, dass die finale Version eines interessanten Dokuments übernommen wird, indem dem bewertenden Archivar alle Versionen zur Prüfung vorgelegt werden.

Zusammenfassend kann festgestellt werden, dass die Methoden ab der Größe von Bestand B, und sofern die Texte zum allergrößten Teil auf Deutsch verfasst sind,

---

[131] Letzteres ergibt insofern Sinn, dass serielle Dokumente sehr ähnliches Vokabular verwenden und somit durch in großen Teilen identische Vektoren repräsentiert werden.

[132] Bei Bestand B beispielsweise waren die zehn ähnlichsten Worte zu „Juden“ die folgenden: „Christen“, „Nichtjuden“, „jüdischen“, „Nazis“, „Holocaust“, „jüdische“, „Sünder“, „Zionisten“, „Gaskammern“ und „Israelis“. Auch bei den anderen Beständen und mit anderen Suchbegriffen liefert die Funktionalität weitgehend plausible Ergebnisse; Auszüge sind auch auf Abbildung 10 und Abbildung 13 zu sehen. Eine systematische Überprüfung fand allerdings nicht statt, da in Stichproben zu keinem Bestand neue Information zutage gebracht wurden.

[133] Die Versionserkennung ist ein Problem mit quadratischer Komplexität, denn jedes Dokument muss mit jedem anderen verglichen werden. Bereits miteinander verglichene Dokumente können nachgehalten werden, um die Zahl der tatsächlich durchgeführten Vergleiche zu verringern. Die Laufzeit ist trotz allem sehr hoch: Bei Bestand A betrug sie etwa acht Stunden, für Bestand D wird es sich um ein Vielfaches davon handeln.

sinnvoll eingesetzt werden können. Die statistischen Methoden können zwar auch für kleinere Bestände verwendet werden, geben aber insgesamt weniger Aufschluss über den Bestand als die semantischen Methoden. Mit genügend Zeit, um Anpassungen für jeden Bestand vorzunehmen, könnten die Ergebnisse der statistischen Auswertungen vermutlich verbessert und nützlicher gemacht werden. Einige der Methoden, zum Beispiel NER und Versionserkennung, führen zu Problemen mit Laufzeit und Arbeitsspeicherbedarf und würden bei einem über einen Prototyp hinausgehenden Softwareprojekt Optimierung und mächtigere Hardware benötigen.

Bei Beständen in der Größe von Bestand D sind die Clustering-Verfahren, die bei den mittelgroßen Beständen die meisten Erkenntnisse über den Inhalt der Ablage hervorgebracht haben, nicht mehr praktikabel. Um sie zu bearbeiten, müssen mindestens Anpassungen an den Bestand und eine Reduktion der Gesamtmenge vorgenommen werden, vielleicht wäre auch ein Umstieg auf überwachtes maschinelles Lernen notwendig.

6 Potentiale für andere archivische Tätigkeitsfelder

Diese Arbeit konzentriert sich auf das Problem der Bewertung von Dateiablagen, da sie ein grundlegender Schritt im archivischen Arbeitsprozess ist. Für die zuvor vorgestellten Methoden sind aber auch andere Einsatzgebiete denkbar. Sie umzusetzen, würde über den Umfang dieser Arbeit hinausgehen; sie sollen aber kurz umrissen werden, da zum Teil große Überschneidungen mit den vorgestellten Ansätzen existieren.

Die Erschließung von Archivalien erfolgt auf Grundlage von Metadaten und dem Überfliegen des (digitalen oder analogen) Volltexts eines Dokuments. Eine zusammenfassende Ansicht mit den häufigsten Wörtern und den signifikantesten Begriffen nach Tf-idf wäre für die schnelle inhaltliche Einordnung eines Dokuments hilfreich. Named Entitys würden sofort Informationen darüber liefern, ob und welche Personen im Dokument genannt werden und kombiniert mit Suchfunktionen eine schnelle Prüfung von Schutzfristen ermöglichen. Die Verknüpfung von Duplikaten und Versionen kann helfen, Redundanzen zu vermeiden.

Für Archivnutzer wäre eine Anwendung im Stil des CollectionExplorers als Ergänzung zu herkömmlichen Archivdatenbanken denkbar. Zum einen bietet sie mit Volltextsuche, der Suche nach Named Entitys und dem Verlinken ähnlicher

Dokumente komfortable Funktionalitäten, die die Recherche vereinfachen würden. Weiterhin könnte sie als Ersatz dienen, wo noch keine Erschließungsdaten vorhanden sind. Denn es ist nicht davon auszugehen, dass die Archive in Zukunft ihre Erschließungsrückstände[134] vollständig beseitigen werden können. Sobald die Schutzfristen abgelaufen, und wenn mit Sicherheit keine sensiblen Personendaten in den Dokumenten enthalten sind, könnten sie so recherchierbar sein, bevor eine Archivarin eine vollständige, fachgerechte Erschließung vorgenommen hat.

7 Fazit

In vorliegender Arbeit wurde untersucht, ob computerlinguistische Methoden dafür geeignet sind, sich einen Überblick über große, unstrukturierte Dokumentensammlungen zu verschaffen. Diese Fragestellung wurde im Kontext archivischer Bewertung betrachtet, da immer mehr Archive mit solchen Sammlungen konfrontiert werden, ohne bisher eine Methodik für den Umgang mit ihnen entwickelt zu haben. Problematisch sind sie für Archive deshalb, weil sie immer häufiger als anbietungspflichtiges Behördenschriftgut an ein Verwaltungsarchiv gelangen. Sie sind oft sehr groß, enthalten Duplikate oder viele Versionen einer Datei, eine gegebenenfalls vorhandene Struktur ist nirgendwo dokumentiert. Die Archivarinnen müssen über die Archivwürdigkeit der Unterlagen in den Dateiablagen entscheiden, ohne gebräuchliche Hilfsmittel wie einen Aktenplan oder andere einen Überblick gebende Quellen zur Verfügung zu haben.

Der für die Bewertung nötige Überblick über die Inhalte einer solchen Ablage soll durch automatische Auswertung mit Hilfe computerlinguistischer Methoden geschaffen werden. Dafür wird eine Reihe von Methoden herangezogen. Um die weitere Verarbeitung vorzubereiten, werden die Dokumente in Sätze aufgespalten, tokenisiert und in Versionen mit und ohne Stoppwörtern gespeichert. Um die Erkennung von Eigennamen zu sichern, werden Groß- und Kleinschreibung beibehalten. Eine Volltextsuche macht die Texte der Dokumente durchsuchbar und fungiert als Bindeglied zwischen den Funktionalitäten, um beispielsweise erkannte N-

[134] Als Erschließungsrückstand wird die Menge der bereits ins Archiv übernommenen, aber noch nicht fachgerecht erschlossenen Unterlagen bezeichnet. Die Relevanz des Themas wird dadurch gut illustriert, dass sich beim Deutschen Archivtag 2017 eine ganze Sektionssitzung diesem Thema widmete. Vgl. Gillner, Bastian: Bericht zur Sektionssitzung 1.

Gramme oder Entitäten in Dokumenten zu finden. Über den Vergleich gehashter N-Gramme werden Dokumente mit großen inhaltlichen Überschneidungen identifiziert, die möglicherweise Versionen voneinander sind. Worthäufigkeiten und N-Gramme werden herangezogen, um die Sammlung dominierende Themen zu identifizieren. Mit Hilfe von Named Entity Recognition werden Personen, Orte und Organisationen in den Dokumenten gefunden. Topic Modelling, Tf-idf und Doc2Vec werden eingesetzt, um die Dokumente nach inhaltlich-textlichen Aspekten zu gruppieren. Außerdem können über Doc2Vec das Vokabular der Sammlung untersucht und einander ähnelnde Dokumente gefunden werden.

Diese Methoden wurden mit Python in einer Django-Webanwendung implementiert. Das Datenbank-Backend ist PostgreSQL, asynchrone Prozesse werden über die Aufgabenverwaltung celery verwaltet.

Die Webanwendung verwaltet und analysiert Sammlungen, die aus Dokumenten bestehen. Letztere können Duplikate oder Versionen voneinander sein. Der Sammlung und den darin enthaltenen Dokumenten können N-Gramme und Named Entitys zugeordnet werden. Voraussetzung für das Finden von N-Grammen, Entitys und Versionen ist die Ausführung verschiedener Vorverarbeitungsschritte. Es müssen zunächst die Dateien zur Sammlung hinzugefügt, auf Grundlage ihrer Volltexte verschiedene Repräsentationen der vorverarbeiteten Korpora erzeugt und zuletzt die Analysen durchgeführt werden.

Die Benutzung der Web App für den Upload-, Verarbeitungs- und Analyseprozess ist ohne größere Umstände möglich, an verschiedenen Stellen sind Usability und Performance aber ausbaufähig. Seitens der Usability sind vor allem das außerhalb der Konsole nicht zugängliche Feedback zu den asynchronen Prozessen und die fehlenden Möglichkeiten zur Verwaltung der Daten kritisch. Die Performance ist bei größeren Sammlungen schlecht, da nur begrenzt Zeit zur Optimierung der Laufzeiten zur Verfügung stand, an einigen Stellen Kompromisse zwischen Zeit- und Platzkomplexität eingegangen werden mussten und die Server-Hardware nicht für ressourcenhungrige Machine-Learning-Anwendungen optimiert war. Die Nutzung leistungsstärkerer Hardware war ebenfalls keine Option, da die verwendeten Testdaten vertraulich waren und die Verarbeitung in der Cloud somit ausgeschlossen war.

Die Vertraulichkeit der Daten ergibt sich aus ihrer Quelle: Es handelt sich um dem Hessischen Hauptstaatsarchiv Wiesbaden angebotene Dateiablagen mit personenbezogenen Daten, die noch Schutzfristen unterliegen. Insgesamt wurden vier solcher Bestände verarbeitet: Zwei Dateiablagen von Schulen und zwei durch die Staatsanwaltschaft beschlagnahmte Festplatten von Personen aus dem rechtsextremen Spektrum. Die Bestände haben unterschiedliche Größen (circa 600, 6.000, 13.500 und 150.000 Textdokumente), einer besteht zum größten Teil aus englischsprachigen Texten. Während eine der Schul-Dateiablagen sehr strukturiert ist, ist die Ordnerstruktur der übrigen Datensätze eher schleierhaft. Zu Demonstrationszwecken wurde außerdem mit einem Testbestand aus 50.000 Wikipedia-Artikeln gearbeitet.

Vorverarbeitung und Suche erfüllen bei allen Beständen ohne Probleme oder Überraschungen ihren Zweck. Die Analyse von Worthäufigkeiten, N-Grammen und Named Entitys funktioniert bei jeder Bestandsgröße, aber nur bei deutschsprachigen Beständen. Der Erkenntnisgewinn von Worthäufigkeiten und den häufigsten N-Grammen ist eher gering; nur bei völliger Ahnungslosigkeit über den Ablageninhalt würden sie wertvolle Informationen beisteuern. N-Gramme mit anderen Ähnlichkeitsmaßen als der Häufigkeit finden vor allem unübliche und fremdsprachige Begriffe in den Texten, sie steuern wenige Erkenntnisse über den Bestand bei. Named Entitys geben Auskunft über relevante Personen und Organisationen im Bestand und erlauben in Verbindung mit der Suche, sie in Dokumenten wiederzufinden. Sie sind insofern hilfreich, aber aufgrund von Problemen bei der Erkennung der Zusammengehörigkeit von beispielweise Vor- und Nachnamen oder aus mehreren Worten bestehenden Organisationsnamen sowie falschen Kategoriezuordnungen sind sie unpräzise.

Die semantische Analyse funktioniert sprachunabhängig, benötigt aber eine Mindestmenge an Text, um sinnvolle Ergebnisse zu liefern. Für Cluster basierend auf Tf-idf und für Topic Models ist der kleinste Bestand, zu dem sinnvolle Ergebnisse geliefert wurden, Bestand C mit circa 6.000 Dokumenten. Mit Doc2Vec wurde selbst mit 150.000 Dokumenten in kurzen Tests keine sinnvolle Verteilung gefunden.[135] Als

[135] Das Problem kann auch die gewählte Anzahl von Clustern gewesen sein, auch wenn mit Clustermengen von 30, 50, 70, 100 und 120 experimentiert wurde. Denkbar wäre auch, dass eine der

weitere Schwierigkeit hat sich die Unübersichtlichkeit zu großer Cluster und Clustermengen erwiesen: Bei 150.000 Dokumenten waren die Cluster für einen menschlichen Bearbeiter nicht mehr analysierbar, es werden für solche Fälle andere Strategien gefunden werden müssen. Denkbar wären zum Beispiel Anpassungen an den Bestand und eine Reduktion der zu verarbeitenden Gesamtmenge durch Beschränkung auf interessante und einzigartige Dokumente. Die Empfehlung ähnlicher Dokumente und die Analyse ähnlicher Worte auf Grundlage des Sammlungs-Vokabulars dagegen funktioniert bei allen Sammlungen außer der kleinsten zuverlässig.

Die Anwendung wurde für die Unterstützung der archivischen Bewertung entwickelt. Der Einsatz einiger der vorgestellten Methoden wäre aber auch für andere Szenarien im archivischen Arbeitsprozess denkbar. So könnte beispielweise die Erschließung durch Analyse des zu bearbeitenden Dokuments unterstützt oder den Archivbenutzern das Browsing noch nicht erschlossener Unterlagen ermöglicht werden.

An vielen Stellen hätte die Anwendung außerdem noch verbessert oder ausgebaut werden können. Es wurden nur originär digitale Textdokumente in die Verarbeitung einbezogen. Die für das Lesen der Dateien eingesetzte Programmbibliothek textract kann aber auch über Speech Recognition Tondokumente transkribieren und mittels OCR Scans oder Fotos von Textdokumenten maschinenlesbar machen. Dies würde allerdings einen Schritt zur Qualitätssicherung oder zumindest eine automatische Bewertung der Erkennungsqualität benötigen, was im Rahmen der vorliegenden Arbeit nicht geleistet werden konnte. Weitere noch einzubeziehende Quellen wären (in großem Maß in den Testbeständen vorkommende) Bild- und Videodateien, die beispielsweise automatisch annotiert oder nach Ähnlichkeit der Pixel gruppiert werden könnten. Die Metadaten der verarbeiteten Dateien wurden weitgehend ignoriert, da sie zu inkonsistent waren; ein Versuch, die Dateien auf Basis von Informationen wie ihrem Autor oder Entstehungszeitraum zu vernetzen, hätte trotzdem unternommen werden können. Das Auslesen zusätzlicher Metadaten würde auch die Anwendung von

gewählten Clustergrößen grundsätzlich funktioniert hätte, aber zufällig die Endposition von k-means nicht zielführend war; es wurde jeweils nur ein Test pro Clusteranzahl durchgeführt. Die Tests wurden nicht intensiviert, weil das Verfahren für unpraktikabel befunden wurde (siehe Kapitel 5.2.2.4).

Standards zur digitalen Archivierung ermöglichen, so dass – mit einer entsprechenden Erweiterung des Datenmodells – Pakete der als archivwürdig erachteten Dokumente zum Import in ein System zur digitalen Langzeitarchivierung erzeugt werden könnten. Auch einige der eingesetzten Methoden könnten noch verbessert werden: Die Sprache eines Dokuments könnte erkannt und die richtigen Modelle für Tokenisierung und NER ausgewählt sowie die passenden Stoppworte entfernt werden. Zur Versionserkennung hätte auf die Implementierung von MinHash LSH Ensemble bestanden werden können, um auch Versionen mit stark unterschiedlicher Seitenzahl zu identifizieren. Und die Evaluation der Ergebnisse hätte mit Hilfe eines durch einen Menschen analysierten und annotierten Testbestands verbessert und quantifizierbar gemacht werden können, wären die Ressourcen dafür vorhanden gewesen. All diese Probleme sind ungelöst und würden sich für vertiefende Arbeiten zu dieser Forschungsfrage anbieten.

Kann also mit computerlinguistischen Mitteln ein Überblick über große unsortierte Dateiablagen gewonnen werden? Der Eindruck aus den Testanalysen legt nahe, dass dies zu einem gewissen Grad funktioniert. Von einer automatischen Sortierung oder ähnlichem kann nicht gesprochen werden, dafür sind die Ergebnisse zu unzuverlässig. Aber ein menschlicher Bearbeiter kann durch die oben genannten Methoden, vor allem durch die Topic Models und die Tf-idf-Cluster, einen Eindruck von der inhaltlichen Struktur der Dateisammlung bekommen und über NER prominente Personen, Orte und Organisationen im Bestand identifizieren. Dafür müssen bestimmte Voraussetzungen erfüllt sein: Die Bestandsgröße muss innerhalb eines bestimmten Fensters liegen und eine Anpassung an seine (Haupt-)Sprache muss vorgenommen werden. Die Ergebnisse davon sind nur eine Annäherung an den Inhalt. Aber eine Annäherung ist besser als der Status Quo: unüberschaubare Ordnersysteme ohne jegliche Dokumentation. Eine Anwendung im Stile des CollectionExplorers wäre auch mit mehr Entwicklungszeit weit davon entfernt, dem Archivar die Strukturierung des Bestands abzunehmen oder eine verlässliche Zusammenfassung aller Inhalte abzuliefern. Sie sollte aber dazu taugen, den Dschungel unbekannter Dokumente ein wenig zu lichten und eine Grundlage für ihre weitere Sortierung und Einordnung zu schaffen. Und so aus einer „archivfeindlichen“ Quellengattung eine handhabbare Größe machen.

Literaturverzeichnis

Belovari, Susanne: Rasche und einfache Bearbeitung von Dateisammlungen. Ein MPLP-Ansatz. In Michael Puchta, Kai Naumann (Hrsg.): *Kreative digitale Ablagen und die Archive.* Ergebnisse eines Workshops des KLA-Ausschusses Digitale Archive am 22./23.11.2016 in der Generaldirektion der Staatlichen Archive Bayerns [Sonderveröffentlichungen der Staatlichen Archive Bayerns 13]. München 2017, S. 17–29.

Berwinkel, Holger: Zur Kanzleigeschichte des 20. Jahrhunderts. Ein Versuch. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde (Hrsg.): *Moderne Aktenkunde* [Veröffentlichungen der Archivschule Marburg 64]. Marburg 2016, S. 29–50.

Berwinkel, Holger, Robert Kretzschmar und Karsten Uhde (Hrsg.): Moderne Aktenkunde [Veröffentlichungen der Archivschule Marburg 64]. Marburg 2016.

Bird, Steven, Ewan Klein und Edward Loper: Natural language processing with python. Sebastopol 2016.

Birn, Marco: Fachverfahren - Terminologie der Archivwissenschaft. Archivschule Marburg 2015. URL: *www.archivschule.de/uploads/Forschung/ArchivwissenschaftlicheTerminologie/Terminologie.html (*15.11.2018).

Blei, David M.: Probabilistic topic models. In *Communications of the ACM* 55 (2012) H. 4, S. 77–84. DOI: 10.1145/2133806.2133826.

Blei, David M., Andrew Y. Ng und Michael I. Jordan: Latent Dirichlet Allocation. In *Journal of Machine Learning Research (*2003) H. 3, S. 993–1022. URL: *www.jmlr.org/papers/volume3/blei03a/blei03a.pdf*.

Buchholz, Matthias: Archivische Überlieferungsbildung im Spiegel von Bewertungsdiskussion und Repräsentativität [Archivhefte 35]. Köln 2011.

Bundeskonferenz der Kommunalarchive: Erstellung eines Dokumentationsprofils für Kommunalarchive. In *Der Archivar* 62 (2009), S. 122–132. URL: *www.bundeskonferenz-kommunalarchive.de/empfehlungen/Arbeitshilfe_Dokumentationsprofil.pdf (*04.12.2018).

CCSDS: Reference Model for an Open Archival Information System (OAIS). The Consultative Committee for Space Data Systems. Washington 2012. URL: *public.ccsds.org/publications/archive/650x0m2.pdf (*14.12.2018).

Chaput, Matt: Whoosh. Version 2.7.4. o. O. 2016. URL: *whoosh.readthedocs.io/en/latest/index.html (*18.09.2018).

Daelemans, Walter und Miles Osborne (Hrsg.): Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003. Morristown (NJ) 2003.

Deecke, Klara und Ewald Grothe (Hrsg.): Massenakten - Massendaten. Rationalisierung und Automatisierung im Archiv [Tagungsdokumentationen zum Deutschen Archivtag 22]. Fulda 2018.

Deutscher Bundestag (2017): Gesetz über die Nutzung und Sicherung von Archivgut des Bundes. Bundesarchivgesetz. In *Bundesgesetzblatt* 1 H. 12, S. 410–416. URL: *www.gesetze-im-internet.de/barchg_2017*.

Django Software Foundation: QuerySet API reference. Version 2.1. o. O. URL: *docs.djangoproject.com/en/2.1/ref/models/querysets/ (*15.11.2018).

Domingos, Pedro: A few useful things to know about machine learning. In *Communications of the ACM* 55 (2012) H. 10, S. 78. DOI: 10.1145/2347736.2347755.

Dunning, Ted: Accurate Methods for the Statistics of Surprise and Coincidence. In *Computational Linguistics* 19 (1993) H. 1. URL: *aclweb.org/anthology/J93-1003*.

Eric Zhu und Vadim Markovtsev: Ekzhu/Datasketch: First Stable Release. o. O. 2017.

Fechner, Martin und Andreas Weiß: Einsatz von Topic Modeling in den Geschichtswissenschaften. Wissensbestände des 19. Jahrhunderts. In *Zeitschrift für digitale Geisteswissenschaften (*2017), ohne Paginierung. DOI: 10.17175/2017_005.

Finkel, Jenny Rose, Trond Grenager und Christopher Manning: Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational*

*Linguistics (*2005), S. 363–370. URL: *nlp.stanford.edu/~manning/papers/gibbscrf3.pdf*.

Gillner, Bastian: Bericht zur Sektionssitzung 1. Verband Deutscher Archivarinnen und Archivare 2017. URL: *deutscher-archivtag.vda-blog.de/2017/09/29/bericht-zur-sektionssitzung-1/ (*15.11.2018).

Harris, Zellig S.: Distributional Structure. In *WORD* 10 (1954) H. 2-3, S. 146–162. DOI: 10.1080/00437956.1954.11659520.

Jaeger, Karina und Maria Kobold: Zwischen Datenwust und arbeitsökonomischer Bewertung. Ein Werkstattbericht zum Umgang mit unstrukturierten Dateisammlungen am Beispiel des Bestandes der Odenwaldschule. In *Der Archivar* 70 (2017) H. 3, S. 307–311.

Jain, Anil K.: Data Clustering. 50 Years Beyond K-Means. Michigan State University, Department of Computer Science & Engineering. o. O. 2008.

Kärberg, Tarvo, Karin Bredenberg, Björn Skog, Anders Bo Nielsen, Kathrine Hougaard Edsen Johansen, Hélder Silva et al.: E-ARK SIP Pilot Specification (revision of D3.2, main part of the D3.3). o. O. 2015. URL: *www.eark-project.com/resources/project-deliverables/51-d33pilotspec/file (*14.12.2018).

Kiss, Tibor und Jan Strunk: Unsupervised Multilingual Sentence Boundary Detection. In *Computational Linguistics* 32 (2006) H. 4, S. 485–525. DOI: 10.1162/coli.2006.32.4.485.

Kretzschmar, Robert: „Akten“. Begriff und Realitäten im zweiten Jahrzehnt des 21. Jahrhunderts. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde (Hrsg.): *Moderne Aktenkunde* [Veröffentlichungen der Archivschule Marburg 64]. Marburg 2016, S. 13–21.

Lafferty, John, Andrew McCallum und Fernando C.N. Pereira: Conditional Random Fields. Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning (*2001), S. 282–289. URL: *repository.upenn.edu/cis_papers/159*.

Landesarchiv Nordrhein-Westfalen: Steuerung der Überlieferungsbildung mit Archivierungsmodellen – Eine Konzeption für das Landesarchiv Nordrhein-

Westfalen 2011. URL: *www.archive.nrw.de/lav/abteilungen/fachbereich_grundsaetze/BilderKartenLogosDateien/Ueberlieferungsbildung/FK_Archivierungsmodelle_Kurzfassung_07_06_11.pdf*.

Leskovec, Jure, Anand Rajaraman und Jeffrey David Ullman: Mining of Massive Datasets. Cambridge 2014.

Liu, Bing: Web Data Mining. Exploring Hyperlinks, Contents, and Usage Data [Data-Centric Systems and Applications]. Berlin, Heidelberg 2011. URL: *dx.doi.org/10.1007/978-3-642-19460-3*.

Malmgren, Dean: textract. Version 1.6.1. o. O. 2014. URL: *textract.readthedocs.io/en/stable/ (*21.11.2018).

Manning, Christopher, Prabhakar Raghavan und Hinrich Schütze: Introduction to Information Retrieval. Cambridge 2009.

Manning, Christopher und Hinrich Schütze: Foundations of statistical natural language processing. Cambridge 2005.

McCallum, Andrew und Wei Li: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In Walter Daelemans, Miles Osborne (Hrsg.): *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. Morristown (NJ) 2003, S. 188–191.

Miegel, Annekathrin und Eva Rödel: Wege aus dem Daten-Dschungel – Bewertung und Übernahme großer Dateisammlungen. In Klara Deecke, Ewald Grothe (Hrsg.): *Massenakten - Massendaten.* Rationalisierung und Automatisierung im Archiv [Tagungsdokumentationen zum Deutschen Archivtag 22]. Fulda 2018, S. 27–36.

Miegel, Annekathrin, Sigrid Schieber und Christoph Schmidt: Vom richtigen Umgang mit kreativen digitalen Ablagen. In Michael Puchta, Kai Naumann (Hrsg.): *Kreative digitale Ablagen und die Archive.* Ergebnisse eines Workshops des KLA-Ausschusses Digitale Archive am 22./23.11.2016 in der Generaldirektion der Staatlichen Archive Bayerns [Sonderveröffentlichungen der Staatlichen Archive Bayerns 13]. München 2017, S. 7–16.

Mikolov, Tomas, Kai Chen, Greg Corrado und Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space. o. O. 2013. URL: *arxiv.org/pdf/1301.3781v3*.

Mikolov, Tomas und Quoc Le: Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning* 32 (2014) H. 2, S. 1188–1196.

Naumann, Kai: Dateisammlungen 2017 [Südwestdeutsche Archivalienkunde]. URL: *www.leo-bw.de/themenmodul/sudwestdeutsche-archivalienkunde/archivaliengattungen/sammlungen/dateisammlungen (*19.11.2018).

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel et al.: Scikit-learn. Machine Learning in Python. In *Journal of Machine Learning Research (*2011) H. 12, S. 2825–2830.

Puchta, Michael und Kai Naumann (Hrsg.): Kreative digitale Ablagen und die Archive. Ergebnisse eines Workshops des KLA-Ausschusses Digitale Archive am 22./23.11.2016 in der Generaldirektion der Staatlichen Archive Bayerns [Sonderveröffentlichungen der Staatlichen Archive Bayerns 13]. München 2017.

Řehůřek, Radim und Petr Sojka: Software Framework for Topic Modelling with Large Corpora. In René Witte, Hamish Cunningham, Jon Patrick, Elena Beisswanger, Ekaterina Buyko, Udo Hahn et al. (Hrsg.): *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta 2010, S. 45–50.

Schiller, Anne, Simone Teufel, Christine Stöckert und Christine Thielen: Guidelines für das Tagging deutscher Textcorpora mit STTS. Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart. Stuttgart 1999.

Schludi, Ulrich: Das Schriftgut der Wirtschaft. In Holger Berwinkel, Robert Kretzschmar, Karsten Uhde (Hrsg.): *Moderne Aktenkunde* [Veröffentlichungen der Archivschule Marburg 64]. Marburg 2016, S. 93–108.

SQLite: Limits In SQLite o. J. URL: *www.sqlite.org/limits.html (*15.11.2018).

Storm, Monika (Hrsg.): Transformation ins Digitale. 85. Deutscher Archivtag in Karlsruhe [Tagungsdokumentationen zum Deutschen Archivtag 20]. Fulda 2017.

Tjong Kim Sang, Erik und Fien de Meulder: Introduction to the CoNLL-2003 Shared Task. Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2003 (*2003), S. 142–147. URL: *www.clips.uantwerpen.be/conll2003/pdf/14247tjo.pdf*.

Underwood, Ted: Topic modeling made just simple enough 2012 [The Stone and the Shell. Using large digital libraries to advance literary history]. URL: *tedunderwood.com/2012/04/07/topic-modeling-made-just-simple-enough/ (*21.09.2018).

Wendt, Gunnar und Sina Westphal: Eine Herausforderung des Übergangs. Fileablagen als Quellen der digitalen Überlieferungsbildung. In Monika Storm (Hrsg.): *Transformation ins Digitale.* 85. Deutscher Archivtag in Karlsruhe [Tagungsdokumentationen zum Deutschen Archivtag 20]. Fulda 2017, S. 105–114.

Witte, René, Hamish Cunningham, Jon Patrick, Elena Beisswanger, Ekaterina Buyko, Udo Hahn et al. (Hrsg.): Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. Valletta 2010.

Anhang A Anleitung zum Serverstart

[…]

Anhang B „Clustering document vectors created by Doc2Vec“

Die Hausarbeit „Clustering document vectors created by Doc2Vec” findet sich auf der beigelegten CD-ROM unter dem Pfad „./Anhang/Saef_TRAM_DocumentClustering/ Saef_TRAM_DocumentClustering.pdf“. Ebenfalls dort zu finden sind die bei der Durchführung der Arbeit erzielten Ergebnisse.

Anhang C Topic Model zum Wikipedia-Bestand (5.2.1)

1. „berlin deutschen deutsche hamburg leipzig ddr berliner hans deutschland dresden text bar deutscher bremen hannover halle karl düsseldorf ernst carl“
2. „wurden jahre 000 jahr ersten jahren konnte jedoch später zwei zeit kam bereits erste ende gab mehr worden drei geschichte“
3. „platz saison gewann team mannschaft beim spieler spiel meister ersten spiele karriere spielte liga jahr konnte verein zwei 2007 spielen“
4. „gibt jedoch dabei mehr bzw bezeichnet siehe beim zwei verwendet beispiel oft immer form meist deutschland möglich beispielsweise denen wurden“
5. „deutscher us amerikanischer politiker siehe folgender personen name fußballspieler deutsche schriftsteller familienname komponist schauspieler österreichischer französischer maler amerikanische schweizer schauspielerin“
6. „www http al mm art kg arten cm lang weblinks html gattung com familie einzelnachweise zwei isbn china park org“
7. „isbn verlag universität münchen hrsg 978 literatur frankfurt geschichte stuttgart band berlin main leben deutschen bd auflage köln gesellschaft werke“
8. „10 12 11 15 14 13 16 18 17 20 19 30 21 22 25 23 24 26 27 28“
9. „film band album leben regie musik jahr titel weblinks frau spielte erschien jahre theater jahren serie records roman veröffentlicht vater“
10. „km liegt county meter liste weblinks provinz stadt region strecke prozent kilometer 000 insel einzelnachweise richtung bahn siehe pdf fluss“
11. „and new us york university john to for london on usa englisch city staaten james national press george american William”

12. “la jpg paris datei le di en et saint frankreich van französischen jean del san les französische bild italien rom“
13. „märz oktober april dezember januar mai november september juni juli august februar mitglied partei seit unternehmen spd jahr gewählt weblinks“
14. „2004 2000 2006 1999 2002 2005 2001 2003 1998 1997 1996 1995 1994 1990 1992 1991 1993 2007 seit 2008“
15. „ii friedrich könig johann sohn iii wilhelm heinrich karl nr graf ludwig kaiser bischof georg herzog iv familie alexander tod“
16. „1945 1933 1939 1956 1938 1944 1957 1946 1942 1958 1943 1960 1961 1950 1949 1955 1964 1934 1953 1959“
17. „gemeinde landkreis st ort stadt kreis gemeinden bahnhof liegt dorf einwohner ortsteil bayern bad straße baden nord kirche einzelnachweise rheinland“
18. „2012 2011 2010 2013 2009 abgerufen 2008 2014 fc juni 2007 juli seit einzelnachweise mai august januar com märz februar“
19. „wien min österreich franz klasse cest wiener sc österreichischen josef nürnberg österreichischer anton at 1914 karl republik max 1918 hans“
20. „stadt jahrhundert kirche heute geschichte seit wurden jahr jahrhunderts chr zeit jahre burg gebäude st befindet haus errichtet 19 wappen“

Anhang D Tf-idf-Cluster zum Wikipedia-Bestand (5.2.1)

1. –[136]
2. 2.476 Dokumente: Kunst/(Populär-)Kultur (v.a. Film/Fernsehen/Comics)
3. –
4. 1.304 Dokumente: Personen
5. 2.451 Dokumente: Listen
6. 656 Dokumente: Orte
7. 1.846 Dokumente: Orte
8. –
9. 1.495 Dokumente: Orte und Verkehr

[136] Für alle Cluster wurde nach Sichtung der Dokumente eine Überschrift gebildet, außer, es gab keinen erkennbaren Zusammenhang zwischen den Dokumenten.

10. 434 Dokumente: Kunst/(Populär-)Kultur (v.a. Musik)
11. 1.013 Dokumente: Kirche/Religion
12. 428 Dokumente: Vereinigte Staaten von Amerika (v.a. Orte)
13. –
14. –
15. 2.656 Dokumente: Kunst/(Populär-)Kultur
16. –
17. 892 Dokumente: Listen
18. 1.125 Dokumente: Orte
19. 3.189 Dokumente: Personen (v.a. Politiker)
20. 1.960 Dokumente: Sport (v.a. Fußball)

Anhang E Topic Model zu Bestand A (5.2.2.1)

1. Stunden- und Vertretungspläne[137]
2. Aufsichtsangelegenheiten
3. Budget
4. Unterricht/Unterrichtsmaterialien/Prüfungen,
5. Zeit- und Raumpläne, An-/Abmeldungen, Krankmeldungen (zahlen-/datenlastig),
6. Abiturunterlagen, Kurs-/Fachwahlen und -einteilungen
7. Budget, Buchungsbelege
8. Veranstaltungen/Termine (Prüfungen, Kurse, Kursfahrten, Wanderwochen, Schnuppertage, …)
9. Zeugnisse und Bescheinigungen
10. Dokumente, die viele Zahlen beinhalten (Haushaltsangelegenheiten, Krankenblätter, Chemie- und Biologie-Unterrichtsunterlagen, Terminpläne)
11. Dokumente mit Bezug zu einer Person (Personal/Schüler), z.B. Krankenblätter, Beurlaubungen, Auslandsaufenthalte, Nachteilsausgleiche, …), Schriftverkehr mit Eltern/Rundschreiben

[137] Die Topic-Beschreibung wurde nach Sichtung der signifikanten Worte und der enthaltenen Dokumente vorgenommen. Aus Datenschutzgründen werden die fürs Topic signifikantesten Begriffe anders als beim Wikipedia-Testbestand nicht zitiert. Alle Topics (und nachfolgend auch alle Dokumentcluster) wurden gesichtet und – sofern möglich – unter einer gemeinsamen Überschrift zusammengefasst.

12. Budget/Vermischtes (klein)

Anhang F Tf-idf-Cluster zu Bestand A (5.2.2.1)

1. 777 Dokumente: Stunden- und Vertretungspläne
2. 1384 Dokumente: Schulkonferenzen, Schülerverwaltung, Gremien, Personalverwaltung
3. 4709 Dokumente: –
4. 588 Dokumente: Aufgaben, Lösungen Gutachten
5. 589 Dokumente: (Abitur-)Aufgaben (v.a. für Leistungskurse)
6. 988 Dokumente: –
7. 461 Dokumente: Budget
8. 712 Dokumente: Verwaltung/Vermischtes (Dienstbesprechungen, Prüfungspläne und -termine, Rundschreiben, Schulkonferenzen, …)
9. 572 Dokumente: Unterrichtsinhalte (v.a. Biologie, Chemie)
10. 445 Dokumente: Krankenblätter
11. 1423 Dokumente: Unterrichtsinhalte (Lehrmaterialien, Kursprotokolle, Schüleraufgaben, Kursarbeiten, …)
12. 876 Dokumente: –

Anhang G Topic Model zu Bestand B (5.2.2.2)

1. Religion, Nationalsozialismus, deutsches Volk
2. Leserbriefe, abgespeicherten Zeitungsartikel
3. Briefe an „Kameraden“, Briefverteiler
4. Zweiter Weltkrieg, Deutschland, Amerika, Krieg, historische Themen (1920er-1940er)
5. Holocaust, Juden, Auschwitz
6. Technische Dokumente (config-Dateien, Lizenzen, Anleitungen)
7. Juden, Banken, Holocaust, Israel
8. Erinnerungen an einen verstorbenen Neonazi, Korrespondenz, Angelegenheiten mit Bezug zu einer Person
9. Nationalsozialistische Ideologie, „Kampfbund deutscher Sozialisten“

10. Inhaltsangaben von Fernsehsendungen, Dokumente bestehend aus (nicht eingelesenem) Bildmaterial[138]

Anhang H Tf-idf-Cluster zu Bestand B (5.2.2.2)

1. 300 Dokumente: Abgespeicherte Zeitungsartikel, Leserbriefe
2. 971 Dokumente: Korrespondenz mit „Kameraden“
3. 297 Dokumente: Holocaust, Juden
4. 331 Dokumente: Adressen (von Privatpersonen, Zeitungen und Organisationen, z.T. ausgezeichnet als Presseverteiler)
5. 516 Dokumente: Abgespeicherte Zeitungsartikel, Leserbriefe
6. 1431 Dokumente: –
7. 404 Dokumente: Juden, Israel, Holocaust
8. 984 Dokumente: Korrespondenz
9. 744 Dokumente: Deutschland, historische Themen (1920er-1940er), Amerika, Krieg, Globalisierung
10. 155 Dokumente: Leserbriefe aus dem „National Journal“

[138] Wenn in einem Word-Dokument ein Bild enthalten ist, so wird es im Dokument mit dem Text „IMAGE“ repräsentiert.