Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 251

Erweiterte Word-Cloud-Visualisierung zur Analyse von mehreren Texten

Eduard Marbach

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dr. Steffen Lohmann, Dipl.-Ling. Florian Heimerl
Beginn am:	11. Juli 2015
Beendet am:	11. Dezember 2015
CR-Nummer:	H.5.2, I.7.5

Kurzfassung

Die Word-Cloud ist eine mächtige Visualisierungsmöglichkeit, um Texte übersichtlich und anschaulich darzustellen. Um einen vergleichenden Überblick von mehreren Texten zu ermöglichen, ist mit einer üblichen Word-Cloud schwierig. Deshalb ist eine erweiterte Word-Cloud, die mehrere Dokumente darstellt, eine Erleichterung. Gemeinsam auftretende Wörter in ihrer Relevanz sortiert, geben einen guten Überblick über die gemeinsamen Themen der Dokumente. Bereits vorhandene Visualisierungen haben sich mit diesem Thema befasst und dazu Konzepte entwickelt. Verschiedene Mängel dieser Ansätze machen es notwendig weitere Forschung zu betreiben. Im Rahmen dieser Arbeit wurde eine Multi-Dokumente-Visualisierung entwickelt, die eine genaue Zuordnung der Worte ermöglicht und den freien Raum optimal ausnutzt. Die generierten Word-Clouds sind optimal verteilt und überspringen keine Wörter, falls für diese kein freier Raum existiert. Anhand der optimalen Verteilung soll der verfügbare Platz bestmöglich ausgenutzt werden, um leere Flächen zu vermeiden. Der Ansatz wird durch eine Expertenmeinung evaluiert und die Auswertung dargestellt. Anschließend wird die Nützlichkeit und Verbesserungen des Ansatzes anhand von Anwendungsfällen aufgezeigt.

Abstract

Word Clouds are powerful visualization techniques to sum up text in a clear and descriptive way. To give a comparable overview over multiple documents the standard Word Cloud is not sufficient. Therefore an extended Word Cloud is needed to visualize multiple documents in a comprehensible way. Common words sorted in their relevance give an overview over the combined topics of the documents. Other works designed concepts to address this problem. But the approaches has deficiencies. In order to solve those this work was created which optimizes the usage of the available space. Additionally the work provides an easy association of words to the different documents at any time. The generated Word Clouds are evenly distributed and do not skip any words if the space would be not sufficient. Based on the even distribution of the words the available space should be optimally filled to reduce the white space. With a user study with experts the concept is evaluated and the results are illustrated. Afterwards the usefulness and improvements of the concept is shown with use cases.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

1. Einleitung

Dokumente und Dokumentmengen sind vielfältige Informationsquellen. Die Analyse und Extraktion der Informationen aus der Dokumentenmenge ist ein aufwendiger und langer Prozess. Dabei muss jeder Satz gelesen und untersucht werden. Einen vergleichenden Überblick über mehrere Dokumente zu bekommen, wird damit noch weiter erschwert. Ohne Extraktion der wichtigsten Informationen und Zusammenfassen zu Notizen lassen sich die Dokumente nur schwer vergleichen.

Zur Zusammenfassung von Texten können Visualisierungen verwendet werden. Die bekannteste unter diesen ist die Word-Cloud. Durch eine automatisierte Verarbeitung der Texte werden die Wörter gezählt und als eine Liste mit Gewichtungen zusammengestellt. Anschließend werden die am häufigsten vorkommenden Wörter dargestellt. Die relevanten Wörter erhalten dabei eine größere Schriftgröße, um ihre Relevanz hervorzuheben. Zur Analyse von mehreren Texten müssen die Dokumente gemeinsam verarbeitet und die Word-Cloud mit den neuen Informationen erweitert werden. Durch die Darstellung aller Dokumente innerhalb einer Word-Cloud lässt sich mithilfe von Metadaten die Zugehörigkeit von den Worten zuordnen. Dadurch lassen sich die Dokumente miteinander vergleichen und Gemeinsamkeiten untersuchen.

In dieser Arbeit wird ein leichtes Vergleichen von Wörter ermöglicht und der verfügbare Platz wird optimal befüllt. Die vorhandenen Visualisierungen RadCloud und ConcentriCloud werden untersucht und anschließend werden die Mängel behoben und optimiert. Anhand einer Expertenstudie werden die Ergebnisse validiert.

Gliederung

Die Arbeit ist in folgender Weise gegliedert:

Kapitel 2 – Grundlagen: In diesem Teil werden die Grundlagen beschrieben, die für ein tiefes Verständnis der entwickelten Visualisierung notwendig sind. Dazu zählen das Konzept einer Word-Cloud, die unterschiedlichen Layouts mit deren Bewertung und die natürliche Sprachverarbeitung.

Kapitel 3 – Verwandte Arbeiten: Hier werden die verwandten Ansätze zu Multidokumenten-Word-Clouds beschrieben.

Kapitel 4 – Konzept: Dieses Kapitel beschreibt die konzeptionellen Ideen des Ansatzes. Es werden die unterschiedlichen Layoutmöglichkeiten erläutert und verschiedene Platzierungsalgorithmen analysiert.

Kapitel 5 – Implementierung: In diesem Teil wird die Umsetzung der Arbeit beschrieben und Prototypen dargestellt. Es wird auf die verwendeten Technologien und Bibliotheken eingegangen und die prototypischen Umsetzungen aufgezeigt.

Kapitel 6 – Anwendungsbeispiel: Hier werden mögliche Anwendungsgebiete der Visualisierung anhand von Beispielen dargestellt.

Kapitel 7 – Evaluation: In diesem Abschnitt wird die durchgeführte Nutzerstudie mit Experten vorgestellt und die Ergebnisse erläutert. Diese umfassen die Einschätzungen der Probanden und mögliche Verbesserungsvorschläge.

Kapitel 8 – Zusammenfassung und Ausblick: Zum Schluss wird eine kurze Zusammenfassung gegeben, mögliche Erweiterungen und Verbesserungen diskutiert.

2. Grundlagen

Eine Word-Cloud-Visualisierung besteht aus einer Liste mit Wörtern und deren Gewichtungen. Die Gewichtung kann dabei beliebig ausfallen, ob in einer prozentualen Aufteilung von 0 bis 100 oder der Anzahl der Vorkommnisse. Dies spielt keine Rolle, da die Informationen je nach Verlangen verarbeitet werden können. Ein Beispiel für eine gewichtete Liste ist in Tabelle 2.1 zu sehen.

Wort	Frequenz
Wandlung	50
Filterung	24
Sachverhalt	5

Tabelle 2.1.: Beispiel für eine gewichtete Liste.

Diese Liste wird dann im einfachsten Fall mithilfe einer Funktion auf eine Schriftgröße abgebildet. Anschließend werden die Wörter auf einer Fläche angeordnet. Eine so entstandene Word-Cloud enthält je nach verwendeter Skalierungsfunktion oder Verteilung auf der verfügbaren Fläche kaum verwertbare Informationen. Das liegt daran, dass die Position des Wortes randomisiert berechnet wird und dadurch keine Bedeutung hat.

Viele Forschungen und Studien haben sich bereits mit Word-Clouds beschäftigt, um unterschiedliche Aspekte bezüglich Wahrnehmung und Auffassung zu analysieren. Die meisten Word-Cloud-Visualisierungen zeigen dabei viele Gemeinsamkeiten. Beispielsweise werden die Relevanzen durch die Schriftgröße ausgedrückt.

Ebenso kommen auch Aspekte wie die Wahl des Layouts einer Word-Cloud in Betracht oder Aspekte der Word-Clouds, die nicht positiv ausfallen. In den nachfolgenden Punkten werden diese genauer dargestellt und erläutert.

Die meisten Word-Cloud-Visualisierungen weisen dabei viele Gemeinsamkeiten auf. Es ist durch Studien [BGN08a] gezeigt, dass die Darstellungen der Relevanzen mithilfe der Schriftgröße am Besten umgesetzt werden kann. Mit der Schriftgröße einhergehend spielt auch die Skalierung der Worte eine wichtige Rolle. Hier wurde ebenfalls mithilfe von Studien gezeigt, dass eine Wurzelskalierung eine verlustfreie Skalierung darstellt.

2.1. Visualisierung von Wörtern

Die Wissenschaft von Wortdarstellungen, oder auch Tags genannt, einer Word-Cloud spielt eine zentrale Rolle. Die Wörter können dabei in allen möglichen visuellen Attributen variieren. Dazu zählen Schriftart, Schriftgröße, Schriftstärke oder Schriftfarbe. Jedes dieser einzelnen Attribute hat dabei unterschiedlich starke Auswirkungen auf die Wahrnehmung der Wörter. Je nach Wahl der Attribute kann dies einen großen Einfluss haben.

Bateman et al. [BGN08b] beschäftigen sich mit den unterschiedlichen Wahrnehmungen dieser Attribute. Es wurden Studien und Tests durchgeführt, wie schnell sich relevante Wörter aus der gesamten Word-Cloud erkennen lassen. Ergebnisse dieser Studien waren, dass sich die Variation in der Schriftgröße am besten eignet, um die Relevanzen darzustellen. Große Wörter werden viel schneller aufgefasst, wobei die kleineren ignoriert werden. Attribute wie Farbänderungen schneiden dabei sehr schlecht ab und erzeugen nicht den gewünschten Effekt. In den meisten Word-Clouds ist daher verbreitet die Schriftgröße als Darstellung für die Relevanzen zu sehen.

Die Wahl der Schriftart sollte mit Betracht auf die Lesbarkeit fallen. Künstlerische Schriftarten finden ihre Verwendung, wenn es um eine ästhetische Darstellung der Word-Cloud geht und der Fokus dabei nicht auf den Wörtern liegt. Daher sollte eine neutrale Schriftart für die Lesbarkeit genutzt werden. Zu denen zählen beispielsweise Tahoma oder Arial.

Einen wichtigen Punkt nimmt dabei auch die Skalierung der Wörter ein. Sie können linear, quadratisch oder auch nach beliebigen Funktionen skalieren. Dabei verändert sich der Eindruck, da die Wörter verglichen werden, um die Wichtigkeit anhand der Schriftgröße zu beurteilen. Der visuelle Eindruck kann sich dabei stark unterscheiden. Wenn sich ein kleiner Unterschied in der Frequenz stark in der Schriftgröße auswirkt, bekommt der Betrachter den Eindruck, dass sich die Wörter bezüglich ihrer Relevanz stark unterscheiden müssen. Stattdessen erzeugt ein geringer Unterschied in der Schriftgröße den Eindruck, dass die Wörter ungefähr gleich wichtig sein müssen. Dies kann durch beliebige Funktionen so variiert werden, wie es gewünscht ist.

Im Zusammenhang mit der Skalierung geht auch der Begriff *Visual Lie* [1]einher. Dies ist ein Begriff für die visuellen Fehlinformationen oder auch Lügen, die eine Visualisierung/Abbildung enthalten kann. Ein Beispiel wo dies zutrifft ist die Skalierung, wenn sich die Schriftgrößen im höheren Frequenzbereich stärker verändern und hingegen im kleineren kaum einen Einfluss hätten. Das kommt in dem einfachsten Beispiel zustande, wenn die Skalierungsfunktion nicht monoton verläuft.

[1] http://www.infovis-wiki.net/index.php/Lie_Factor

Abbildung 2.1.: Ein Beispiel für verschiedene visuelle Attribute einer Word-Cloud. [2]

Die Rotation von Wörtern ist eine weitere, wenn auch kritische, Möglichkeit Wörter zu variieren. Dabei werden Wörter je nach Gradzahl rotiert, um mögliche Freiräume zu füllen (siehe Abbildung 2.1). Der Rotationsradius kann dabei je nach Implementierung manuell festgelegt werden oder wird zufällig berechnet. Jedoch ist die daraus resultierende schlechte Lesbarkeit negativ zu bewerten. Der Betrachter muss den Blick entsprechend der Wortrotation anpassen, um ein Wort lesen zu können. Die Verwendung von Rotation kann zwar den ästhetischen Eindruck einer Word-Cloud aufwerten, verschlechtert zugleich jedoch immer die Lesbarkeit und benötigt mehr Zeit.

2.2. Layouts von Word-Clouds

Die Nutzung von unterschiedlichen Layouts bietet verschiedene Möglichkeiten wie eine Word-Cloud Informationen präsentieren kann. Die bekanntesten Layouts sind dabei ein rechteckiges, kreisförmiges und geclustertes Layout.

Die Blickrichtung des Nutzers unterscheidet sich dabei stark zwischen den Layouts. Wie in Abbildung 2.2 zu sehen liegt der Fokus des Nutzers beim rechteckigen Layout in der oberen linken Ecke. Die Blickrichtung des Nutzers wandert dabei zu unteren rechten Ecke. Die meiste Zeit liegt der Blick des Nutzers jedoch im oberen linken Bereich.

Im Gegenteil zum rechteckigen Layout liegt der Fokus bei dem kreisförmigen Layout im Zentrum. Hier verweilt der Blick des Nutzers die längste Zeit und wandert nur kurz zu den äußeren Bereich des Kreises. Es ist üblich, dass bei dem kreisförmigen Layout die relevanten Wörter im Mittelpunkt platziert werden und die Schriftgröße nach außen hin abnimmt. Dadurch wird die Informationen schnell erfasst und aufgenommen (siehe Abbildung 2.2).

[2] http://livespeaklove.com/2012/05/21/fun-with-word-clouds/

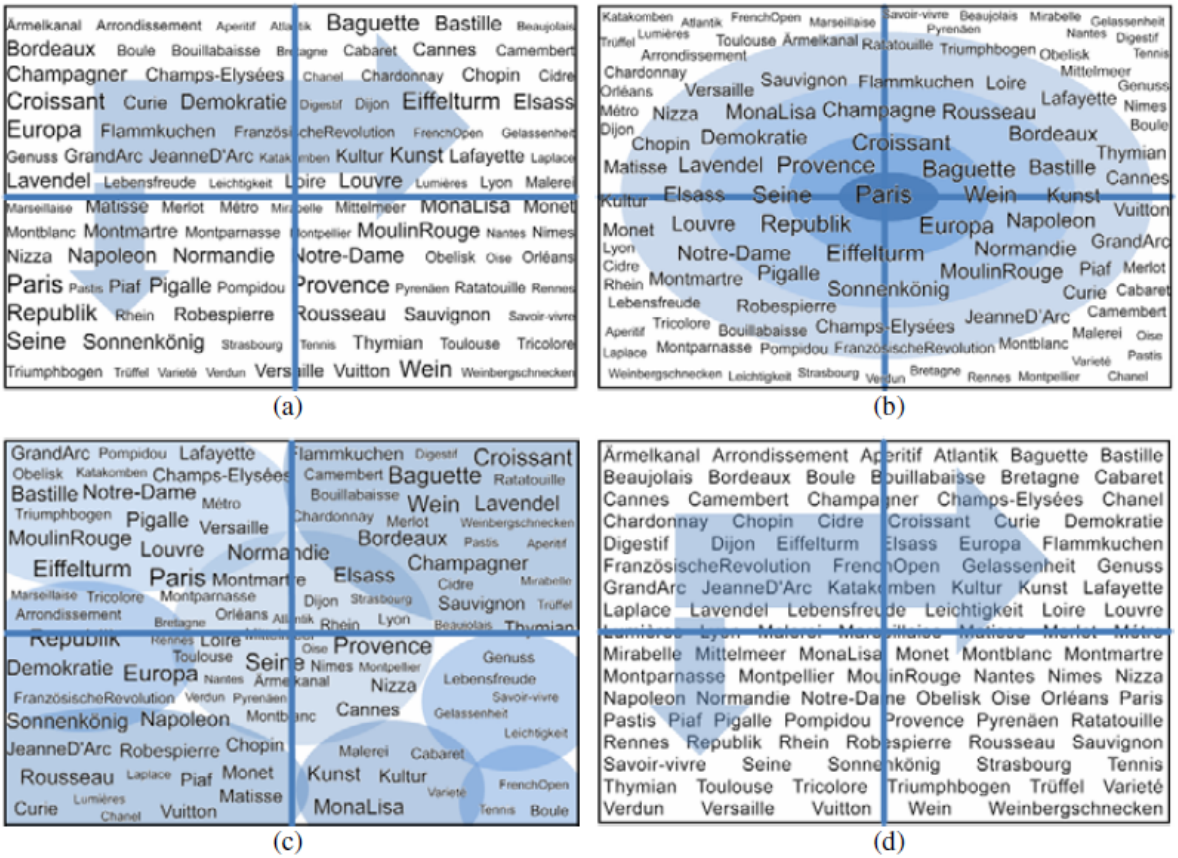


Abbildung 2.2.: Ergebnisse der Lohmann et al. [BGN08a] Studie über die Fokussierungen des Betrachters.

Bei dem geclusterten Layout bilden sich Gruppen von Wörtern, die Cluster darstellen und eigene Themen oder Bereiche repräsentieren. Daher unterscheidet es sich von den beiden zuvor erwähnten Layouts. Dieses Layout lässt sich nicht auf alle Word-Clouds anwenden, da die Daten die erforderlichen Metadaten benötigen, um sie zu einem Cluster zuordnen zu können. Ein Clusterlayout mit einem einzigen Cluster würde sich nicht von einem rechteckigen oder kreisförmigen Layout unterscheiden. Da sich diese Arbeit mit dem Fokus auf Multidokumenten-Visualisierungen beschäftigt, wird das Clusterlayout eine große Rolle einnehmen. Dies wird in Kapitel 5 weiter erläutert.

Diese Erkenntnisse und Studien zu den Layouts und ihrem Blickverlauf wurden von Lohmann et al. [BGN08a] untersucht und bewertet. Zusätzlich wurden noch Methoden wie das Eye-Tracking verwendet, um die Ergebnisse zu unterstützen.

2.2.1. Kräftebasiertes Layout

In Zusammenhang mit Word-Clouds kann auch das kräftebasierte Layout betrachtet werden. Ein kräftebasiertes Layout besteht aus Knoten, Kanten und einwirkenden Kräften. Die Knoten haben dabei abstoßende oder anziehende Kräfte und werden durch die Kanten zusammengehalten. Liegen zwei Knoten nah nebeneinander und besitzen abstoßende Kräfte, werden diese solange auseinander gedrückt bis die Kraft ausgeglichen ist. Dies geschieht so lange bis ein Kräftegleichgewicht entstanden ist. Als Ergebnis liegen die Knoten auf einer maximalen Distanz zueinander und werden durch die Kanten zusammengehalten.

Mit Blick auf Word-Clouds können Knoten als Wörter repräsentiert werden. Diese müssen im Idealfall eine so hohe Abstoßung aufweisen, dass eine Überlappung der Wörter nicht auftreten kann. Die Kanten können dabei je nach Bedarf verwendet werden. Da die visuelle Darstellung der Kanten keinen Nutzen aufweisen, werden diese üblicherweise nicht dargestellt und nur als einwirkende Kraft verwendet. Dabei können alle Wörter miteinander verbunden sein, um die Word-Cloud kompakt zu halten. Ein weiteres Beispiel wäre die Verbindung aller Wörter zu einem visuellen Mittelpunkt, um den sich die Wörter ansammeln.

2.3. Darstellungsvielfalten

Eine Word-Cloud, die nur in Schriftgröße und Layout variiert, stellt nicht alle visuellen Möglichkeiten einer Word-Cloud dar. Die Word-Clouds können noch mit zusätzlichen Informationen angereichert werden. Dies ermöglicht es verschiedene Aspekte in die Word-Cloud zu integrieren.

Abbildung 2.3.: Beispiel für Glyphen anhand von SparkCloud [LRKC10].

Eine Form dieser visuellen Erweiterung ist die Nutzung von Glyphen pro Wort. Dabei kann ein Glyph im Hintergrund, Vordergrund oder in der Nähe des Wortes platziert werden. Das würde Möglichkeiten bieten wie die Darstellung von Zeitereignissen oder das Vorkommen der Wortes innerhalb der Datenquelle. Wie in Abbildung 2.3 zu sehen, werden die Informationen dargestellt ohne die Aufmerksamkeit des Nutzers komplett auf sich zu richten. Der Verwendung der Glyphen sind dabei keine Grenzen gesetzt. Je nach Integrationsart können dabei simple Diagramme wie ein Balkendiagramm oder komplexere Glyphen verwendet werden.

Zu den weiteren Möglichkeiten der Darstellung gehört die Integration der Word-Cloud in eine vorhandene Visualisierung. Ein Beispiel für die Art ist die Integration der Word-Cloud

in eine Landkarte. Wörter würden beispielsweise innerhalb der Ländergrenzen gezeichnet werden. Eine weitere Möglichkeit wäre die Darstellung auf einer Zeitachse mit einzelnen Word-Clouds. Die Word-Cloud reichert dabei die Informationen der vorhanden Visualisierung an.

2.4. NLP

NLP steht für *Natural Language Processing* und bedeutet, dass natürliche Sprache automatisiert verarbeitet wird [Cho03]. Dabei werden Satzstrukturen analysiert, Wortstämme verglichen und generiert und auch einzelne Wörter in ihrer Bedeutung untersucht. Es können Personen, Währungen und andere Objekte durch die automatisierte Verarbeitung erkannt werden, wobei sich unterschiedliche Methoden in ihrer Qualität stark unterscheiden können. NLP-Tools sind kompakte Anwendung zur Verarbeitung von Texten, die alle Komponenten für die Sprachverarbeitung enthalten und in einer Pipeline abarbeiten. Die NLP-Tools bedienen sich dabei dem Konzept der Annotation. Eine Annotation ist dabei eine zusätzlich hinterlegte Informationen zu einem Wort, in diesem Fall ein Wort, Satz oder Satzkonstellationen. Diese Annotationen werden meist als Baumstrukturen hinterlegt und bearbeitet.

Die Grundfunktionen aller NLP-Tools besteht dabei aus dem Trennen der Sätze in einzelne Satzzeichen oder Worte, dem sogenannten *Tokenizer*. Anschließend oder auch bereits vor dem Tokenizer werden Sätze erkannt und als solche annotiert. Ab diesem Zeitpunkt kann es zu Variationen zwischen unterschiedlichen Tools kommen. Bekannte und wichtige Verfahren bezüglich dieser sind die *POS-Tagger* und *NER*. Der POS-Tagger ist wichtig, da er die Satzstrukturen in die grammatikalischen Komponenten zerteilt und die NER findet namentlich genannte Objekte.

POS-Tagger POS steht hierbei für *Part-of-Speech*. Wie der Name es bereits verdeutlicht, besteht die Aufgabe des POS-Taggers darin den Text in seine grammatikalische Struktur zu zerlegen und jedes Wort in der Bedeutung zu untersuchen. Wörter werden als Nomen, Verben, Adjektive, etc. annotiert.

Das POS-Tagging ist mitunter die schwierigste Aufgabe eines NLP-Tools. Die Verarbeitung der Sprache ist ein sehr komplexes Forschungsgebiet. Es ist möglich dieselbe Bedeutung in unzähligen verschiedenen Sätzen wiederzugeben. Für eine Maschine, die die Bedeutung nicht kennt und sich rein an dem Text und der Reihenfolge von Wörtern orientieren kann, ist es eine komplizierte Aufgabe.

NER NER ist die Kurzform für *Named Entity Recognition*. Dieser Verarbeitungsschritt eines NLP-Tools hat das Ziel Personen, Orte oder andere benannte Entitäten zu erkennen. Wenn in einem Text über „Max“ gesprochen wird, sollte das NLP-Tool erkennen, dass es sich dabei um eine Person handelt. In einer fortgeschrittenen Erkennung erkennt das Tool sogar, dass es um eine männliche Person handelt.

Weitere Aufgaben sind zum Beispiel *Coreference Resolution*[3], Beantwortung von Fragen oder die Extraktion von Beziehungen. [4] Die Arbeitsschritte nehmen dabei an Komplexität zu je später der Verarbeitungsschritt einsetzt.

2.5. Negative Aspekte

Word-Clouds können auch negative Aspekte aufweisen. Die zuvor erwähnten Visual Lies entsprechen einem negativen Punkt. Durch Anwendung von falschen Skalierungen können die Informationen falsch überliefert werden und erzeugen damit einen falschen Eindruck, den der Ersteller der Word-Cloud möglicherweise nicht erreichen wollte. Es muss beachtet werden, wie sich Falschinformationen in die Word-Cloud finden oder vermeiden lassen, damit keine ungewünschten Effekte auftreten.

Ein weiterer Fehleindruck entsteht, wenn die Wörter falsch angeordnet sind und so die Information verkehrt aufgenommen werden. Das andere Szenario ist, dass die gewünschten Informationen überhaupt nicht überbracht werden. Die entstandene Word-Cloud liefert dabei keine verwertbaren Informationen. Dies wurde auch von Jacob Harris einem erfahrenen Softwarearchitekt der New York Times angeprangert. Er findet, dass Word-Clouds gefährlich sein können und oftmals für die falschen Zwecke eingesetzt werden.[5]

[3] Steht für die Zuordnung von Entitäten über Sätze und Abschnitte hinweg.
[4] http://nlp.stanford.edu/software/corenlp.shtml
[5] http://www.niemanlab.org/2011/10/word-clouds-considered-harmful/

3. Verwandte Arbeiten

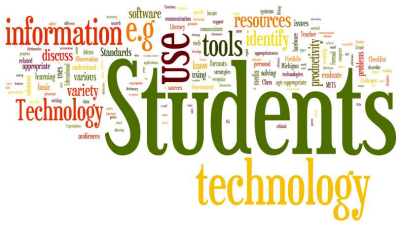
Es gibt viele unterschiedliche Visualisierungen, die den Ansatz der Word-Cloud verwendet und variiert haben. Je nach Fokus können sich die Visualisierung sehr stark unterscheiden. Es können beispielsweise geografische oder temporale Themen mithilfe von Word-Clouds visualisiert werden. Genauso können manche Visualisierung ihren Fokus auf die ästhetischen Aspekte setzen, wie es bei Wordle der Fall ist.

Diese Arbeit beschäftigt sich mit dem Thema der Multidokumenten Visualisierung. Das Thema ist eng mit der Clustervisualisierung verwandt und hat einige Bereiche, bei denen sich diese Kategorien ähnlich sind oder nur gering unterscheiden.

Zu den wichtigen vorhandenen Arbeiten gehören die RadCloud (Abschnitt 3.6) und ConcentriCloud (Abschnitt 3.5) Visualisierungen.

Im Folgenden werden die verwandten Arbeiten dargestellt und am Ende des Kapitels ein kurzes Resultat gezogen.

3.1. Wordle



a)

b)

Abbildung 3.1.: Beispiele für Wordle Visualisierungen einmal als normale Form und in einer definierten Form.[1]

[1] http://newteach.pbworks.com/w/page/11763529/Wordle und http://www.pointit.com/blog/a-picture-that-can-paint-a-thousand-words/

Die wohl bekannteste Visualisierung von Word-Clouds ist Wordle [FSI10] [VWF09]. Wordle wird auf vielen Websites oder auch zu rein privaten Zwecken verwendet, um möglichst schnell visuell ansprechende Word-Clouds generieren zu können. Es bietet Unterstützung hinsichtlich der Darstellung der Wörter durch Farben, Schriftarten oder Schriftgrößen. Auch die Form der Word-Cloud lässt sich beeinflussen und in einer Form darstellen wie in Abbildung 3.1 zu sehen. Dadurch lässt sich eine Word-Cloud nach Belieben generieren.

In Wordle wird dabei ein Algorithmus verwendet, der die Wörter in einer Spiralform platziert. Die genauen Werte von der Laufbahn der Spirale und den Testpunkten für die Positionierung werden mit Zufallszahlen bestimmt. Der Algorithmus von Wordle ist nicht frei zugänglich und wurde so optimiert, dass sich in schneller Laufzeit ansehnliche Word-Clouds generieren lassen [FSI10].

Da die Positionierung zufällig abläuft und die Position keine genaue Aussage hat, ist der Ansatz nur bedingt für die Visualisierung von Multidokumenten-Word-Clouds geeignet.

3.2. TagClusters

TagClusters ist eine Visualisierung, um die Zusammenhänge von Clustern darzustellen. Dabei werden auch Strukturen und Zusammenhängen von Wörtern und Clustern dargestellt. Dabei entspricht die Farbfläche einem Cluster, das durch ein einen zentralen blaugefärbtem Begriff zusammengefasst [CSBT09]. Dies ermöglicht einen schnellen Überblick über die Hauptkategorien ohne auf die genauen Wörter einzugehen. Dies erspart Suchzeit, falls das Thema keine Relevanz für den Betrachter hat.

Sollten sich bestimmte Wörter nicht nur in ein Cluster einordnen lassen, so werden diese durch überlappende Cluster dargestellt. Der Farbton wird je nach Anzahl der Überschneidung dunkler und intensiver.

Die Größe der Wörter werden nicht nach der Häufigkeit des Auftretens bestimmt, sondern nach der Anzahl der Kookurrenzen eines Wortes. Dies soll verhindern, dass bekannte Themen möglicherweise größere Themen verdecken und so die Informationen verfälschen können.

Wie in Abbildung 3.2 zu sehen, werden Cluster für unterschiedliche Musikrichtungen gebildet. Der Ansatz ermöglicht eine gute Unterscheidung zwischen Clustern und dadurch erkennt der Nutzer eine Zuordnung. Jedoch ist auch zu sehen, dass der vorhandene Platz nicht optimal genutzt wird und noch viel weißer Freiraum übrig bleibt.

Durch den erwähnten Ansatz ist eine Funktionalität für Multidokumenten-Zwecke sehr beschränkt. Es würde möglicherweise mit einer geringen Anzahl an Dokumenten gut funktionieren, aber mit steigender Anzahl steigen die Schnittbereiche so stark an, dass eine klare Unterscheidung der Bereiche und damit eine Zuordnung von Tags nicht mehr möglich wird.

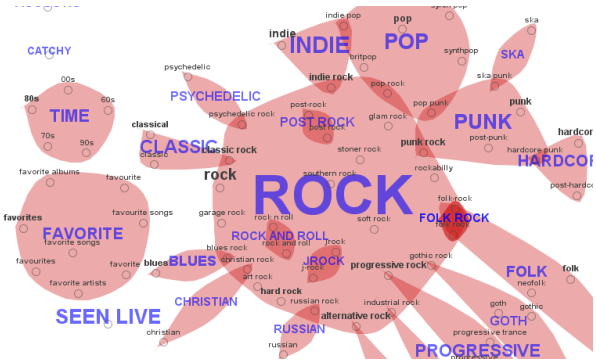


Abbildung 3.2.: Eine geclusterte Darstellung von unterschiedlichen Musikrichtungen mit Subgenre. [CSBT09]

3.3. DocuBurst

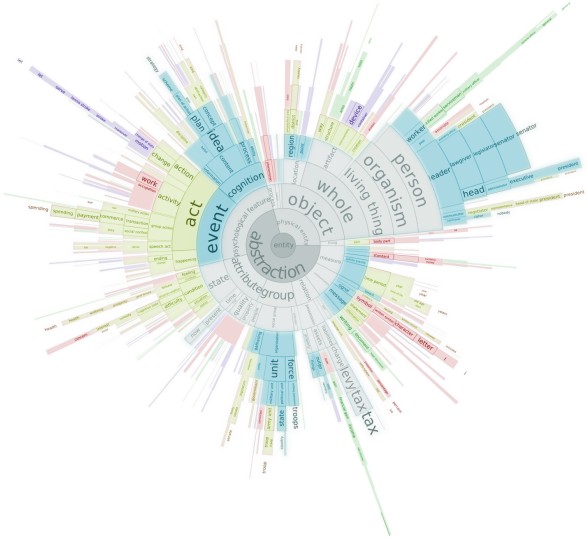


Abbildung 3.3.: Die hierarchische Darstellungen von Themen von analysierten Texten.[2]

Die DocuBurst Visualisierung basiert auf der sunburst[3]-Visualisierung [CCP09]. Damit wird die Hierarchie von unterschiedlichen Inhalten und Konzepten in Texten dargestellt.

Für die Extraktion von den Themen und Wörtern wird eine Vorverarbeitung von den rohen Texten durchgeführt. Dabei wird der zu analysierende Text in zusammenhängende Abschnitte zerlegt und so immer weiter unterteilt. Dabei entsteht auch die benötigte hierarchische

[2] http://vialab.science.uoit.ca/wordpress/wp-content/uploads/2014/10/1414705453docuburst-treecut-small2.png
[3] http://www.cc.gatech.edu/gvu/ii/sunburst/

Struktur für die Visualisierung. Zusätzliche werden noch Wortarten analysiert und die Wortstämme der Wörter analysiert. Diese nun verarbeiteten Daten werden in der Visualisierung verwendet.

Unterschiede oder Gemeinsamkeiten von Dokumenten lassen sich mit einer hierarchischen Darstellung nur schwer darstellen. Mithilfe von Interaktion und Hervorhebungen lassen sich Verbindungen zwischen unterschiedlichen Wörtern darstellen und ermöglichen es die Hierarchie nachzuverfolgen. Es ist aber bei größeren Verschachtelungen schnell klar, dass die Übersicht der Visualisierung darunter leidet und der Nutzer die Informationen nicht mehr schnell verarbeiten kann. Wie in Abbildung 3.3 zu sehen, können sich die Ebenen sehr weit schachteln. Mit zunehmender Ebene nimmt die Schriftgröße der Wörter zusätzlich noch immer weiter ab, so dass die Wörter nur noch mithilfe von Interaktion zu lesen sind. Auch können die Ebenen nur einzelne Wörter enthalten, wie sich am Rand der Visualisierung erkennen lässt.

3.4. Parallel Tag Clouds

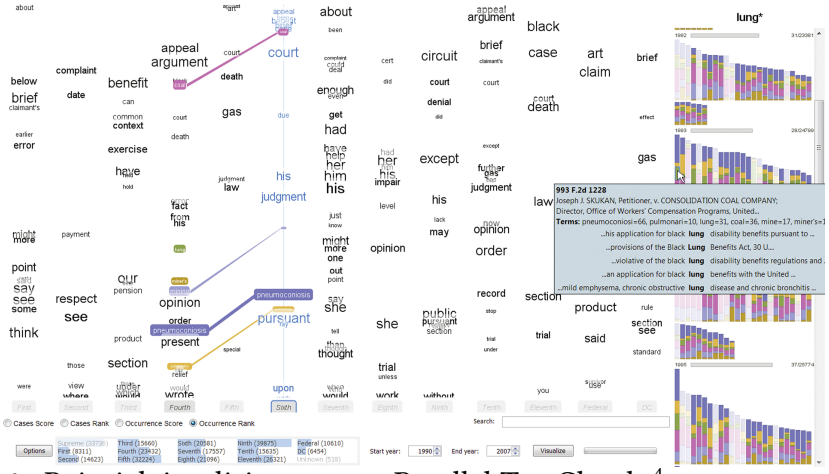


Abbildung 3.4.: Beispielvisualisierung von Parallel Tag Clouds.[4]

Einen anderen Ansatz liefert die Parallel Tag Clouds-Visualisierung [CVW09]. Dabei ist die Visualisierung umfangreich und bietet eine schwer überschaubare Anzahl an Interaktionsmöglichkeiten, um den eingelesenen Inhalt zu explorieren. Es werden unterschiedliche Modi für die Skalierung der Wörter angeboten, entweder nach Anzahl oder Relevanz.

[4] http://vialab.science.uoit.ca/portfolio/parallel-tag-clouds-to-explore-faceted-text-corpora

Der zeitliche Verlauf eines Wortes wird dabei mit dem *Parallel Coordinates* Ansatz visualisiert. Zusammen mit der Liniendicke werden die Auftreten der Wörter dargestellt. Zudem unterstützen noch weitere Diagramme die Visualisierung. Balkendiagramme zeigen die Verteilung von Wörtern in den einzelnen Clustern an.

Eine vertikale Linie kann dabei einem oder auch mehreren Dokumenten entsprechen. Die Wörter auf dieser Linie sind je nach Auswahl geordnet. Dabei werden zum Beispiel die alphabetische Sortierung oder die Sortierung nach zeitlichem Auftreten angeboten.

Diese sind nur ein Teil der genannten Interaktionsmöglichkeiten, die Parallel Tag Clouds anbietet. In Abbildung 3.4 ist ein Teil und die Vielfalt der Interaktion zu sehen. Die Information wird mit unterschiedlichen Mitteln dargestellt.

In Betracht auf die Multidokumenten-Word-Clouds ist der Ansatz von Parallel Tag Clouds nicht anwendbar, da die Informationen sehr spärlich auf ein einer vertikalen Linie angezeigt werden können. Die Zuordnung und Verteilung zu unterschiedlichen Dokumenten lässt sich zudem nur mithilfe der Interaktion erschließen.

3.5. ConcentriCloud

ConcentriCloud bietet eine gute Visualisierung mit dem Fokus auf der Darstellung von einzelnen Dokumenten und dem Schnittbereich aller Dokumente [LHB+15]. Dies wird mithilfe von Kreissegmenten und einem zentralen Bereich im Kreismittelpunkt erreicht. Die Visualisierung besteht im wesentlichen aus drei Kreisringen.

Abbildung 3.5.: Visualisierung der sechs Harry Potter Romane. [LHB+15]

Einzelne Kreissegmente entsprechen dabei einzelnen Dokumenten, die durch klare Linien getrennt werden und damit harte Grenzen bilden. Hier werden nur die Wörter angezeigt,

die einzigartig in den benachbarten Segmenten auftreten. Gegenüber liegende Dokumente können dabei gleiche Wörter enthalten.

Im inneren Bereich kommen jedoch nur Wörter vor, die auch mindestens einmal in jedem Dokument auftreten.

Der mittlere Kreisring entspricht dabei Schnittmengen von Dokumenten, die in Richtung der Kreismitte immer größere Schnittmengen bilden. Dieser Ring hat keine harten Grenzen wie der Bereich der Dokumente und lässt daher keine genaue Rückschlüsse auf die Zugehörigkeit zu. Ein weiterer Punkt ist der wenig verfügbare Platz, sodass viele Wörter ausgelassen werden. Dies lässt sich mithilfe einer Funktion darstellen, wobei die Bereiche je nach Anzahl übersprungener Wörter intensiver in roter Farbe hervorgehoben werden.

In ConcentriCloud ist zu sehen, dass für denn mittleren Ring nur wenig Platz verfügbar ist und somit nur kleinere Wörter platziert werden. Die anderen beiden Ringe haben Platz, um auch einige größere Wörter zu positionieren.

Das größte Problem dieses Ansatzes ist das Auslassen von Wörtern, wenn nicht ausreichend freier Platz zur Verfügung steht. Es ist kritisch zu bewerten Wörter auszulassen und vor allem ist es nicht klar welche Wörter übersprungen werden.

Ein positiver Aspekt ist die genaue Darstellung und Trennung der einzelnen Dokumente [LHB+15]. Zusammen mit den Schnittmengen und dem kompletten Schnittbereich werden die wichtigsten Kategorien für Multidokumenten Word-Clouds abgedeckt.

3.6. RadCloud

Die Visualisierung RadCloud [BLB+14] verfolgt einen etwas anderen Ansatz als die ConcentriCloud. Es wird zwar auch eine Kreisform als Form genommen, jedoch befinden sich am Rand nur die Dokumentnamen mit einer eindeutigen Farbe als Orientierungshilfe. Diese Positionen entsprechen dann einem Endpunkt, zu dem die Wörter innerhalb des Kreises je nach Zugehörigkeitsgrad im 2D-Raum verteilt werden. Im ersten Schritt bekommen die Wörter einen infinitesimalen kleinen Punkt, damit die originalen Positionen ermittelt werden können. Ausgehend von den nun berechneten Punkten werden die Tags gezeichnet. Es wird in Reihenfolge der Häufigkeit gezeichnet. In der Visualisierung kommen die Wörter nur einmalig vor und werden nicht wie bei ConcentriCloud unter bestimmten Konstellationen mehrfach gezeichnet.

Der Algorithmus geht dabei so vor, dass die Wörter direkt platziert werden sofern ausreichend Platz vorhanden ist. Falls es zu einer Kollisionen zwischen Tags kommt werden beide Tags verschoben. Dabei wird das bereits positionierte Wort so wenig wie möglich verschoben und das zu platzierende Wort wird in einer Spiralform von der Ausgangsposition durchlaufen. Wenn das Wort einen Platz gefunden hat, bildet es mit dem bereits platzierten Wort ein

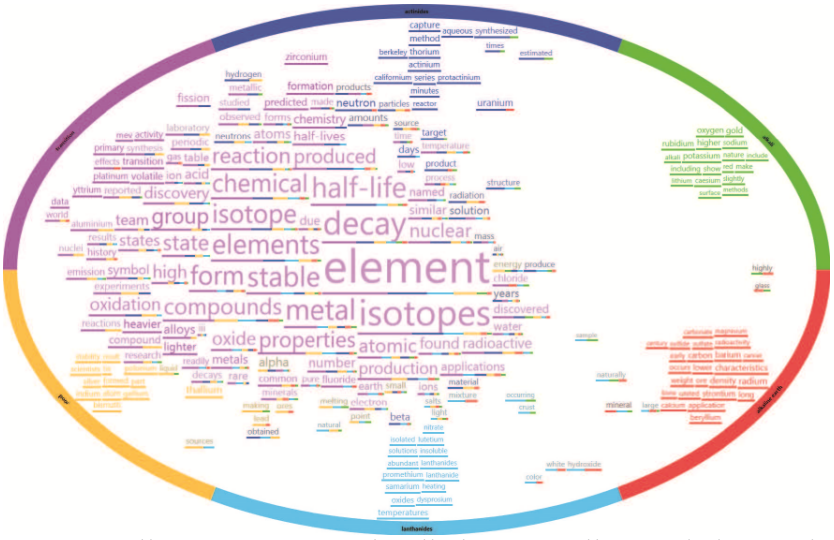


Abbildung 3.6.: Darstellung von unterschiedlichen Metallen und chemischen Elementen mithilfe der RadCloud-Visualisierung. [BLB+14]

Cluster [BLB+14]. In den nachfolgenden Schritten würde dann nicht mehr ein einzelnes Wort verschoben werden, sondern es werden gesamte Cluster bewegt, um die Positionen möglichst unverändert zu lassen. So wird die WordCloud nach und nach aufgebaut.

Durch die Verschiebungen der gesamten Cluster kommt es dazu, dass einige Tags auf weit entfernte Positionen verschoben werden. Dadurch kann der Eindruck des Tags manipuliert werden, wenn es plötzlich auf einer falschen Position landet. Dies ist einer der Nachteile dieser Visualisierungen.

Ein weiterer Nachteil ist der große verschwendete Platz durch Whitespace. Dies ist gut in RadCloud zu erkennen. Ein Teil wird gut mit Wörtern gefüllt, wobei der Rest nur spärlich besetzt bleibt. Der Algorithmus geht nicht effizient mit dem Ausnutzen des verfügbaren Platzes vor. Die Positionen der Tags werden zwar in den meisten Fällen in der Nähe ihrer Originalposition gebracht, jedoch bleibt viel Whitespace erhalten, da die Schriftgrößen beispielsweise nicht mit dem verfügbaren Platz skalieren.

Jedoch bietet die Visualisierung auch hilfreiche Konzepte. Dazu zählt die Farbgebung der Tags zu den Dokumente. Hierbei werden die Tags mit der Farbe des Dokumentes gezeichnet, zu dem die Häufigkeit maximal ist. Die restlichen Verteilungen zu den Dokumenten werden durch eine Linie unter dem Wort dargestellt [BLB+14]. Dies ermöglicht einen schnellen und leichten Vergleich zu der Verteilung des jeweiligen Wortes.

3.7. Weitere Ansätze

Die zuvor erwähnten Visualisierung sind der benötigten Aufgabenstellung am ähnlichsten und wurden daher detaillierter beschrieben. Dies war dabei nur ein kleiner Ausschnitt aus der Vielfalt an unterschiedlichen Möglichkeit eine Word-Cloud darzustellen.

Es existieren noch weitere Visualisierungen, die sich mit der Multidokumenten-Visualisierung beschäftigen. Dazu zählen [PMC+13] und [EBC+13]. Diese Visualisierung haben ihren Fokus in der Visualisierung von sehr hohen Dokumentmengen. Diese befassen sich mit 100 oder mehr Dokumenten und versuchen die Verbindungen dieser darzustellen.

Es gibt auch Cluster-Visualisierungen, die ihren Fokus auf das Layout setzen. Dazu zählt [PTT+12]. Eine speziellere Ausprägung ist dabei [OSR+14].

Die Idee zur Nutzung von weiteren Diagrammen innerhalb der Word-Cloud-Visualisierung ist auch bei [LRKC10] und [LBSW12] zu sehen, wo die Diagramme zusätzliche Informationen innerhalb eines Wortes widerspiegeln.

Zur Übersicht zu verfügbaren Word-Clouds von unterschiedlichen Kategorien bietet die Fachstudie [KLM15] einen umfassenden Überblick. Die Word-Clouds werden hinsichtlich unterschiedlichen Kriterien bewertet und eingestuft.

4. Konzept

In diesem Teil der Arbeit werden die einzelnen Ideen und Konzepte für die Multidokumenten-Visualisierung erläutert. Die Ideen wurden in der Anfangsphase aufgestellt und gesammelt. Im Laufe der Implementierung sind zudem noch weitere Ideen hinzugekommen, die hier entsprechend ausgeführt werden.

Der größte Teil befasst sich mit der Darstellung der Word-Cloud. In Anbetracht der Visualisierungen ConcentriCloud und RadCloud sollen die Nachteile entfernt werden. Daher muss das Layout der Word-Cloud entsprechend gewählt werden, damit die Nutzung des verfügbaren Platzes möglichst optimal ist. Damit würde der RadCloud Nachteil beseitigt werden.

Die Darstellung der verfügbaren Wörter konkurriert mit dem Nutzen des Platzes. In ConcentriCloud bestand das Problem mit dem Überspringen von Wörtern, sobald diese keinen Platz in dem Layout gefunden haben. Dieses muss beachtet werden, da es stets in Konkurrenz mit dem Ausnutzen des Platzes steht.

Ein Aufsetzen auf die bisherigen Implementierungen wurde bereits nach kurzen Diskussionen ausgeschlossen. Das liegt daran, dass die vorhandenen Codebasen der Arbeiten nicht geeignet sind. Das Verwenden der Codebasen würde ein intensives Einarbeiten und Studium der vorhandenen Implementierungen erfordern, um mögliche Modifikationen und Erweiterungen einzufügen.

Als weitere Hürde kommt der Unterschied in den Programmiersprachen von ConcentriCloud und RadCloud hinzu. ConcentriCloud wurde mit der Programmiersprache Java geschrieben. Hingegen wurde RadCloud in C# geschrieben. Das Umschreiben von einer Codebasis würde einen hohen zusätzlichen Aufwand erfordern.

Diese genannten Nachteile und Schwierigkeiten der vorhandenen Implementierungen hat zu der Entscheidung geführt, dass MultiCloud neu geschrieben wird.

4.1. Datenstrukturen

Rückblickend auf die Datenstruktur einer Word-Cloud besitzen die Wörter eine Gewichtung. Transformiert man diese Struktur auf eine Multidokumentdatenstruktur benötigen die Wörter die Gewichtungen oder Anzahl der Auftreten zu den Dokumenten. Da die Vorkommnisse in

den Dokumenten unterschiedlichen sind, sowie die Dokumentlängen eine Relevanz haben können, sollte dies nicht außer Acht gelassen werden. Für die Gewichtung von Auftreten mit der Dokumentlänge existieren bereits verschiedene Ansätze.

In dieser Arbeit werden nur zwei unterschiedliche Ansätze vorgestellt, da bei einer geringen Dokumentanzahl die Auswirkungen nicht so hoch sind. Die zwei Maße sind das *tf*-Maß (Termfrequenz) und das *tf-idf*-Maß (Termfrequenz und Inverse Dokumenthäufigkeit).

tf Das Maß ordnet die Frequenz eines Terms zu einem Dokument zu. Es handelt sich um die reine Vorkommenshäufigkeit des Wortes in einem Dokument. Das Maß wird wie folgt nach [HUA08] definiert:

Sei *t* ein Term und sei *d* ein Dokument.

(4.1) $\mathrm{tf}(t, d) = \#$ der Auftreten

Dies lässt sich durch Darstellung aller Terme als Vektor zusammenfassen, wobei *i* dem *i*-ten Term entspricht.

(4.2) $v = \mathrm{tf}(t, d)$, wobei $v_i = \mathrm{tf}(t_i, d)$

Als Beispiel ein Satz der zu Dokument 2 gehört.

```
Der Hund bellt, weil der Hund des Nachbars bellt.
```

In diesem Beispiel ist die Termfrequenz für bellt: $tf(bellt, 2) = 2$. Durch Berechnung aller Wörter werden so die Vektoren erzeugt.

Als weiteren Schritt lassen sich die Werte normalisieren, damit sie im Bereich von 0 bis 1 liegen.

(4.3) $$\mathrm{tf}(t_i, d)_N = \frac{\mathrm{tf}(t_i, d)}{max\left\{\mathrm{tf}(t_j, d) : \forall j\right\}}$$

Da die normalisierten Werte einen guten Ausgangspunkt zum Rechnen haben, werden sie in dieser Arbeit verwendet.

tf-idf Im Gegensatz zum *tf*-Maß, dass keinerlei Verbindungen zu den Dokumentlängen aufstellt, wird bei der inversen Dokumenthäufigkeit darauf geachtet. Dabei werden die berechneten Termfrequenzen *tf* mit der inversen Dokumenthäufigkeit *idf* multipliziert. Diese ist wie folgt definiert:

Sei N die Gesamtanzahl der Dokumente und n_i die Anzahl der Dokumente, in denen Term t_i vorkommt.

$$\text{idf}(i) = log\frac{N}{n_i} \tag{4.4}$$

Zusammenfassend mit Gleichung 4.2 und Gleichung 4.4 ergibt sich die Formel zur Berechnung des Gewichts für den Term i und Dokument j:

$$w(i,j) = \text{tf}(i,j) \cdot \text{idf}(i) \tag{4.5}$$

Da sich diese Multidokumentenvisualisierung auf eine geringe Anzahl an Dokument beschränkt, ist die Entscheidung auf das *tf*-Maß gefallen. Durch dieses Maß kommen die wichtigen Wörter mehr zur Geltung mit dem Nachteil keine Rücksicht auf die Dokumentenlängen zu nehmen.

4.2. Schnittmengen und Dokumentwörter

Ein wichtiger Teil des Konzepts ist die Darstellung der Wörter von Schnittmengen und Dokumenten. In ConcentriCloud wurden die Schnittmengen im Inneren in zwei Schichten dargestellt, wohingegen die Wörter der Dokumente nur im äußeren Ring dargestellt wurden. Jedoch wurden nicht alle Schnittmengen dargestellt, da diese nur mit benachbarten Dokumenten gebildet wurden. Das Konzept mit den durch Linien getrennten Dokumentwörtern ist jedoch eine interessante und gute Idee und stellt damit eine der Möglichkeiten dar.

Eine andere Idee ist die Darstellung aller Wörter, sowohl Schnittmengen als auch Dokumentwörter, in einer gemeinsamen Word-Cloud. So wurde es in RadCloud verwendet, wobei Dokumentwörter kaum auftraten, sondern überwiegend Schnitte. Die Visualisierung wird dadurch entlastet, da es nur eine Word-Cloud ohne Abtrennungen gibt. Jedoch fehlen die harten Grenzen, wie es bei ConcentriCloud der Fall war. Dadurch findet die Orientierung anhand von Positionen und anderen Hilfsmitteln statt, um die Zugehörigkeit eines Wortes zu ermitteln.

Aus Zeitgründen und ästhetischen Aspekten wurde die Darstellung als eine gemeinsame Word-Cloud näher in Betracht gezogen und als Startkonzept ausgewählt.

4.3. Layout

Die ersten Schritte der Arbeit bestanden mit der Konzeptionierung eines geeignet Layouts als Basis für die Implementierung. Das Layout sollte möglichst wenig Whitespace erzeugen und gleichzeitig alle Wörter darstellen können.

Als eine der frühen Ideen hat sich die Variationen in den Layoutformen geäußert. Wie in Layouts von Word-Clouds beschrieben, sind unterschiedliche Formen möglich. Da ConcentriCloud und RadCloud bisher nur in ellipsenförmigen Layouts dargestellt wurden, hat sich die Frage gestellt, ob eine wählbare Form besser wäre. Die Form der Word-Cloud sollte nicht nur auf eine Form beschränkt sein. Als Layoutformen sollten vorerst eine Rechtecks- und Kreisform wählbar sein.

In den nachfolgenden Ideen werden diese dargestellt, wobei mit Skizzen die Punkte diskutiert und analysiert wurden.

4.3.1. Kräftebasiertes Layout

Bei diesem Ansatz wird die Word-Cloud mithilfe eines kräftebasierten Layouts erstellt. Das Gerüst besteht aus Knoten und Kanten, wobei sich Knoten abstoßen und Kanten anziehen. Eine genauere Beschreibung ist unter Kräftebasiertes Layout zu finden.

Die Dokumente werden als Fixpunkte am äußeren Bereich des Layouts angeordnet. Da unterschiedliche Layoutformen unterstützt werden sollen, müssen die Fixpunkte je nach Form unterschiedlich positioniert werden. Außerdem sollten die Abstände zwischen den Dokumenten möglichst einheitlich sein, damit die Verteilung durch die Kräfte des Layouts nicht verfälscht wird.

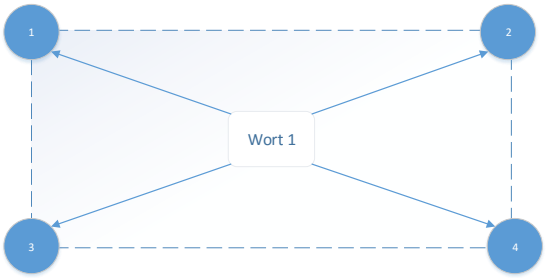


Abbildung 4.1.: Konzept des Layouts mit vier Dokumenten, die am rechteckigen Rand angeordnet sind. Das Wort hat eine gleichverteilte Gewichtung zu allen Dokumenten.

Da die Dokumente nun eine Position haben und fixiert sind, werden die Kanten des Layouts definiert. Durch die zuvor definierte Struktur besitzen die Wörter eine Gewichtung zu jedem

Dokument. Für jede Gewichtung > 0 wird nun eine Kante zu dem jeweiligen Dokument erzeugt. Durch die Kräfte und Kanten verteilen sich die Wörter in der vorgesehenen Fläche.

Eine Beispieldarstellung ist in Abbildung 4.1 zu sehen. Die blauen Kreise entsprechen dabei den Dokumentfixpunkten. Der Rahmen mit den abgesetzten Strichen entspricht dem Layout. Wie zuvor beschrieben werden die Dokumente am Rand des Layouts, in diesem Beispiel ein Rechteck, verteilt. Zur Vereinfachung wurden die Dokumente auf die Eckpunkte des Rechtecks gesetzt. Die Pfeile entsprechen den Kanten. Ist die Gewichtsverteilung gleich, so liegt das Wort im Zentrum des Layouts.

Liegen die Gewichtungen nicht gleichverteilt vor, so werden die Wörter durch das kräftebasierte Layout an die korrekte Position gedrückt. Sollte ein Wort nicht in einem Dokument auftauchen und das Gewicht 0 haben, so wird die Kante nicht hinzugefügt. Wie in Abbildung 4.2 zu sehen, besitzt das Wort in dem Fall nur noch drei anstatt vier Kanten.

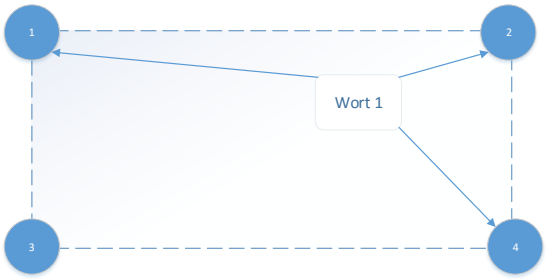


Abbildung 4.2.: In diesem Beispiel besitzt das Wort nur noch drei Gewichtungen, da die Gewichtung zu Dokument 3 den Wert 0 hat.

Im Falle eines Dokumentwortes, das nur eine Bindung hat, wird dieses direkt in die Dokumentnähe gezogen. Dies ist in Abbildung 4.4 zu sehen.

Abbildung 4.3.: Ein Dokumentwort das nur eine Kante besitzt.

Sollten die Anzahl der Tags nicht ausreichen, um die vorhandene Fläche ausreichend auszufüllen, so müssen die Kräfte des Layouts variiert werden, damit sich die wenigen Wörter optimal verteilen. Dies wirkt ästhetisch ansprechender, anstatt einer gehäuften Word-Cloud

mit viel Whitespace. Dies würde zudem die Anforderung dieser Arbeit, mit dem optimalen Ausnutzen des freien Raumes, verletzen. Eine daraus resultierende Veränderung der Positionen darf keine starken Änderungen aufweisen, damit die Zuordnung erhalten bleibt. Hier würde sich eine Skalierung der Schriftgrößen anbieten, um den vorhandenen Platz besser zu füllen siehe Abbildung 4.4.

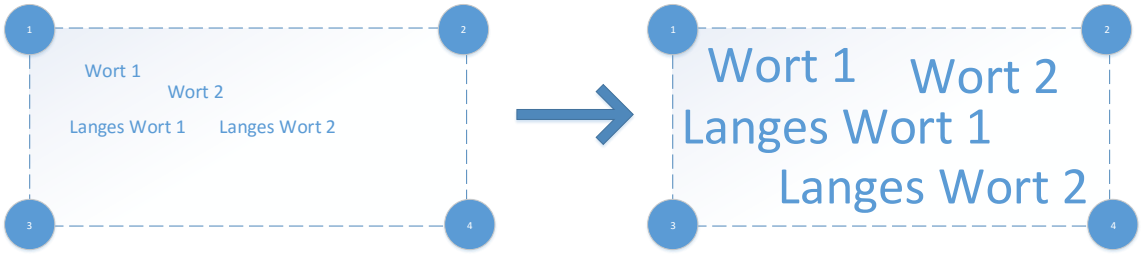


Abbildung 4.4.: Beispiel für eine mögliche dünnbesetzte Word-Cloud. Durch Skalierung der Wörter entsteht ein ästhetisch besserer Eindruck.

So wurde das Konzept für die Visualisierung mittels eines kräftebasierten Layouts definiert. Dabei könnten auch Probleme auftreten, die im Kapitel 5 erläutert werden und eine Lösung für das Problem gefunden wird. Unter die Probleme fallen die Knotenüberlappungen und eine Abstoßung der Knoten außerhalb des verfügbaren Rahmens.

4.3.2. Kräftebasiertes Layout mit Richtung

Dies ist eine Modifikation des zuvor beschriebenen Ansatzes. Es basiert ebenfalls auf dem kräftebasierten Layout, das mit Hinblick auf das Einfügen von Wörtern variiert werden. Bei dem normalen Ansatz wird davon ausgegangen, dass alle Wörter von Beginn an im Layout integriert sind. Durch Diskussionen haben sich mögliche Probleme mit den Positionen der Wörter herausgestellt. Aufgrund einer schlechten Verteilung könnte die Originalposition der Tags von ihrer aktuellen Position stark abweichen. Dies liegt an den Abstoßungskräften der Knoten.

Die Idee von diesem Ansatz besteht darin, dass die Knoten schrittweise aus einer festgelegten Richtung außerhalb des Rahmens eingefügt werden. Dies soll verhindern, dass bestehende Knoten in ihrer bisherigen Position stark verändert werden. Bereits eingefügte Knoten werden durch die Kräfte nicht mehr an falsche Positionen gedrückt. In Abbildung 4.5 wurde diese Idee dargestellt.

Über die Optimierung in einer tatsächlichen Umsetzung kann keine Prognose getroffen werden, sodass diese Idee nicht weiter verfolgt wurde. Die Idee würde zusätzlichen Aufwand benötigen, um das Hinzufügen der Wörter aus einer Richtung einzubauen.

Abbildung 4.5.: Ein Beispiel für die Einfügerichtung der neuen Wörter zum vorhandenen Layout.

4.3.3. Initiallayout mit Kollisionserkennung

Dieser Ansatz unterscheidet sich zu den zwei vorgestellten Ansätzen und basiert nicht auf einem kräftebasierten Layout. In diesem Ansatz verfolgt man das Konzept der Kollisionserkennung.

Die Wörter werden dabei auf der korrekt berechneten Position auf der verfügbaren Fläche gezeichnet. Dabei kommt es in den meisten Fällen zu Überlappungen der Wörter. Anstatt der Verwendung von Kräften zur Beseitigung der Überlappung wird mithilfe von Kollisionsdetektionen und darauf basierenden Verschiebungen versucht die Überlappung zu entfernen.

Eine optimale Verteilung der Knoten mit gleichzeitiger Berücksichtigung der Knotenüberlappung ist jedoch kein triviales Problem. Es gibt unterschiedliche Methoden, die sich mit diesem Problem befassen. [DMS06]

Die Implementierung eines Prototyps zum Testen ist aufgrund der hohen Komplexität dieser Algorithmen nur schwer zu bewerkstelligen und wurde aus diesem Grund nicht gestartet. In Betrachtung einer schneller Laufzeit und Beachtung der Ausnutzung des verfügbaren Platzes könnte ein solcher Algorithmus jedoch gute Ergebnisse liefern.

4.3.4. Münzschieber Automat

Diese Idee hat sich mit dem Gedanken an einen Münzschieber Automaten orientiert bei dem Münzen eingeworfen werden und diese durch regelmäßige Stöße auf dem verfügbaren Platz verteilt werden (siehe Abbildung 4.6). Ist der verfügbare Platz für die Münzen aufgebracht, werden die außen liegenden Münzen ausgestoßen.

Mit diesem Verhalten kam die Idee auf, dass dadurch eine gute Verteilung der Tags erzeugt wird und eine nahezu optimale Whitespace-Aufteilung entstehen könnte. Die Wörter würden sich dabei ähnlich zu den Münzen ständig gegenseitig stoßen und in freie Lücken gedrückt werden, um keine Stapelungen oder Überlappungen zu erzeugen.

Ein Problem bei diesem Ansatz ist die ungewollte Verschiebung in zufällige Richtungen. Das Grundkonzept sieht dabei keine Begrenzungen der Schieberichtungen vor und dadurch könnte in ungünstigen Fällen ein Wort an falsche Positionen verschoben werden. Dadurch würde eine Grundanforderung verletzt werden und dieser Ansatz ist daher nicht akzeptabel.

Ein weiteres Problem ist das zuvor erwähnte Ausstoßen aus dem verfügbaren Rahmen. Bei dem Automaten ist dieses Verhalten das Grundkonzept, um Besucher anzulocken. Im Bereich der Word-Cloud ist dies kein gewünschtes Verhalten. Wörter sollen nicht weggelassen werden, falls kein Platz mehr da ist. Da dies nicht jedoch kaum anders zu lösen ist, dürfen Wörter nicht zufällig ausgelassen werden.

Zu den vorhandenen Problemen stellen sich noch weitere Fragen. Wo müssten die Wörter „eingeworfen“ werden, damit diese an die korrekten Positionen gedrückt werden? Oder wie ist die Performanz eines solchen Verfahrens und wann würde der Algorithmus enden?

All diese Fragen haben die Entwicklung eines Prototypen in den Hintergrund gerückt, da das Konzept ein hohes Risiko für unzuverlässige Resultate bietet.

Abbildung 4.6.: Als Ausgangsbasis dient ein Münzschieber Automat. [1]

4.3.5. Entscheidung

Nachdem die Konzepte diskutieren waren, begann die Implementierung von einigen Testprototypen, um die engeren Ansätze auszuprobieren. Unter diese Konzepte sind das kräftebasierte Layout, das kräftebasierte Layout mit Richtung und die Kollisionserkennung gefallen.

[1] http://eigenbaukombinat.de/wp-content/uploads/2013/02/IMG_67946.jpg

Wobei die Kollisionserkennung nur auf einer einfachen Implementierung gehalten wurde, um den Rahmen der Prototypsphase zu überziehen. Außerdem wurde eine versucht eine Standardimplementierung eines Wordle-Algorithmus so zu modifizieren, dass sie für die Anforderungen passen würde. Die Implementierung hat dabei auf die D3.js [2] basiert.[3]

Kräftebasiertes Layout Da es sich bei diesem Prototyp um ein kräftebasiertes Layout ohne Zusatzfunktionen handelt, war die Erstellung eines Prototyps schnell zu bewerkstelligen. Die Ergebnisse des Prototyps waren anschaulich und kamen bereits in die Vorstellung. Einige Probleme sind im Prototypen enthalten gewesen. Diese umfassten die Knotenüberlappungen und Knotenverteilung.

Insgesamt bot der Prototyp bereits einen guten und anschaulichen Ausgangspunkt für die Arbeit.

Kollisionserkennung Die Implementierung mit der Kollisionserkennung erwies sich als kompliziert. Da Algorithmen wie von [DMS06] beschrieben, das Problem der Knotenüberlappungen in einer akzeptablen Laufzeit lösen können, jedoch einen komplizierten Algorithmus als Basis verwenden, wurde ein simplerer Algorithmus verwendet. Die Umsetzung des aufwändigen Algorithmus würde die verfügbare Zeit in der Prototyppphase weit überschreiten, daher wurde darauf verzichtet.

Als Algorithmus wurde hierbei auf die simple Kollisionserkennung gesetzt. Das Konzept ist weit in der Spieleentwicklung verbreitet und es wurde die Kollisionserkennung für 2-dimensionale Rechtecke verwendet. Die Resultate des Prototyps waren nur sehr beschränkt nutzbar, da der verfügbare Platz und der Whitespace kaum beachtet wurde. Das liegt vor allem an dem einfachen Algorithmus, der keine zusätzlichen Parameter prüft und nur auf Kollisionen prüft und die Wörter so verschiebt, dass die Kollision beseitigt wird.

Wordle Modifikation Wie der Wordle Algorithmus funktioniert wird in Abschnitt 3.1 beschrieben. Die Wörter werden spiralförmig ausgehend aus dem Zentrum der verfügbaren Fläche positioniert.

Um den Algorithmus als Prototyp zu verwenden, ist eine verfügbare freie Implementierung notwendig. Es hat sich schnell ein Algorithmus gefunden, der auf dem Wordle Algorithmus basiert und zudem auf D3.js basiert. Diese Implementierung wurde als Vorlage für diesen Ansatz gewählt, um einen möglichen Prototypen zu generieren.

Die Änderung der Imlementierung bestand darin, dass eine definierte Startposition für die einzelnen Wörter gesetzt wird mit dem Gedanken, dass sich die Wörter dadurch im Idealfall nicht weit von dieser Position entfernen.

[2] http://d3js.org/
[3] https://github.com/jasondavies/d3-cloud

Das Ergebnis der kleinen Modifikationen des Algorithmus lieferte jedoch keine nützlichen Resultate, da die Wörter weiterhin randomisiert verteilt wurden und nur kleine Änderungen bezüglich der Position aufwiesen.

Verifiziert wurde die Position indem ein Pfeil zwischen die Originalposition und tatsächlichen Position gelegt wurde. Je länger die Pfeile wurden desto weiter liegen die Wörter von ihrer eigentlichen Position weg. Dies sollte soweit reduziert werden, wie es nur möglich ist. Auch viele Überkreuzungen von Pfeilen würden auf mögliche Fehler hindeuten. Eine genauere Beschreibung mithilfe dieser Verifikation ist in Kapitel 5 zu finden.

Nach der Prototypphase hat sich das kräftebasierte Layout als Favorit hervorgehoben. Mit diesem Layoutkonzept wurde die Arbeit umgesetzt und wird in Kapitel 5 genauer erläutert mit den entstanden Problemen und Lösungen.

4.4. Visuelle Darstellung der Wörter

In Kapitel 2 wurden bereits unterschiedliche Aspekte der visuellen Variationen erläutert. In dieser Arbeit wurden bewusst die nützlichsten Attribute ausgewählt. Darunter fallen:

- Schriftgröße als Relevanzdarstellung
- Keine Rotation der Wörter
- Schriftfarbe entsprechend der Dokumentfarben. Als Farbe wird dabei die Farbe des Dokuments genommen zu dem das Wort die höchste Bindung hat.
- Farbsättigung je nach Bindungsstärke zu den Dokumenten. Dies bedeutet, dass die Farbe grauer wird umso niedriger die höchste Bindung hat. Dabei soll das Wort bereits komplett grau sein, sobald die höchste Bindung unter 50% fällt.

Mit den definierten Richtlinien bleiben die Wörter gut lesbar und lassen sich leicht zuordnen, sobald ein Wort eine starke Bindung zu einem Dokument aufweist. Die Reduzierung der Sättigung soll einer automatischen Zuordnung entgegen wirken, damit keine falschen Zuordnung entstehen und kein Visual Lie hervorgerufen wird. Damit hergehend kommt auch die Beachtung einer geeigneten Schriftskalierung. Wie in Abschnitt 2.1 erläutert, erzeugt eine falsch gewählte Skalierung Visual Lie und verfälscht somit die dargestellten Informationen.

4.5. Server-Client-Konzept

Die Auswahl zwischen einer Anwendung, die auf dem Computer des Nutzers ausgeführt werden muss und einer Anwendung, die mithilfe des Browsers aufgerufen werden soll, war zügig getroffen. Dies hängt damit zusammen, dass viele Anwendung bereits heute gezielt auf webbasierte Anwendungen und Apps fokussieren, damit die Anwendung dem Nutzer so komfortabel wie möglich nahe gebracht wird. Dazu kommen auch die regelmäßigen Weiterentwicklungen der Browser und mit ständig steigender Performanz. Die Umsetzung von WebGL ermöglicht es grafisch und rechnerisch intensive Arbeiten auf die Grafikeinheit des Computers auszulagern. Selbst 3D-Visualisierung und weitere anspruchsvolle Aufgaben, die viel Rechenleistung benötigen, lassen sich direkt im Browser realisieren. Da eine Grafikeinheit viele Rechenkerne besitzt, kann viel Parallelität durch Verteilung der Berechnung erzeugt werden.

Jedoch bringt die Auslagerung auf den Browser auch Nachteile mit sich. Durch die vielen unterschiedlichen Browser kann eine umfassende Kompatibilität nicht garantiert werden. Die Anwendungen werden auf einer Auswahl an Browsern getestet, meistens an den populärsten, und die nicht getesteten werden dabei außer Acht gelassen. In den meisten Fällen werden mit heutigen Stand die Kompatibilität in *Mozilla Firefox*[4] und *Google Chrome*[5] gewährleistet. Der *Internet Explorer*[6] hat ständig mit Problemen zu kämpfen, da viele Konzepte teilweise schlecht oder auch gar nicht umgesetzt werden. Vor allem grafisch basierte Anwendungsgebiete wie zum Beispiel *SVG (Scalable Vector Graphics)*[7] werden im Internet Explorer fehlerhaft dargestellt.

Nicht nur die unterschiedlich Browser können Kompatibilitätsprobleme auslösen. Die Nutzung des verwendeten Betriebssystem kann ebenso Auswirkungen haben. Zusammen mit der Kombination aus Browsern können dadurch viele unvorhersehbare Probleme auftreten. Linux, Windows oder MacOS besitzen unterschiedlich Integrationen der Browser. Im Idealfall würde man die Webanwendung in unterschiedlichen Kombinationen aus Betriebssystem und Browser testen, aber dies ist sehr unwahrscheinlich.

Durch die vorhandenen Toolkits im Web-Bereich, werden Browser stärker unterstützt. Das auf JavaScript basierende Visualisierungskit D3.js bietet performante und konfigurierbare Visualisierungsmöglichkeiten an. Es bietet für fast alle Anwendungsfälle Visualisierungen an von simplen Diagrammen wie Liniendiagrammen oder Balkendiagrammen bis hin zu kräftebasierten Layouts, Parallel Coordinates oder auch 3D-Visualisierungen. Da das Layout auf einem kräftebasiertes Layout aufbauen soll, bietet sich D3.js durch die vorhandene Unterstützung gut an. Zudem ist durch bereits erlangte Erfahrungen mit D3.js und besuchten

[4] https://www.mozilla.org/de/firefox/new/
[5] https://www.google.com/chrome/browser/desktop/index.html
[6] http://windows.microsoft.com/en-us/internet-explorer/download-ie
[7] http://www.w3.org/TR/SVG/

Seminaren zu Visualisierungstoolkits, sowohl im Clientbereich als auch Webbereich die Tendenz zu D3.js gezeigt worden.

Ein weiterer Erfahrungsbereich ist durch das Studienprojekt *ViTA*[8] dazugekommen. Es wurde ebenfalls ein Server-Client-Konzept genutzt, um die Anwendung über den Browser aufrufen zu können. Die Interaktion mit dem Server wurde dabei nur über den Browser geführt.

4.6. Dokumentverarbeitung

Die Dokumentverarbeitung soll für den Anfang der Umsetzung einfach und performant gehalten werden. Es sollen keine aufwändigen Berechnungen oder Verarbeitungen durchgeführt werden. Es genügt, dass die *tf* der Wörter berechnet wird, um eine Gewichtung zu erhalten, die für die Word-Clouds genutzt werden kann.

Für die fortgeschrittene Verarbeitung der Dokumente kann NLP verwendet werden. Dies hat jedoch für die ersten Versionen der Word-Cloud keine hohe Relevanz. Durch die Verarbeitung der Sprachformen wird die Filterung nach Wortarten wie Nomen ermöglicht. Für viele Fälle der Dokumentanalyse haben Nomen die primäre Bedeutung, da Verben oder Adjektive alleinstehend keine Aussage über den Text treffen. Im Gegensatz bieten Wörter wie „Krieg“ oder „Streit“ konkrete Anhaltspunkte über den Inhalt und Bedeutung der Texte. Der Nutzen der Word-Cloud würde dadurch enorm gesteigert werden.

4.7. Nutzerempfinden und Bedienung

Die Anforderung an die Visualisierung und Anwendung soll verständlich, leicht bedienbar sein und stets visuell ansprechende Ergebnisse erzeugen. Die Bedienung und Manipulation der Visualisierung soll einfach sein. Ein Problem, dass die RadCloud-Visualisierung hat, ist die willkürliche Generierung von gut aussehenden Word-Clouds. Nur durch eine geschickte Wahl an Dokumenten lässt sich in RadCloud eine verwertbare Visualisierung generieren.

Dies soll in dieser Arbeit stark optimiert werden, sodass bei beliebiger Auswahl an Dokumenten die generierte Word-Cloud ansehnlich und nützlich ist. Das umfasst zum einen die optimale Ausnutzung des Whitespaces. Zum anderen beinhaltet dies eine optimale Verteilung der Wörter im verfügbaren Raum, sowie eine Skalierung der Wörter bei geringer Anzahl an Wörtern.

[8] https://github.com/vita-us/ViTA

4.8. Interaktion

Der Überblick über die Word-Cloud kann schnell verloren gehen, wenn zu viele Daten oder visuelle Elemente auf einmal dargestellt werden. Dieses Problem kann mithilfe von einer geeigneten Interaktionsform gelöst werden, sodass nicht die gesamte verfügbare Informationen auf einmal präsentiert wird.

Als Interaktionsformen bieten sich Hovering, Selektion, Zoomen, Filterung und Hervorhebung an.

Hover Das Hovern beschreibt die Interaktion, wenn der Nutzer den Mauszeiger über ein Element positioniert lässt. Dies ist ein übliches Mittel um Popups hervorzuholen, die zusätzliche Informationen zu dem gesuchten Element zeigen. Beispielsweise werden im Browser beim Hovern der Tabs die Titel in dem Popup nochmals angezeigt.

Für diese Arbeit bietet es sich an ebenfalls Popup-Elemente für die Wörter zu integrieren. In diesem Popup können zusätzliche Informationen zum Wort wie zum Beispiel: eine Dokumentverteilung, Auftreten oder Schriftgröße angezeigt werden.

Selektion Mithilfe der Selektion lassen sich zusätzliche Informationen darstellen. Wenn ein Wort selektiert wird, könnten Details zu dem Wort in einem seitlichen Element dargestellt werden. Darin könnten NLP Informationen oder genaue Positionen oder Vorkommnisse in den einzelnen Dokumenten präsentiert werden.

Eine weitere Möglichkeit, die die Selektion bietet, ist die mehrfache Auswahl an Wörtern oder Dokumenten. Durch Auswahl von mehreren Wörtern lassen sich Gemeinsamkeiten oder Unterschiede dieser darstellen. Wenn jedoch ein Dokument selektiert werden, können nur die am stärksten gebunden Wörter (>50%) hervorgehoben werden. Falls mehrere Dokumente selektiert werden, würde dies die Wörter hervorheben, die im Schnitt dieser Dokumente liegen.

Zoomen Das Zoomen ist eine verbreite Interaktionsmethode und wird häufig zusammen mit dem Panning (Schwenken) verwendet. Panning bedeutet, dass die Fläche der Visualisierung verschoben werden kann. Mit dem Zoomen wird eine bessere Exploration der Word-Cloud ermöglicht. Wörter mit geringer Schriftgröße können dadurch vergrößert werden, falls die Interesse besteht. Ebenso lassen sich damit auch nur Bereiche in den Fokus nehmen, an denen ein Interesse vorhanden ist. Wenn nur Wörter von einem Dokument gewünscht sind, kann man diese dadurch in den Fokus ziehen.

Filterung Die Methode Filterung ist für das temporäre Entfernen von gewünschten Wörtern aus der Visualisierung. Dadurch wird es ermöglicht Wörter nach den eigenen Interessen zu entfernen beziehungsweise anzuzeigen. Sollen nur Wörter von einer bestimmten Dokumentmenge angezeigt werden, so lassen sich diese filtern. Eine anderes Einsatzgebiet wäre die Kombination mit Annotationen aus einem NLP-Modul. Durch

die Annotationen kann man Wörter nach Wortarten oder Wortstämmen filtern. Der Nutzen von NLP wäre damit flexibel durch den Nutzer variierbar.

Hervorhebung Bei einer hohen Anzahl an Wörtern ist eine Unterscheidung schwierig. Vor allem, wenn es sich um kleine Wörter handelt. Über welchem Wort schwebt die Maus zurzeit? Diese Frage kann mithilfe der Hervorhebung eines Wortes gelöst werden. Das Wort über dem der Mauszeiger schwebt, wird in dieser Zeit mit einer auffallenden Schriftfarbe dargestellt. Dadurch weiß der Benutzer stets, welches Wort er gerade fokussiert.

Durch Kombination mehrerer Interaktionsformen kann der Nutzer stark beim Untersuchen der Word-Cloud unterstützt werden. Es wird vermieden, dass der Nutzer das Gefühl bekommt den Überblick zu verlieren.

4.9. Benutzeroberfläche

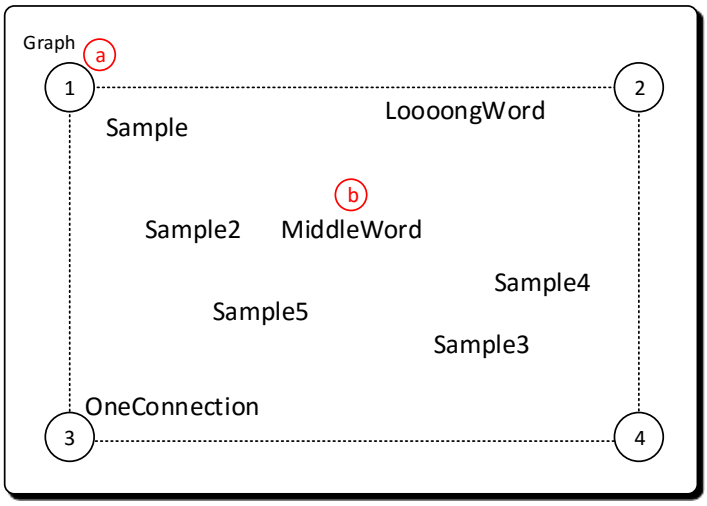


Abbildung 4.7.: Skizzierte Darstellung des Graphens innerhalb eines Containers. An den Rändern sind die Dokumente positioniert und zwischen den Dokumenten werden die Wörter im Raum verteilt.

Teile der Benutzeroberfläche sind bereits in der Anfangsphase entstanden und wurden mithilfe von Skizzen festgehalten. Der Fokus lag dabei auf der Darstellungen des Graphen zusammen mit der Interaktionsmöglichkeiten.

Der Graph soll als eigene Einheit dargestellt werden. Dafür bietet sich ein Container an, in dem der Graph gelegt wird. In Abbildung 4.7 wurde die Form als Skizze dargestellt.

Die Dokumente werden im äußeren Bereich angeordnet und je nach Form verteilt (siehe Abbildung 4.7 a)). Die abgesetzte Linie grenzt dabei den verfügbaren Platz für die Wörter ein. Innerhalb dieser Linie befindet sich die Zeichenfläche in der die Wörter gezeichnet werden (siehe Abbildung 4.7 b)).

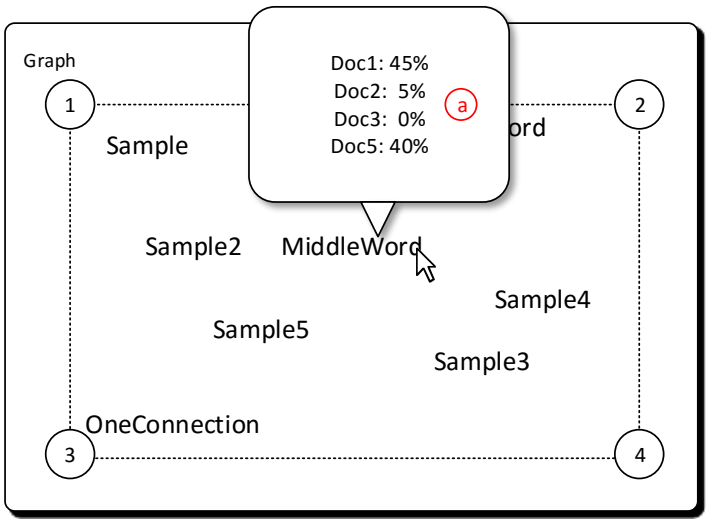


Abbildung 4.8.: Skizze eines Popups, das sich öffnet, sobald der Mauszeiger über einem Wort schwebt.

Als visuelles Element der Interaktionsmöglichkeiten kommt das Popup hinzu. In diesem sollen weitere Informationen bezüglich des markierten Wortes auftauchen. Das Popup soll angezeigt werden, sobald der Mauszeiger über ein Wort fährt und einen kurzen Moment auf diesem verbleibt.

Wie in Abbildung 4.8 a) zu sehen, können beispielsweise die Verteilungen des markierten Wortes angezeigt werden. Es soll möglichst generisch und beliebig gestaltbar sein, daher sollte der Inhalt durch HTML und CSS definiert werden. Durch den generischen Ansatz soll es möglich sein, Diagramme und komplexere Elemente als reinen Text darzustellen.

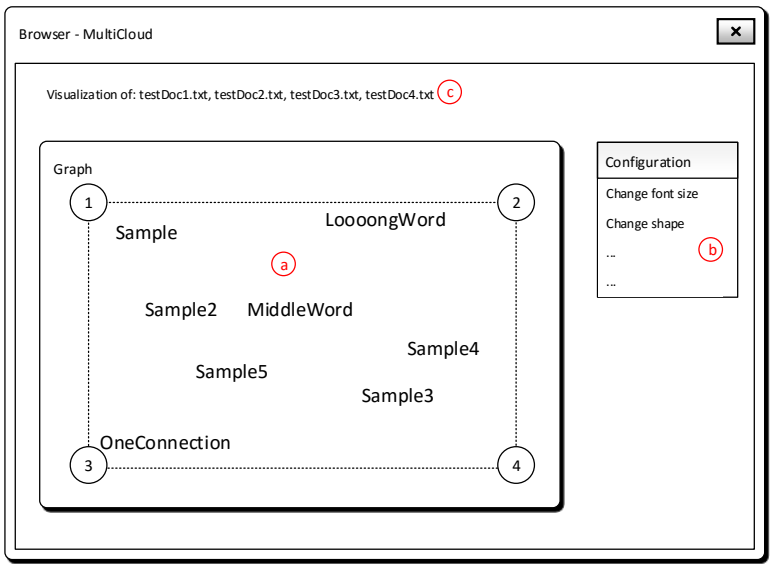


Abbildung 4.9.: Die gesamte Skizze der Anwendung mit dem Graphen auf der linken Seite, der den größten Platz einnimmt. Über dem Graph eine Auflistung der Dokumente und rechts eine Interaktionsliste.

Zusammengefügt ergeben die Skizzen innerhalb des Browsers die Abbildung 4.9. Der Graph liegt dabei links im Fenster (siehe a)). Oberhalb des Graphs befindet sich die Darstellung der dargestellten Dokumente (siehe b)). Dadurch sollen sich die verwendeten Dokumente leicht identifizieren lassen, damit diese nicht in Vergessenheit geraten. Rechts im Browser befindet sich ein weiterer Interaktionsbereich, in dem der Graph manipuliert werden kann. Die Elemente sind variabel. Dies bedeutet, dass es Knöpfe, Checkkästchen oder weitere Elemente sein können. Auch hier spiegelt sich das generische Konzept wieder.

5. Implementierung

In diesem Teil der Arbeit wird die Umsetzung und Implementierung beschrieben. Die Implementierung wurde dabei in zwei Bereiche aufgeteilt: der Server-Implementierung und der Client-Implementierung. Wie im Abschnitt 4.5 beschrieben, handelt es sich beim Client dabei um eine Website-Implementierung. Bei der Implementierung wurden unterschiedliche Möglichkeiten getestet, von denen einige im Laufe der Arbeit verworfen wurden. Die einzelnen Kernpunkte der Implementierung werden dabei genauer erläutert.

5.1. Übersicht

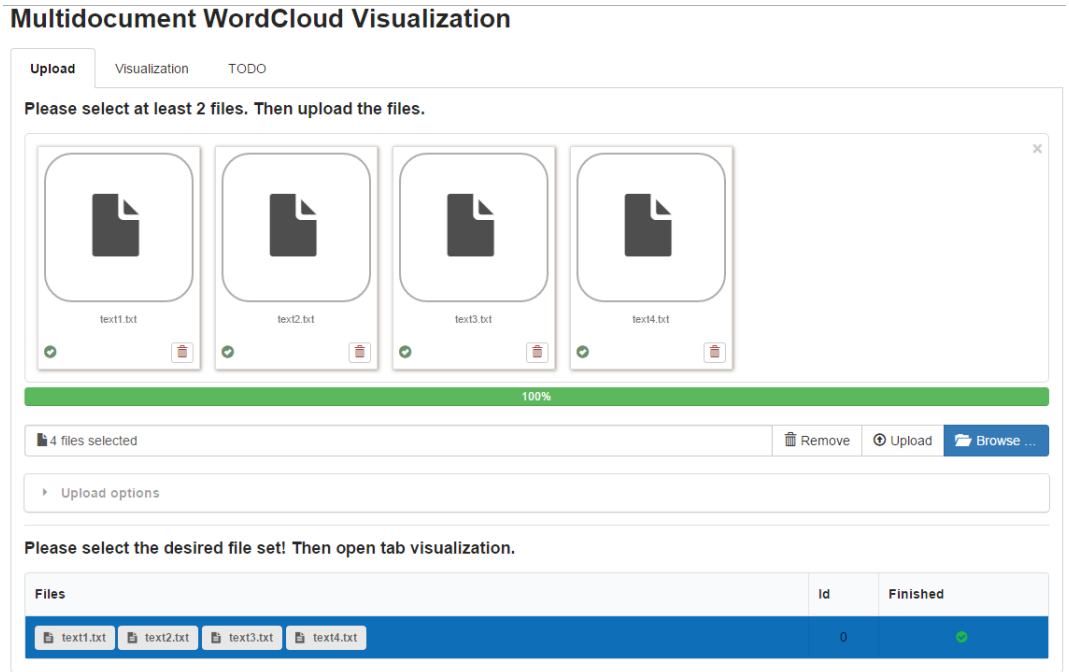


Abbildung 5.1.: Uploadfenster für den Graphen.

In Abbildung 5.2 ist das Hauptfenster der Anwendung zu sehen. In diesem wird der Graph dargestellt.

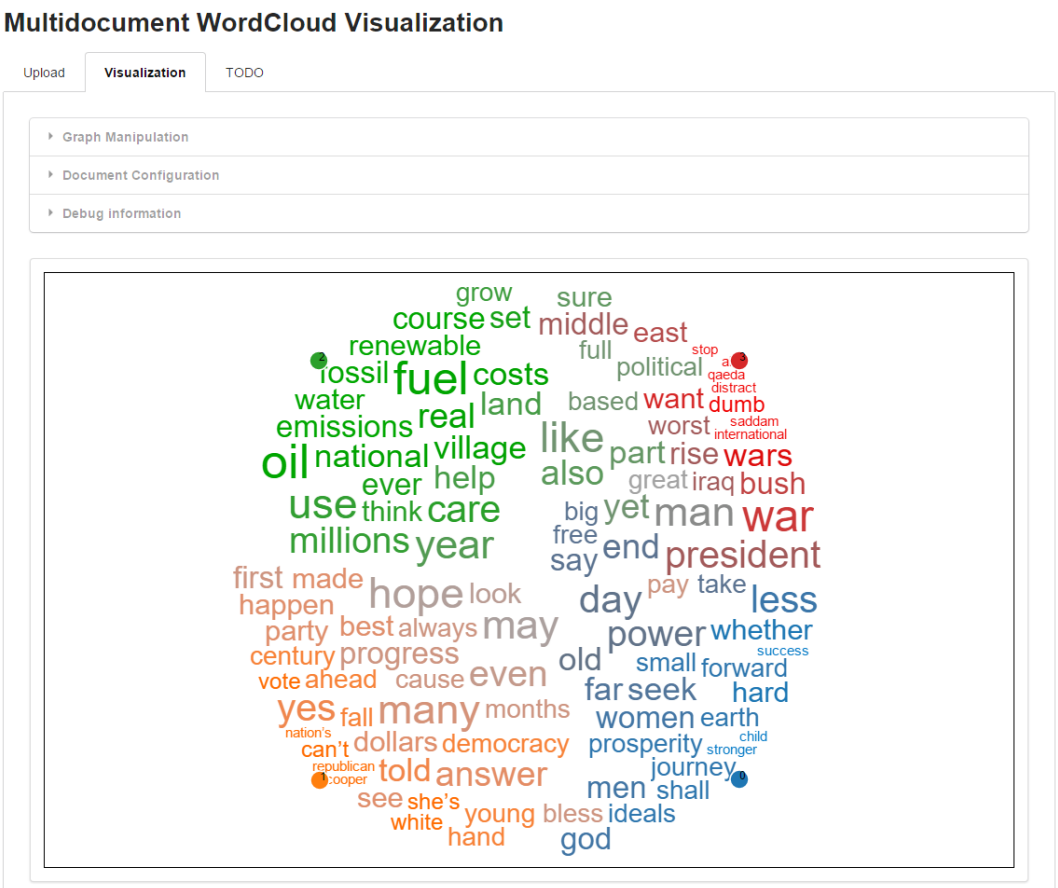


Abbildung 5.2.: Übersicht über die umgesetzte Anwendung. Im Zentrum befindet sich die Visualisierung der Word-Cloud. Über der Visualisierung befindet sich ein Akkordeonmenü in denen sich unterschiedliche Einstellungen verbergen, mit denen die Visualisierung modifiziert und erweitert werden kann.

Die Anwendung ist dabei in zwei Bereiche aufgeteilt: In einen Uploadbereich und den Graphbereich. Getrennt sind diese durch Tabs wie unter Abbildung 5.2 zu sehen. Diese können beliebig umgeschaltet werden, falls ein neuer Korpus gewählt wird oder ein neuer Korpus hochgeladen wird. Im Graphbereich wird dabei der selektierte Korpus genutzt, um den Graphen zu generieren. Sobald ein Korpus gewählt und in den Graphbereich gewechselt wird, ist keine Form zu erkennen, da zu diesem Zeitpunkt nur ein zentriertes Kräftelayout aktiv ist. Um eine Form zu generieren kann direkt auf den Knopf „Schnelle Form generieren“ geklickt werden. Hier wird die Form mit den Standardeinstellungen generiert. Über das Menü „Graphmanipulation“ können die Grapheigenschaften direkt geändert und mithilfe des Buttons „Graph generieren“ der Graph generiert werden.

Die genaue Beschreibung der einzelnen Funktionen und Parameter wird in den nachfolgenden Kapiteln erläutert.

5.1.1. Begrenzung

Durch das Layout ergibt sich bereits eine Beschränkung der Visualisierung. Durch die Verteilung der Dokumente am äußeren Rand kann eine hohe Anzahl an Dokumenten nicht mehr übersichtlich dargestellt werden. Auch mithilfe von verschiedenen Farben für die Dokumente kann dies nicht kompensiert werden. Der Mensch kann nur eine geringe Anzahl an Farben klar unterscheiden [Hea96]. Liegen die Farben in einem zu ähnlichen Farbbereich fällt das unterscheiden schwer oder ist gar nicht mehr möglich. Durch die genannten Punkte ergibt sich eine Begrenzung hinsichtlich der unterstützten Dokumentanzahl.

Bis zu einer Anzahl an zehn Dokument ist noch eine gute Analyse der Wörter und Strukturen möglich. Unterschiedliche Farben für zehn Dokumente zu finden, die einen akzeptablen Kontrast aufweisen, ist ebenso möglich. Als obere Grenze werden 20 Dokumente empfohlen, wobei bereits hier eine Differenzierung sehr stark erschwert wird und eine genaue Analyse nur schwer möglich ist.

Da es sich um eine Multidokumentenvisualisierung handelt, kann nicht nur ein Dokument für die Visualisierung hochgeladen werden. Um eine Visualisierung generieren zu lassen, muss ein Korpus mit mindestens zwei Dokument hochgeladen werden.

5.2. Server-Architektur

Die Implementierung des Servers ist für die einfache Tokenisierung kompakt und leichtgewichtig gehalten. Die Aufgabe des Servers besteht darin, mehrere Dokumente einzulesen und die Wörter zu verarbeiten. Für diese Aufgabe kann dabei ein eigener Algorithmus geschrieben werden, der die Wörter je nach Satzzeichen trennt und so die Wörter generiert und anschließend zählt. Dabei muss jedoch auf Wortkonstellationen geachtet werden, die mit Zeichen wie zum Beispiel „-“ getrennt sind. Diese sollten nicht getrennt werden, sondern als ein Wort erkannt werden.

Die Serverimplementierung ist in der Programmiersprache Java geschrieben. Es wurde die Java Version JDK8 genutzt. Dies ist auch die einzige Version, die zur Zeit der Implementierung noch Support erhält. Mit Support ist die Auslieferung von sicherheitskritischen Updates gemeint. Die Vorgängerversion JDK 7 erhält für Privatanwender keinen Support (Supportende April 2015[1]). Mit Java kann ein plattformunabhängiges Programm geschrieben werden. Da es eine eigene Laufzeitumgebung integriert hat, macht es kaum Unterschiede, ob es auf Windows, Linux oder Mac OS ausgeführt wird.

[1] http://www.oracle.com/technetwork/java/eol-135779.html

Als Build-Management-Tool wird *Apache Maven*[2] verwendet. Ein solches Tool verwaltet die verwendeten Bibliotheken, Buildversionen und viele weitere Konfigurationen. Dadurch ist es möglich schnell und sauber neue Bibliotheken einzubinden, die automatisch heruntergeladen und direkt verwendet werden können. Zusätzlich lässt sich bei Bedarf die Dokumentation zu einer Bibliothek herunterladen, um die Methoden des Codes besser zu verstehen. Ein weiterer wichtiger Punkt ist die Unterstützung eines Standard-Lebenszyklus von Software. Darunter fallen die Validierung, Kompilierung, Testausführung oder auch Bündelung zu einem Paket. Der Server lässt sich lokal in der Entwicklungsumgebung ausführen oder als gepacktes Archiv in einen *Apache Tomcat*-Server integrieren.

Für die Vereinfachung der Tokenisierung wurde die Bibliothek *Apache Lucene*[3] verwendet. Lucene ist eine Programmbibliothek für die Volltextsuche und ein mächtiges Werkzeug. Die Verarbeitung von einem 1000 Seiten Roman benötigt nur wenige Sekunden. Es wird nur ein sehr geringer Teil der Bibliothek verwendet, da eine Volltextsuche nicht nötig ist. Lucene beinhaltet eine vollständige Tokenisierung und zählt auch automatisch die Wörter pro Dokument. Zudem ist eigene Implementierung für die Wortgewichtung in der Bibliothek vorhanden und könnte für die Visualisierung genutzt werden. Für den Anfang werden die Frequenzen der Wörter genutzt und damit die Termfrequenz *tf* für alle Dokumente berechnet. Dieser berechnete Wert wird normalisiert und als Gewichtung genutzt. Zugunsten einer schnelleren Umsetzung wurden für Lucene keine lokalen Daten angelegt, sondern der RAM als Speicher verwendet. Dies bedeutet, dass alle hochgeladenen Korpora sofort gelöscht werden, sobald der Server heruntergefahren wird.

Als Server-Framework wird auf *Spring*[4] zurückgegriffen. Spring ist ein Framework mit dem schnell und ohne hohen Konfigurationsaufwand ein Java-Server aufgesetzt werden kann. Eingebunden wird das Framework mit Maven. Mit Spring lassen sich Webschnittstellen und die Zuordnung zu URL-Adressen einfach umsetzen. Die genaue Beschreibung der Schnittstellen wird in Abschnitt 5.3 beschrieben.

Um den Wörtern noch mehr Bedeutung zuordnen zu können, wurde NLP (siehe Abschnitt 2.4) genutzt. Als Framework, um unterschiedliche NLP-Tools zu kapseln und beliebig verwenden zu können, wurde *GATE (General Architecture for Text Engineering)* [5] verwendet. GATE bietet ein generisches Modell für Annotationen und Verarbeitungsreihenfolgen an. Dadurch lassen sich beliebige NLP-Tools (Stanford CoreNLP, OpenNLP, ANNIE) verwenden oder auch kombinieren. Da die Annotationen generisch und damit einheitlich hinterlegt werden, können die Tools auf die Annotation zugreifen und diese verarbeiten. Durch die Integration der NLP-Funktionalitäten in die Anwendung wächst die Größe des Servers stark an. Um unnötige Verschwendung des Hauptspeichers zu verringern wurde nur Stanford CoreNLP

[2] https://maven.apache.org/
[3] https://lucene.apache.org/core/
[4] http://projects.spring.io/spring-framework/
[5] https://gate.ac.uk/

genutzt. Die Universiy of Stanford forscht bereits seit Jahren an ihrem NLP-Tool (erste Veröffentlichung Januar 2010) und dadurch ist die Effizienz und Genauigkeit von Stanford CoreNLP der aktuelle Forschungsstand für automatisierte Sprachverarbeitung.

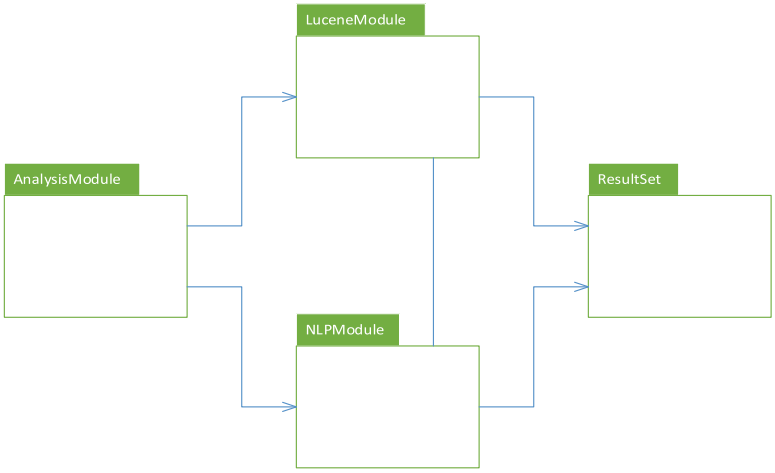


Abbildung 5.3.: Skizzierung der Serverarchitektur mit den Hauptpaketen.

Eine skizzierte Übersicht über die Serverarchitektur ist in Abbildung 5.3 zu sehen. Eine Zusammenfassung der verwendeten Tools:

- Java JDK 8
- Apache Maven
- Apache Lucene
- GATE mit Stanford CoreNLP
- Spring Framework

5.3. Webschnittstelle

Die Schnittstellen wurden nach der *REST* Struktur aufgebaut. REST steht für *Representational state transfer* und bedeutet, dass über die URL alle notwendigen Informationen an den Server übermittelt werden. Der Server muss nicht den Zustand des Client speichern und damit ist REST zustandslos. Für eine Word-Cloud-Visualisierung wird eine gewichtete Liste an Wörtern benötigt und das entspricht auch überwiegend den benötigten Schnittstellen. Es sollen Wörter für die Schnitte und Wörter für einzelne Dokumente abgerufen werden.

Im nachfolgenden steht $\{uploadId\}$ für die ID des hochgeladenen Korpus.

/uploadMulti Diese URL ist die Haupt-URL mit dem die Dokumente hochgeladen werden. Über ein POST werden dabei eine Liste an Dateien und zusätzlich ein Parameterobjekt. Das Parameterobjekt entspricht dabei den Uploadparameter wie in Abschnitt 5.5 beschrieben.

/upload/{**uploadId**}**/progress** Mit dieser Schnittstelle ist es möglich den Fortschritt eines Korpus zu erhalten. Diese Schnittstelle liefert keinen prozentualen Fortschritt, sondern einen boolschen Wert. Es wird nur dann *True* zurückgegeben, wenn die Analyse für den Korpus abgeschlossen ist.

/upload/{**uploadId**}**/numWords** Diese Schnittstelle ermöglicht es, die Wörter der analysierten Korpora abzurufen. Dabei identifiziert man den Korpus mit der ID und übergibt zwei zusätzliche GET-Parameter. Zum einen die gewünschte Anzahl der Wörter aus der Schnittmenge und die Anzahl der Wörter für reine Dokumentwörter. Die Wörter werden nach absteigender Relevanz sortiert und dementsprechend zurückgegeben.

/upload/availableResources Die Schnittstelle hat die Aufgabe alle verfügbaren Korpora in einer Liste zurückzugeben. Die Liste enthält dabei die Dateinamen, die Korpus-ID und den Zustand der Verarbeitung.

/upload/parameters Diese Schnittstelle gibt die verfügbaren Uploadparameter zurück. Diese enthält vorerst nur die Stopwortlisten.

5.4. Client-Architektur

Da es sich bei der Clientanwendung um eine rein browserbasierte Umsetzung handelt, werden die Standardtechnologien im Webbereich verwendet. Dazu zählen *HTML 5*, *CSS* und *JavaScript*. Zur Zeit der Implementierung wurde die *ECMA 6* Version von JavaScript verwendet. Die neue Version bringt zahlreiche Erneuerung mit sich. Es lassen sich beispielsweise komfortabel echte Klassenstrukturen mit JavaScript umsetzen, die zuvor nur über kompliziertere Wege generiert werden konnten. Mit der Klassenstruktur einhergehend kann eine Vererbung verwendet werden. Dadurch lassen sich Hierarchien und Strukturen aufbauen. Der Nachteil von ECMA 6 ist die noch fehlende Unterstützung der Browser. Dies hat zur Folge, dass der Code vor der Bereitstellung zurück in das ECMA 5 Syntax konvertiert werden muss, um vollständig interpretiert werden zu können. Mit Buildtools kann dieser Prozess voll automatisiert ausgeführt werden und erzeugt damit keinen zusätzlichen Aufwand.

Zur Verwaltung von JavaScript-Paketen werden *npm*[6] und *bower*[7] verwendet. Npm zählt zu den meist genutzten Frameworks, um die Pakete für JavaScript Projekte zu verwalten. Mithilfe

[6] https://nodejs.org/en/
[7] http://bower.io/

einfacher Befehle wie zum Beispiel `npm install bower` werden das Paket Bower und all ihre Abhängigkeiten installiert. Bower unterstützt die Kompatibilität von npm, da manche Buildtools npm nicht vollständig integriert haben. Die Befehle von bower unterscheiden sich dabei nur durch das erste Schlüsselwort `bower install additionalPackage`.

Als Buildtool wird *Brunch*[8] genutzt. Es kompiliert die Dateien, fasst Skripte und Vorlagen zu Modulen zusammen und konkateniert diese. Zudem lassen sich auch Funktionen wie die Minimierung von Skriptdateien einstellen, um die Dateigrößen zu reduzieren. Dies beschreibt nur einen kleinen Teil des Funktionsumfangs von Brunch. Erweitern lassen sich Funktionen auch mit weiteren Paketen. Alternative Buildtools wären zum Beispiel *Grunt* oder *Gulp*. Nach einigen Tests und eigenen Erfahrungen bewies sich Brunch als das schnellere Buildtool, bei dem die Kompilierung innerhalb weniger Millisekunden fertig ist. Brunch ist der in der Lage einen eigenen lokalen Webserver zu starten, um die Webanwendung zu testen. Eine alternative Funktion, falls der Server nicht benötigt wird, ist der Beobachtungsmodus. Dabei werden alle Dateien der Webanwendung beobachtet und bei einer Änderung direkt neu kompiliert. Dadurch hat man stets einen aktuellen Stand beim Testen und muss das Kompilieren nicht manuell starten.

Für den Entwurf und die Gestaltung der Oberfläche wird *SemanticUI*[9] genutzt. SemanticUI ist ein Framework, das für eine einfache, verständliche und reagierende Oberfläche genutzt werden kann. Dabei basieren die meisten Elemente auf HTML-Basis unter Verwendung von sprechenden CSS-Klassennamen. Ein Beispiel für die Darstellung eines einfach gestalteten Knopfes ist `class=„ui basic button“`. Das *ui* steht hierbei für Semantic-UI und die weiteren Klassennamen können beliebig vertauscht werden und erzeugen je nach Wahl unterschiedliche Elemente. Für aufwändigere Elemente, wie zum Beispiel ein Akkordeonmenü, muss JavaScript verwendet werden. Mit JavaScript werden die Elemente im Normalfall konfiguriert und dann aktiviert. Mithilfe der vielen verfügbaren Elemente und Layouts lassen sich schnell ästhetische Websites erstellen.

Zur Erstellung der Visualisierung wurde *D3.js* gewählt. D3.js ist ein bekanntes Visualisierungstoolkit auf JavaScript-Basis und für die guten und performanten Visualisierungen bekannt. Dadurch lässt es sich leicht in die Webanwendung integrieren. Echte Alternativen gibt es im Webbereich nur kaum, die einen ähnlichen Funktionsumfang liefern. Die meisten Alternativen sind dabei Erweiterungen von D3.js wie zum Beispiel *cola.js*[10], dass sich auf Layouts mit Beschränkungen (Constraints) fokussiert. Im Laufe der Implementierung wurden beide genannten Toolkits verwendet mit dem Ergebnis, dass cola.js verworfe und auf D3.js gesetzt wurde.

[8] http://brunch.io/
[9] http://semantic-ui.com/
[10] http://marvl.infotech.monash.edu/webcola/

Eine skizzierte Übersicht über die Serverarchitektur ist in Abbildung 5.3 zu sehen. Eine Zusammenfassung der verwendeten Tools:

- HTML5, CSS und JavaScript (ECMA 6)
- npm und bower
- Brunch
- SemanticUI
- D3.js und cola.js

5.5. Uploadbereich

Der Uploadbereich umfasst alle Aufgaben, die mit dem Upload von Dokumenten zu tun haben. Diese umfassen eine Dokumentenauswahl, eine Parametereinstellung und die Ansicht der hochgeladenen Korpora.

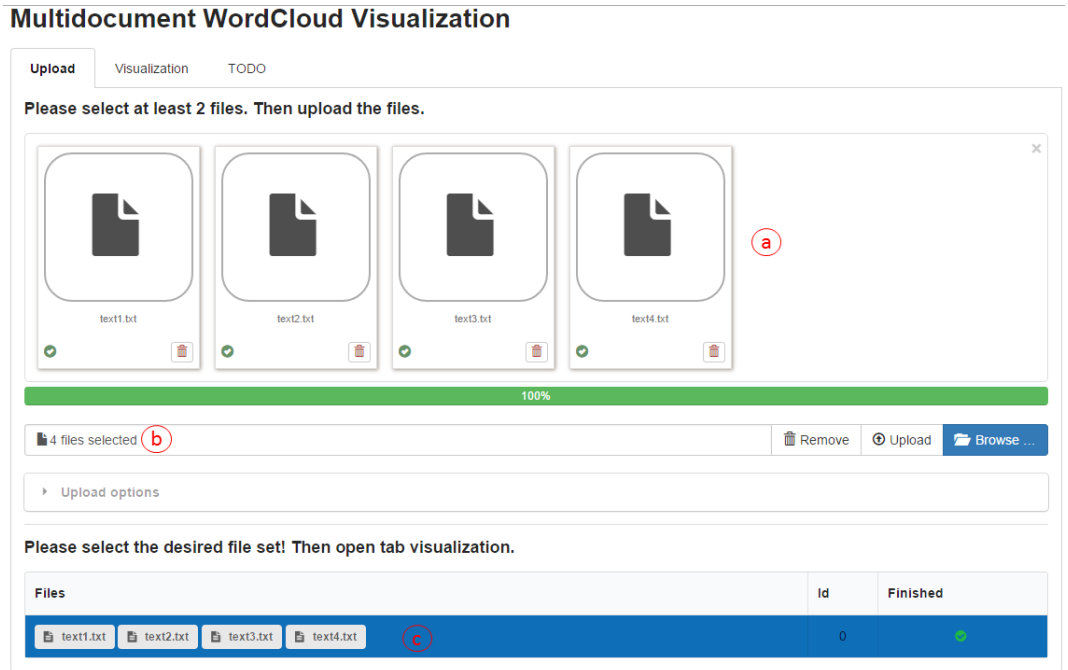


Abbildung 5.4.: Ansicht des Uploadbereiches

In Abbildung 5.4 ist der komplette Uploadbereich zu sehen.

Unter a) ist die aktuelle Dokumentauswahl zu sehen. Es lassen sich einzelne Dokumente entfernen und neue hinzufügen. Die Mindestanzahl an Dokumenten beträgt zwei und maximal

20. Die Gesamtgröße der Dokumente ist auf 10 MB beschränkt, um zu große Dateiuploads zu verhindern. Reine Textdokumente können mit 10 MB mehrere Tausend Seiten Text enthalten. Über die Buttons auf der rechten Seite findet die Bedienung statt. Mit „Browse ...“ öffnet sich ein Dateiauswahlfenster mit dem die Dateien ausgewählt werden können. Die Anzahl an Dokumenten wird bei b) angezeigt. „Remove“ entfernt alle ausgewählten Dokumente und „Upload“ startet den Uploadvorgang. Die grüne Leiste direkt darunter gibt dabei den Fortschritt des Uploads an.

Die Tabelle unter c) zeigt alle verfügbaren Dokumente an. Diese enthalten die Dateinamen, die Korpus-ID und den Analysezustand. Sobald der Zustand einen grünen Haken enthält, lässt sich der Korpus visualisieren. Der ausgewählte Korpus wird dabei farblich hervorgehoben wie es in der Abbildung zu sehen ist.

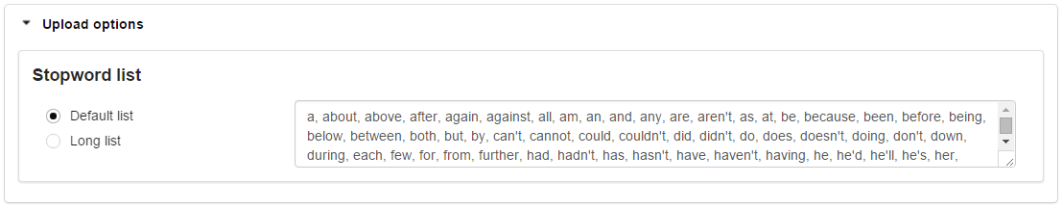


Abbildung 5.5.: Die Uploadparameter. Derzeit bestehen diese nur aus der Stopwortliste.

Abbildung 5.5 zeigt die verfügbaren Parameter für den Uploadprozess an. Dies ist vorerst auf die Stopwortliste begrenzt. Bei den Stopwortlisten stehen zwei vordefinierte Listen zur Verfügung. Über die Checkboxen kann dabei eine der beiden ausgewählt werden und die Liste auf der rechten Seite wird sofort aktualisiert. Die „Default list“ enthält dabei eine kleinere Ansammlung an englischen Stopworten. Darunter fallen zum Beispiel *a, an* oder *be*. Die „Long list“ hingegen enthält sehr viele Wörter, die bei der Analyse gefiltert werden. Zusätzlich kann die Liste manuell durch Komma getrennte Wörter beliebig erweitert werden. Die definierten Stopwörter werden bei der ersten Analyse berücksichtigt und können nicht im Nachhinein verändert werden. Das bedeutet, dass eine Änderung der Liste einen neuen Upload zur Folge hat.

5.6. Visualisierungsbereich

Dieser Teil der Arbeit befasst sich mit der Implementierung und Gestaltung des Visualisierungsbereichs. Der Bereich teilt sich in zwei Bereiche: Dem Graphen und der Steuerung. Der Graph ist immer sichtbar und wird unter dem Akkordeonmenü angezeigt (siehe Abbildung 5.6 b)). Das Akkordeonmenü ist dabei die Steuerung und ist in unterschiedliche Bereiche unterteilt. In Abbildung 5.6 a) sind die drei Bereiche zu sehen.

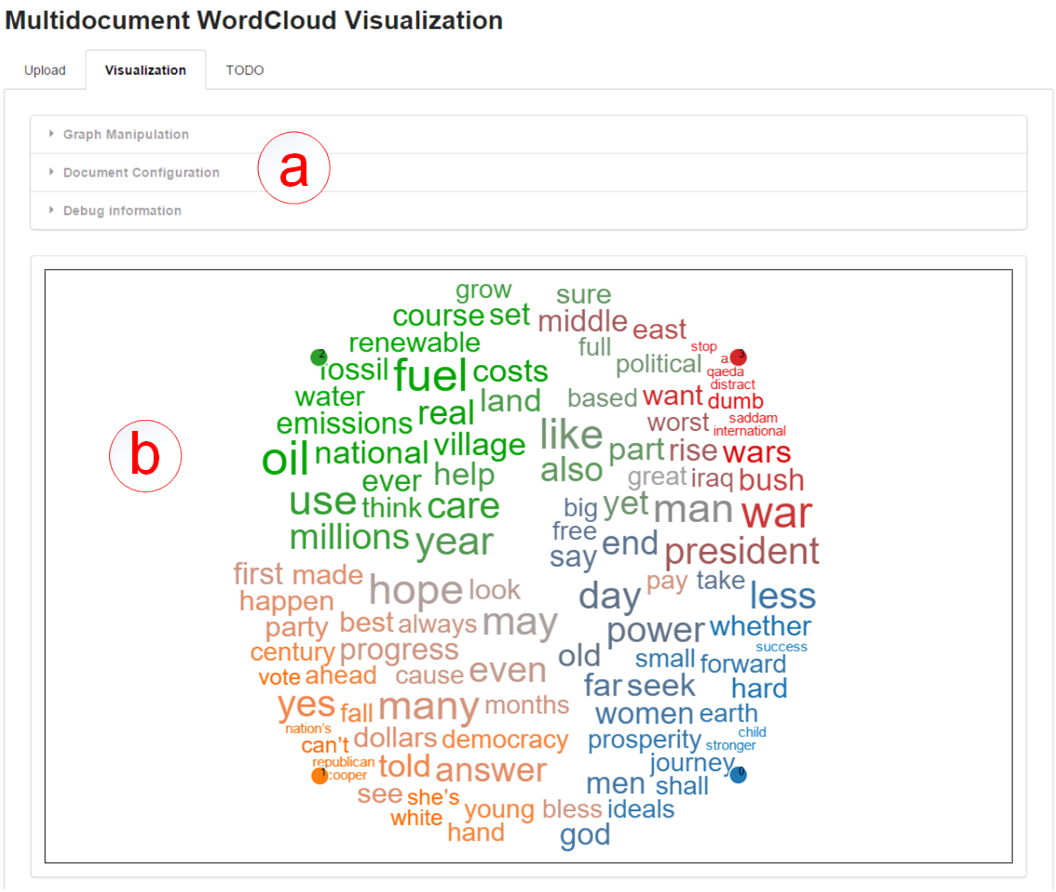


Abbildung 5.6.: Darstellung des Visualisierungsbereichs.

Graphmanipulation In diesem Bereich wird der Graph in der Gestaltung modifiziert. Dabei kann man Parameter wie Schriftgröße und Skalierung verändern. Die Änderungen werden direkt auf den Graphen übertragen.

Dokumentkonfiguration Die Konfiguration des Korpus besteht aus der Anzahl der Wörter. Unterteilt wird in Schnittworte und Dokumentworte, die separat ausgewählt werden können.

Debuginformationen In diesem Bereich werden Informationen angezeigt, die erweiterte Testinformationen zeigen. Beispielsweise wird die Anzahl an ausgelassen Wörtern angezeigt, falls der Graph so konfiguriert ist, dass Wörter ausgelassen werden können.

5.6.1. Visualisierung

Der Fokus wurde auf die Visualisierung und ihre Parameter gesetzt. Bei der Gestaltung des Layouts kam es dabei zu unterschiedlichen Prototypen mit *D3.js* und *cola.js*. Zu Anfang hatte cola.js den Anschein, dass es für diese Arbeit geeignet wäre. Im Laufe der Erweiterung der Anforderungen haben sich die Mängel gezeigt und wurde dadurch verworfen.

Layout

Hier werden die einzelnen Meilensteine der Implementierung aufgeführt und die Umsetzungen erklärt. Als Basis diente D3.js und das unterstützte Force-Layout. Als Daten werden Knoten und Kanten benötigt. Wie im Konzept Kapitel 4 entsprechen Wörter den Knoten und die Verbindungen zum Dokument den Kanten. Des Weiteren besitzt das Layout viele weitere Parameter, die das Layout und Kräfte beeinflussen. Dazu zählen die Gravitation, Kantenstärke und Kantendistanz.

Gravitation Die Gravitation ist ein Wert der zwischen 0 und 1 liegt und die Stärke der Anziehung zum Zentrum des Layouts angibt. Je höher der Wert, desto stärker werden die Knoten in die Mitte gezogen.

Kantenstärke Die Flexibilität der Kanten wird auch zwischen 0 und 1 angegeben und bedeutet wie stark sich die Kanten unter Einfluss der Abstoßungskräfte dehnen können. Je höher der Wert, desto starrer ist die Kante und verliert an Flexibilität.

Kantendistanz Gibt die Distanz der Kante an, wobei diese je nach Festlegung der Kantenstärke und wirkender Kräfte stark variieren kann.

Dem Konzept nach entsprechen die Kräfte der Knoten infinitesimal kleinen Punkten. Das bedeutet, dass die Ursprungskräfte aus einem unendlich kleinen Punkt entspringen. Je nach Festlegung der Abstoßungkräfte breitet sich die Kraft kreisförmig um diesen Punkt aus. Diese Punkte entsprechen dem Konzept des D3.js Force-Layout.

Zu Beginn der Implementierung war die Idee, durch geeignete Wahl der Parameter ein überschneidungsfreies und gut verteiltes Layout zu erhalten. Wörter werden mit SVG gezeichnet und als Knoten dargestellt. Zu jedem Dokument erhält ein Wort eine Kante mit der Stärke der Verbindung zum Dokument.

Das Resultat ist in Abbildung 5.7 zu sehen. Trotz Variationen der Parameter und dem Versuch die Überschneidungen zu entfernen, kam es durch die vielen einwirkenden Kräften doch immer zu Überschneidungen. Die Wörter haben zu diesem Zeitpunkt noch eine zufällig verteilte Zugehörigkeit erhalten. Die kleinen schwarzen Linien geben die Position der Originalposition an. Trotz der Überlappungen ist jedoch zu sehen, dass die Positionen kaum

Abbildung 5.7.: Erster Versuch der Darstellung. Die Dokument wurden an den äußeren Bereichen angebracht und das kräftebasierte Layout wurde angesetzt.

von den ursprünglichen Positionen abgewichen sind. Das kräftebasierte Layout ist daher für diese Aufgabe gut geeignet.

Eine Idee, um das Problem der Überlappung zu lösen, war eine Kollisionserkennung zu integrieren. Dieses sollte nach jeder Iteration des Kräftelayouts eine Kollisionserkennung starten und die kollidierten Wörter auseinander bewegen. Dieser Ansatz funktioniert nicht sehr gut, da Wörter eine rechteckige Form haben (Bounding-Box) und um diese Kollisionen aufzulösen, bedarf es aufwendigen Algorithmen (siehe Kapitel 4). Ein einfacher Test hat gezeigt, dass die Kollisionserkennung mit Kreisformen einfach und effektiv und ist in Abbildung 5.8 zu sehen.

Das Problem bei Kreisformen ist jedoch anhand des benötigten Platzes schnell erkennbar. Es würde viel Platz über und unter einem Wort übrig bleiben und je größer ein Wort ist, desto mehr Platz geht verloren. Mögliche Optimierungen wären die Schachtelung von mehreren Wörtern in einem Kreis, wenn diese klein genug sind. Das könnte ein besseres Resultat erzeugen, aber aufgrund von hohem Risiko wurde die Idee nicht weiter verfolgt.

Die Bibliothek cola.js ist eine Modikfikation der D3.js Bibliothek. Der Fokus liegt dabei bei der Nutzung von Beschränkungen (Constraints), um ein überlappungsfreies Layout zu erzeugen. Eine direkte Portierungsanleitung von dem D3.js Force-Layout zum cola.js Constraint-Layout ist auch direkt auf der Seite angegeben.

Für kleine Beispielgraphen, bei denen die Positionen keine Aussage haben, wirkt das generierte Layout gut. Die zuvor erwähnten Überlappung werden sowohl bei Kreisen als auch bei Rechtecken und beliebigen Formen gelöst. Gleichzeitig ist diese starke Begrenzung von Wörtern auch der größte Nachteil im Bezug auf diese Arbeit. Die Constraints erlauben es

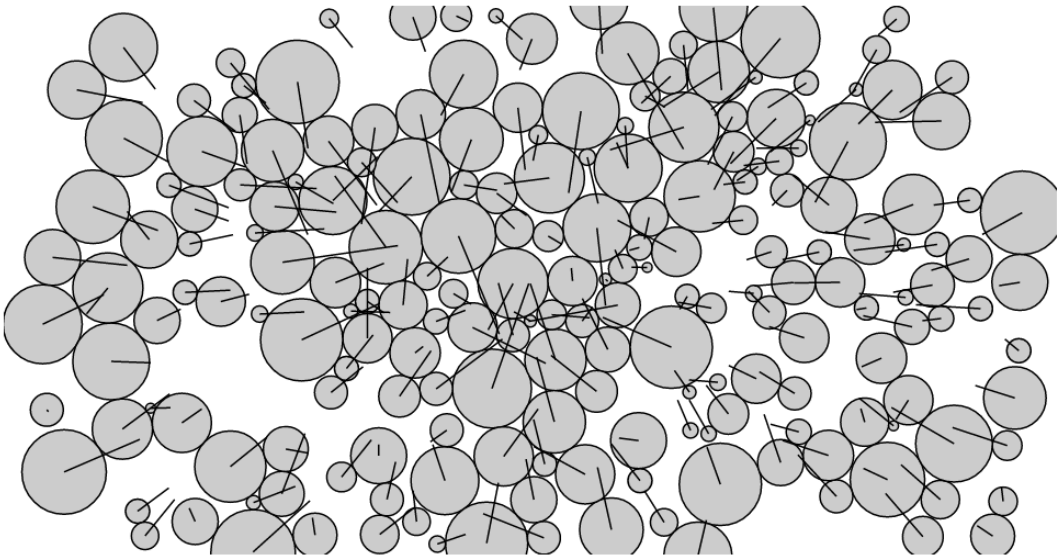


Abbildung 5.8.: Versuchsimplementierung, um die Kollisionserkennung an Kreisformen zu testen.

kaum eine Form auszufüllen da sich die Rechtecke blockweise bewegen. Für eine zufällig gewählte Position sieht das Layout ansprechend aus wie in Abbildung 5.9. Durch ungeschickte Positionen können sich auch ganze Blöcke verschieben und es entsteht ein Bild wie in Abbildung 5.10.

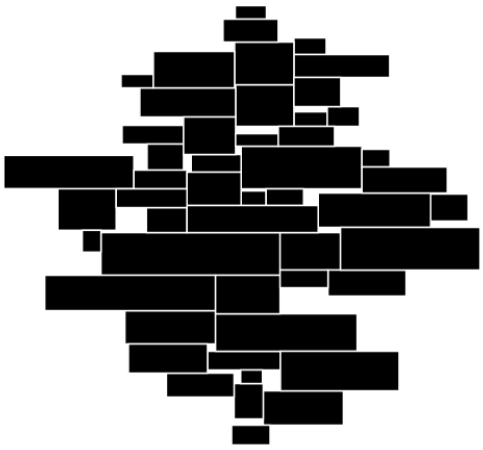


Abbildung 5.9.: Auf den ersten Blick ansprechende Visualisierung mit Cola.js.

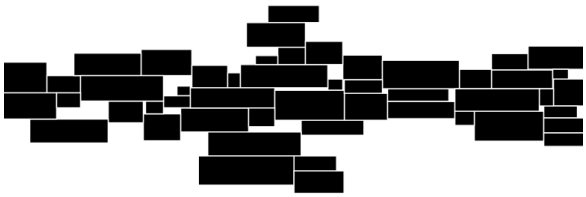


Abbildung 5.10.: Der Nachteil der cola.js Constraints ist schnell sichtbar. Die Rechtecke ordnen sich zu einer länglichen Form an anstatt ein kompaktes Layout zu erzeugen.

Durch Erweiterung der Constraints mit weiteren Begrenzungen wurde versucht, dieses Problem zu lösen. Jedoch brachte es kein befriedigendes Ergebnis, da durch unglückliche Verteilungen immer wieder ein schlechtes Layout bilden konnte.

Nach Verwerfen der cola.js Bibliothek wurde die Idee mit der Verschiebung von Wörtern in eine Ecke entworfen. Zu Beginn sollten die Wörter für Testzwecke in die obere rechte Ecke der verfügbaren Zeichenfläche geschoben werden.

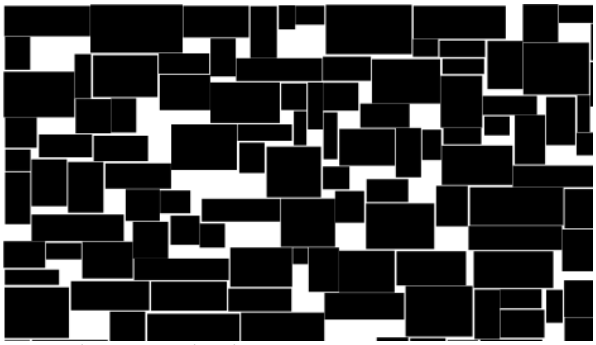


Abbildung 5.11.: Prototyp der Verschiebung von Worten in eine Ecke.

Durch eine einfache Implementierung, bei der die Wörter nach der Entfernung zur oberen linken Ecke sortiert und anschließend zur Ecke verschoben werden, wurde der Prototyp erzeugt. In Abbildung 5.11 ist das Ergebnis zu sehen. Dieses sieht vielversprechend und kompakt aus. Dies war der Grundstein für die endgültige Version der Visualisierung. Die Idee die Wörter je nach Entfernung von einem Fixpunkt in eine Richtung zu verschieben.

Das Endergebnis geht dabei wie folgt vor. Zuerst werden die Dokumentpunkte verteilt. Dabei werden die Dokumente auf einem Kreis verteilt, der den Radius von der Breite oder Höhe der Zeichenfläche hat. Dabei wird stets die größte Einheit genommen. Anschließend wird der Kreismittelpunkt in den Mittelpunkt der Zeichenfläche gelegt und die Dokumente gleichmäßig auf dem Kreis verteilt. Sobald die Position der Dokumente berechnet wurde, werden diese in Richtung des Mittelpunktes verschoben bis die Grenzen der Zeichenfläche erreicht sind. Dies ist in Abbildung 5.12 zu erkennen. Durch das kräftebasierte Layout verteilen sich die Wörter nun auf der Zeichenfläche. Zwar haben diese noch Überschneidungen werden mit aber in der weiteren Verarbeitung gelöst.

Ausgehend von den Dokumentpositionen werden die Wörter neu auf der Zeichenfläche verteilt. Der Algorithmus geht dabei so vor, dass die Wörter pro Dokument nach der Entfernung sortiert werden. Diese sortierte Liste gibt die Reihenfolge der Platzierungen der Wörter an.

Im zweiten Schritt werden die Pixel nach der Entfernung zum Dokument sortiert. Diese Sortierung gibt die Reihenfolge an, an denen versucht werden soll, ein Wort zu platzieren. Sollte ein Wort nicht an der Pixelposition passen, so wird der nächste Pixel getestet. Ein skizzierter Positionierungsverlauf ist in Abbildung 5.13 zu sehen.

Nach der Vorverarbeitung der Position beginnt der Positionsalgorithmus. Dabei wird rundenbasiert jeweils ein Wort für ein Dokument gesetzt. Zum Beispiel wird für Dokument 1 das erste Wort aus der sortieren Wortliste genommen und platziert. Anschließend wird ein Wort aus Dokument 2 platziert etc.

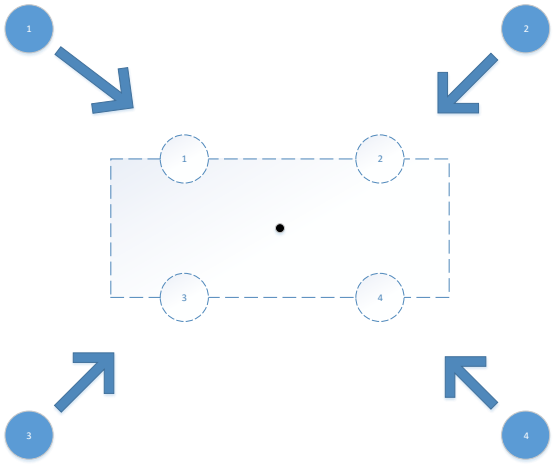


Abbildung 5.12.: Zeigt den Verlauf der Positionierung der Dokumente an. Die Dokumente werden gleichmäßig auf einem Kreis verteilt und anschließend in Richtung der Zeichenfläche geschoben.

Der Algorithmus läuft so lange bis keine Worte mehr in den Listen verfügbar sind. Falls ein Wort nicht positioniert werden kann, wird es aus der Liste genommen und als nicht platzierbares Wort gekennzeichnet. Um diesen Fall zu umgehen, können Parameter eingestellt werden (siehe Abschnitt 5.6.2).

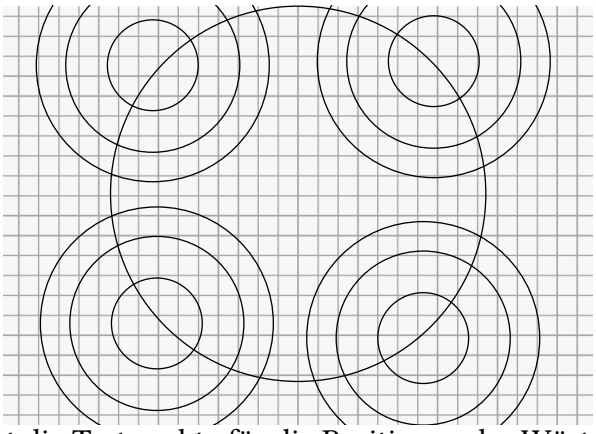


Abbildung 5.13.: Zeigt die Testpunkte für die Positionen der Wörter. Ausgehend von Dokumenten werden die Pixel radialförmig getestet, ob das Wort an der Position genügend Platz hat.

Durch die Sortierung der Wörter und rundenbasierten Positionierung wird ermöglicht, dass stark zugehörige Wörter sehr nah den Dokumenten liegen.

Unter Umständen könnte auch der Fall eintreten, dass Wörter keinen Platz in ihrer ursprünglichen Umgebung haben. Der Algorithmus läuft dabei solange weiter bis alle Pixel der

Zeichenfläche getestet wurden. Dadurch könnte es vorkommen, dass Wörter an komplett falsche Positionen gezeichnet werden. Um diesen Fall zu unterbinden gibt es einen Toleranzbereich für die Wörter, in dem eine Platzierung erlaubt ist. Liegen Testpixel außerhalb des Bereiches werden diese ignoriert. Durch die Einführung des Toleranzbereiches steigert sich die Anzahl der ausgeschlossenen Wörter, da ein eine zusätzliche Beschränkung hinzu kommt.

Farbgebung

Um Dokumente und Wörter gut unterscheiden zu können, wird eine gut unterscheidbare Farbgebung benötigt. Die Anzahl an unterscheidbaren Farben, die ein Mensch differenzieren kann, liegt dabei nicht sehr hoch (siehe Kapitel 2). D3.js bietet gute Farbskalen an, die sich gut unterscheiden.[11] Die Kategorien 10 und 20 sind dabei die nützlichsten. Diese bieten eine vordefinierte Farbpalette für 10 oder 20 Farben, die sich visuell bestmöglich unterscheiden lassen. Für eine vorläufige Version unterstützt die Visualisierung die 10er Kategorie. Dadurch sind die Dokumente und Wörter gut voneinander zu trennen.

Jedes bekommt dabei eine Farbe aus der Kategorie. Anschließend bekommen die Wörter die Farbe des Dokuments, zu dem die Zugehörigkeit am größten ist. Eine weitere farbrelevante Funktion ist die Variation in der Sättigung der Farbe. Dabei wird die Sättigung nach der Zugehörigkeit skaliert (siehe Abschnitt 4.4). Die Funktion sieht dabei wie folgt aus:

$$f(x) = \begin{cases} 0 & x \leq 0,3 \\ \frac{1}{0,8-0,3} \cdot (x - 0,3) & 0,3 < x < 0,8 \\ 1 & x \geq 0,8 \end{cases} \tag{5.1}$$

Wie in Gleichung 5.1 zu sehen, wird einer Zugehörigkeit von 0,3 oder niedriger bereits die Sättigung auf 0 gesetzt. Bei einer Zugehörigkeit von 0,8 oder höher wird die Farbe mit voller Sättigung gezeichnet. Zwischen 0,3 und 0,8 skaliert die Funktion die Sättigung linear. Diese Sättigungsfunktion soll eine Trennung von stark zugehörigen Wörtern erleichtern und verteilte Wörter klar abtrennen.

Interaktion

Als Interaktion wurde ein Tooltip integriert, dass erscheint, sobald der Mauszeiger einen kurzen Moment über einem Wort schwebt. Dieser Tooltip wird dann direkt neben dem Wort angezeigt.

[11] https://github.com/mbostock/d3/wiki/Ordinal-Scales

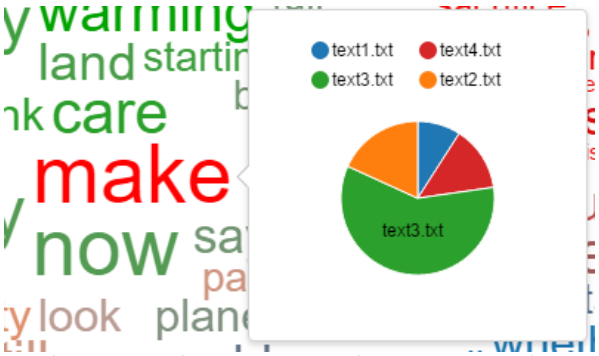


Abbildung 5.14.: Ein Tooltip zeigt die genaue Dokumentverteilung an, sobald der Mauszeiger über einem Wort schwebt.

Das Tooltip ist in Abbildung 5.14 zu sehen. Das Wort über dem der Mauszeiger aktuell schwebt, wird dabei mit einer roten Farbe hervorgehoben. Eine Pfeilform am Tooltip verdeutlicht die Zuordnung zum Wort.

Im Tooltip wird die genaue Dokumentenverteilung durch ein Kreisdiagramm angezeigt. Zur Visualisierung dieses Diagramms wurde eine erweiterte Implementierung von D3.js verwendet. Die Bibliothek heißt *NVD3* [12] und fokussiert sich auf wiederverwendbare Diagramme. Damit lassen sich schnell kleine Diagramme generieren. Zudem wird bereits eine Möglichkeit geboten eine Legende zu integrieren.

5.6.2. Grafische Manipulation

Die Manipulation der Visualisierung im Bezug auf Skalierung der Wörter, Formen und Nutzung von Parametern war ein Kriterium, das bereits zu Beginn der Konzeption gewünscht war. Daher wurden unterschiedliche Manipulationsmöglichkeiten integriert und werden in einem eigenen Bereich des Akkordeonmenüs dargestellt.

Schriftgröße und Skalierung

Die initiale Schriftgröße wird anhand der Wurzelskalierung berechnet. Eine Wurzelskalierung verhindert den Visual-Lie-Faktor (siehe Abschnitt 2.1) und bietet gleichzeitig eine ansprechende Aufteilung der Schriftgrößen an. Relevante Wörter erhalten durch die Skalierung eine große Schrift und stechen hervor.

Nach der initialen Berechnung wird die minimale und maximale Schriftgröße in eigenen Feldern angezeigt (siehe Abbildung 5.15). Diese lassen sich anschließend je nach Wunsch

[12] http://nvd3.org/

verändern und werden sofort auf die Visualisierung übertragen. Zusätzlich werden noch drei weitere Skalierungsarten angeboten. Die lineare und quadratische Skalierung weisen den Wörtern dabei überwiegend kleine Schriftgrößen zu und unterscheiden sich eher bei großen Differenzen in dem Auftreten. Die logarithmische und Wurzelskalierung ordnen den Wörtern dabei eher große Schriftgrößen zu. Dabei sind die Schriftgrößen und Skalierungen beliebig kombinieren und haben sofortigen Effekt auf den Graphen.

Abbildung 5.15.: Die unterschiedlichen Anpassungen der Schriftgrößen und deren Skalierungen.

Formen und Parameter

Das Layout der Visualisierung kann mit drei unterschiedlichen Formen generiert werden. Diese können unter Abbildung 5.16 gewählt werden.

Rechteck Bei der Rechtecksform wird das komplette Feld des Graphen verwendet und die Dokumente am Rahmen verteilt.

Kreis Bei der Kreisform wird die kleinere Breite oder Höhe genommen und als Radius für den Kreis verwendet. Durch die Wahl der kleinen Größe bleibt der Kreis innerhalb des Rahmens.

Ellipse Da die Ellipse zwei Parameter benötigt wird die Höhe und Breite genommen und gezeichnet.

Zusätzlich werden noch Parameter angeboten, die die Visualisierung sehr stark beeinflussen.

Maximale Platzausnutzung Mit dieser Option wird der Graph dahingehend verändert, dass der verfügbare Platz möglichst optimal genutzt wird. Der Algorithmus wird dabei so angepasst, dass vor der Verteilung der Wörter die Schriftgrößen angepasst werden. Der Pseudecode ist in dem Algorithmus 5.1 dargestellt. Die Wörter werden, wenn

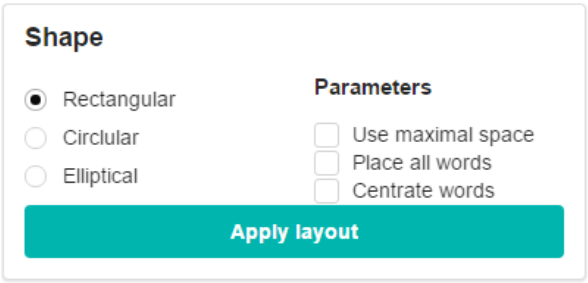


Abbildung 5.16.: Die Darstellung der unterschiedlichen Formen und Parameter, die das Layout der Visualisierung beeinflussen.

nötig, so lange in ihrer Größe inkrementiert bis der benötigte Platz größer ist als der verfügbare. Anschließend wird die Schriftgröße einmal dekrementiert, damit der vorhandene Platz wieder größer ist.

Platzierung aller Wörter Diese Option ist wichtig, wenn alle Wörter dargestellt werden sollen und nicht ausgelassen werden können. Je nach Anzahl an Wörtern kann jedoch eine Situation eintreten bei der nicht alle Wörter platziert werden können. Dies hängt mit dem Toleranzradius der Wörter zusammen in denen sie platziert werden können. Bei vielen Wörtern kann es durchaus vorkommen, dass dieser Toleranzradius immer überschritten wird, so dass der Algorithmus dann fehlschlägt. Diese Option hat eine höhere Priorität als die maximale Platzausnutzung. Der Algorithmus wird dabei so angepasst, dass die Schriftgrößen der Wörter reduziert werden, sobald ein Wort ausgelassen werden würde. Anschließend wird die Zeichenfläche zurückgesetzt und ein erneuter Versuch für die Platzierungen wird gestartet. Durch das ständige Zurücksetzen der Zeichenfläche verlängert diese Option die Laufzeit des Algorithmus stark. Der Code ist im Pseudocode 5.2 skizziert.

Zentrierung der Wörter Mit dieser Option wird am Ende der Platzierung, versucht das erstellte Layout noch weiter zu optimieren. Dabei werden die Wörter ausgehend vom Zentrum sortiert und anschließend nacheinander soweit in Richtung der Mitte verschoben bis es zu einer Kollision kommt. Dieses Vorgehen orientiert sich dabei an dem vorgestellten Prototypen in Abbildung 5.11. Der freie Platz im Zentrum wird reduziert und das Gesamtbild in die Mitte geschoben. Ein Nachteil dieser Option ist, dass die Form dabei stark verändert wird und nicht mehr der ursprünglichen gewählten Form entsprechen kann.

Die drei vorgestellten Optionen sind beliebig miteinander kombinierbar. Wie bereits erwähnt besitzen die Optionen unterschiedliche Prioritäten. Die Platzierung aller Wörter hat dabei die höchste Priorität, da dies eine der wichtigsten Parameter ist, wenn eine Auslassung der Wörter nicht tolerierbar ist.

Algorithmus 5.1 Pseudocode für die Option: Maximale Platzausnutzung

```
procedure OptimalSpace(availableSpace, wordList)
    usedSpace = calculateUsedSpace(wordlist)
    while availableSpace > usedSpace do
        incrementSize(wordList)
        usedSpace = calculateUsedSpace(wordlist)
    end while
    decrementSize(wordList)
end procedure
```

Algorithmus 5.2 Pseudocode für die Option: Platzierung aller Wörter

```
procedure PlaceAllWords(wordList)
    for all word in wordList do
        wordSkipped = placeWord(word)
        if wordSkipped is True then
            decrementSize(wordList)
            resetGraph()
            return
        end if
    end for
end procedure
```

5.6.3. Dokumentinformationen

Bei dieser Modifikation wird die Anzahl der Wörter verändert. In Abbildung 5.17 ist das Menü zu sehen.

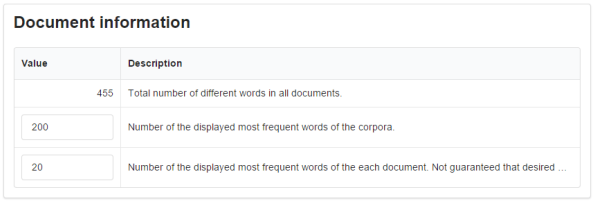


Abbildung 5.17.: Es wird die Anzahl der unterschiedlichen Wörter angezeigt. In zwei seperaten Feldern können die Anzahl an Schnittwörtern und Dokumentwörtern ausgewählt werden. Der Graph wird anschließend sofort aktualisiert.

In einem Feld wird die Anzahl der unterschiedlichen Wörter des Korpus angezeigt. Wiederholte Wörter werden hier nicht mitgezählt. Gleichzeitig gibt dieser Wert maximale Anzahl an Wörtern an, die in der Visualisierung ausgewählt werden können. Im zweiten Feld kann die Anzahl für Wörter eingestellt werden, die in einem Dokumentenschnitt vorkommen.

Aus einer nach Relevanz sortierten Liste wird die gewünschte Anzahl an Wörtern herausgenommen. Das letzte Feld gibt die Anzahl an reinen Dokumentwörtern an. Diese tauchen in keinem Schnitt auf und haben damit nur eine Bindung zu einem Dokument. Die Änderungen der Anzahlen werden sofort durchgeführt und aktualisieren den Graphen. Dabei kann eine gewählte Form wieder aufgelöst werden, da das kräftebasierte Layout wieder aktiviert wird.

5.6.4. Zusätzliche Informationen

In diesem Bereich werden Informationen angezeigt, die keine direkte Abhängigkeit zum Graphen haben. Dazu zählen beispielsweise die Anzahl der Wörter. Zudem kann über eine Checkbox die Darstellung der Originalposition aktiviert werden. Wenn die Box aktiviert ist, wird eine Kante von den Wörtern zu den ursprünglichen Positionen gelegt. Damit können mögliche Fehler oder starke Differenz erkannt werden.

5.6.5. Fehlermaß

Ein interessanter Aspekt bei neuen Visualisierungen ist der Fehlergrad oder Abweichungsgrad. Dieser lässt sich in diesem Fall auch gut mithilfe visueller Elemente darstellen.

Im Falle von Word-Clouds bei denen die Position der Wörter eine Rolle spielt, kann eine Kante oder Pfeil zur Hilfe genommen werden. Die Kante wird dabei von der aktuellen Position des Wortes zu der ursprünglichen Position. Dadurch werden Fehler erkannt, wenn die Kanten sehr lang werden. Im Optimalfall sollten alle Kanten kurz bleiben und sich in den Richtungen nicht stark unterscheiden. Im Falle der Multidokumenten-Word-Cloud würde ein starker Fehler eine falsche Dokumentenzugehörigkeit suggerieren (siehe Abbildung 5.18).

Abbildung 5.18.: Die Fehlerinformation wird mithilfe von einer Linie dargestellt, die zur ursprünglichen Position zeigt. Je länger die Linie desto größer die Abweichung.

6. Anwendungsbeispiel

Im Folgenden wird als Anwendungsbeispiel auf die Harry Potter Buchreihe eingegangen. Harry Potter ist ein sehr verbreiteter Roman und bietet einen Korpus mit sieben Dokumenten. Dadurch bietet es sich an, die Buchreihe mit dem vorgestellten Ansatz zu untersuchen. Dabei können Gemeinsamkeiten und Unterschiede in den einzelnen Buchteilen aufgedeckt werden.

Für den folgenden Anwendungsfall wurden die ersten sechs Harry Potter Teile in die Anwendung geladen. Alle Bücher sind in der englischen Sprache verfasst. Zur Untersuchung der Bücher werden folgenden Gesichtspunkte betrachtet.

1. Die am häufigsten auftretenden Wörter wollen Gemeinsam betrachtet werden. Unter diesen Wörter wird vor allem das Wort „Harry“ erwartet, da dies die zentrale Hauptfigur der Romane ist und hervorstechen muss.
2. Einen Überblick über die dokumentspezifischen Wörter erhalten. Dadurch sollen eventuelle Unterschiede in den Büchern erkannt werden. Welche Szenen und Ereignisse sind individuell in den einzelnen Teilen aufzufinden.
3. Eine kombinierte Ansicht sowohl von der Schnittmenge als auch von den Dokumentwörtern. Das sollte den typischen ersten Einblick auf den hochgeladen Korpus geben und den Nutzer in die Anwendung hinführen.

Im ersten Schritt wird nach Auswahl des Korpus ein Layout generiert. Dabei wurde für den ersten Schritt ein Kreislayout gewählt und die Optionen *Use maximal space* und *Place all words* gewählt. Damit wollen wir eine schnelle und gute Visualisierung für den Anfang generieren. Das Ergebnis ist in Abbildung 6.1 zu sehen.

Die Standardkonfiguration ist dabei eine Menge von 200 Wörtern aus dem Schnittbereich und jeweils 20 Wörter aus den einzelnen Dokumenten. Im Zentrum sind die grauen Wörter gut zu erkennen. Da die Wörter grau sind, bedeutet es, dass sie verteilt in den Dokumenten auftreten und keine richtige Zugehörigkeit besitzen. Zusätzlich ist ihre Position auch noch im Zentrum verteilt. Der Whitespace ist dadurch zu erklären, dass die Wörter wie in der Implementierung beschrieben einen Toleranzradius aufweisen und die Wörter nicht an beliebige Positionen gesetzt werden können. Die größten Wörter der Visualisierung sind unter anderem Harry, Hermione und Dumbledore. Diese drei Charaktere sind Hauptrollen in

Abbildung 6.1.: Als Überblicksvisualisierung wurde ein Kreislayout mit den beiden Parameter *Use maximal space* und *Place all words* gewählt. Im Zentrum der Word-Cloud sind die Schnittwörter gut zu erkennen. An den Dokumenten liegen die dokumenspezifischen Wörter.

den Harry Potter Büchern und sollten klar hervorstechen. Eine Verteilung zu den einzelnen Dokument lässt sich mithilfe des Tooltips bewerkstelligen (siehe Abbildung 6.2).

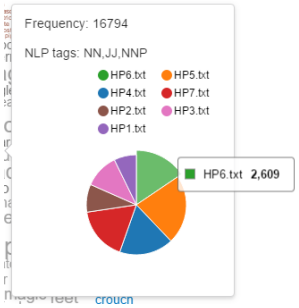


Abbildung 6.2.: Das Tooltip, das beim Hovern über ein Wort dargestellt wird.

Wenn der Fokus jetzt nur auf die Schnittmenge gesetzt werden soll, wird die Anzahl der Dokumentwörter auf 0 gesetzt und die Schnittwörter auf einen beliebigen Wert gesetzt. Ein Beispielgraph ist in Abbildung 6.3 zu sehen. Je mehr Wörter aus der Schnittmenge genommen werden, umso wahrscheinlich wird es auch Wörter zu erhalten, die stärke Anziehungen zu einzelnen Dokumenten haben. Dies ist zum Beispiel an lupin und umbridge zu erkennen.

Wenn die Analyse sich auf die einzelnen Dokumentwörter beschränkt, können die Schnittmengenwörter auf 0 gesetzt werden. Anschließend wird die Dokumentwortanzahl auf 70

Abbildung 6.3.: In dieser Visualisierung wurde die Anzahl an Dokumentwörtern auf 0 gesetzt, damit die reinen Schnittwörter dargestellt werden.

erhöht und die Word-Cloud generiert. In Abbildung 6.4 ist das Ergebnis zu sehen. Die Dokumentwörter verteilen sich ästethisch am Kreisrand und Dokumentwörter verteilen sich zum Zentrum hin. Da die großen Schnittmengenwörter nicht enthalten sind, kommen die Dokumentwörter zum Vorschein und werden höher skaliert.

Abbildung 6.4.: In dieser Visualisierung liegt der Fokus auf den Dokumentwörtern. Die Schnittmenge wurde auf 0 gesetzt und stattdessen die Dokumentwortanzahl auf 70 erhöht. Dadurch kommen die Wörter besser zur Geltung.

Eine andere Methode wäre noch das Hovering über ein Dokument, wenn man auch Schnittmengen enthalten hat. Dadurch werden die Dokumentwörter hervorgehoben wie in der Abbildung 6.5 zu sehen.

Abbildung 6.5.: Darstellung der Dokumentwörter, wenn über ein gewünschtes Dokument gehovered wird.

Im Gegensatz zu ConcentriCloud (siehe Abbildung 6.6) werden alle Wörter dargestellt, wenn es gewünscht ist. Im äußeren Ring werden noch überwiegend die meisten Worte dargestellt. Je weiter Schnittmengen gebildet, desto größer wird der Fehler von ConcentriCloud. Im Zentrum wird nur ein ganz geringer Anteil an Wörtern dargestellt, da die Bücher sehr viele Gemeinsamkeiten aufweisen. Im Bereich des mittleren Ringes werden viele größere Wörter ebenfalls ausgelassen, da diese nicht genügend Platz haben um gezeichnet zu werden. Falls zu viel Platz zur Verfügung ist, entstehen freie Flächen wie in der Abbildung zu sehen ist.

Abbildung 6.6.: Die Visualisierung von ConcentriCloud für die Harry Potter Bandreihe.

Ein weiterer negativer Punkt von ConcentriCloud ist die fehlende Zuordnung von Wörtern. Zwar lässt sich die Anzahl der Wörter zu den einzelnen über einen simplen Tooltip anschauen, jedoch wird keine weitere visuelle Unterstützung gegeben. Man kann dadurch nicht sagen, zu welchem Dokument ein Wort eher tendiert.

7. Evaluation

Um den implementierten Ansatz zu validieren, wurde eine Nutzerstudie durchgeführt. Als Probanden wurden Experten im Bereich der Visualisierung zugezogen. Die Befragung mit den Antworten sind unter Anhang B und Anhang C zu finden. Diese wiesen ein solides Vorwissen im Bereich der Word-Cloud-Visualisierungen auf, sowie im Bereich der Multidokumenten-Visualisierungen. Teilgenommen haben sieben Probanden, davon sechs männlich und eine weiblich. Das Alter lag von 27 bis 32 mit einem Schnitt von 30 Jahren. Zusätzlich wurde nach dem Vorwissen auf einer Skala von 1 bis 10 im Bereich von Word-Cloud-Visualisierungen und Multidokumenten-Visualisierungen gefragt. Die Skala geht dabei von keinem Wissen (1) bis zu sehr gutem Wissen (10). Im Bereich der Word-Cloud-Visualisierung lag der Schnitt bei 7,28. Für die Multidokumenten-Visualisierungen war der Schnitt minimal schlechter mit 6,85. Die Tendenz lag in beiden Bereichen bei einem guten Vorwissen. Damit waren die Probanden ideal für die Studie geeignet.

Der Ablauf der Studie sah dabei wie folgt aus:

1. Vorstellungen der beiden Visualisierungen ConcentriCloud und MultiCloud.
2. Probeaufgaben anhand eines kleineren Korpus.
3. Die Buchreihe von Harry Potter wurde als Eingabe für die Visualisierungen verwendet. Anschließend wurden Aufgaben gestellt, die der Proband zu lösen hatte. Die Aufgaben befinden sich im Anhang A.
4. Abschließend wurde eine Befragung bezüglich des Vorwissens und der Einschätzung welche Aspekte positiv und negativ bei beiden Visualisierungen aufgefallen sind.

Die Probanden kamen mit den gestellten Aufgaben überwiegend problemlos zurecht. Bei Aufgaben, die mit dem Suchen von Worten verbunden waren, stach ConcentriCloud negativ heraus. Das liegt daran, dass keine Suchfunktion in die Word-Cloud integriert ist und keine Sortiertfunktion bei der Wortliste vorhanden ist. In diesem Punkt sah man einen klaren Vorteil in MultiCloud.

Zu der Frage welche Visualisierung komfortabler zu bedienen sei, wurde fast einheitlich für die MultiCloud gestimmt. Ein Proband jedoch, sah keine der beiden Visualisierung im Vorteil. Der Grund für diese Meinung bestand aus einer nicht intuitiven Bedienung beider. Gleichzeitig erwähnte der Proband jedoch, dass es aufgrund von fehlender Erfahrung mit

den Visualisierung zusammenhängen könnte. Zusammenfassend wurden folgende Punkte für die Präferenz genannt:

- Die MultiCloud umfasst nicht so viele Einzelelement wie in der ConcentriCloud und wirkt dadurch besser verteilt und übersichtlich.
- Schnitte lassen sich besser untersuchen, da man die Dokumente auswählen kann. Bei ConcentriCloud muss zuerst ein Überblick über die Schnittkonstellationen gewonnen werden.
- Manche Aufgaben waren nicht mit der ConcetriCloud lösbar, wie zum Beispiel den Schnitt von gegenüberliegenden Dokumenten.
- MultiCloud ist intuitiver zu bedienen als die ConcentriCloud.
- Die Verteilung über den Tooltip ist sehr ansprechend.
- Die Farbcodierung mithilfe der Sättigung ist sehr gelungen.
- Webanwendung ist zukunftssicher und aktuell.
- Die Suchfunktion und Interaktion von MultiCloud ist besser als bei ConcentriCloud.
- Konfigurierbarkeit von MultiCloud.
- Das Zoomen und Verschieben der Visualisierung.

Ein Punkt sprach jedoch für die ConcentriCloud und zwar bei einer Aufgabenstellung bei der die Frequenzen verglichen und gesucht werden sollen. Da ist die Auflistung aller Wörter in der ConcentriCloud besser geeignet, da eine solche Funktionalität in MultiCloud nicht vorhanden ist.

Insgesamt wurde es einige Male erwähnt, dass ein Radiallayout für Vergleiche nur bedingt geeignet ist. Das trifft auf die ConcentriCloud und die MultiCloud zu. Bei ConcentriCloud wurde die Radialverteilung der Dokumente genutzt und bei MultiCloud bei den Wortverteilungen und der Dokumentgröße.

Im folgenden werden die Bemerkungen zu ConcentriCloud und MultiCloud dargestellt.

7.1. ConcentriCloud

Bei ConcentriCloud wurden folgende Punkte als positiv angemerkt:

- Der Bereich im Zentrum für die Gesamtschnittmenge.
- Die Strukturierung durch die unterschiedlichen Bereiche. Explizite Schnitte sind dadurch leichter zu erkennen und ermöglichen unerfahrenen Nutzern ein leichteres Einarbeiten.
- Eine tabellarische Auflistung der Wörter innerhalb eines Bereichs sortiert nach der Frequenz.
- Dokumente erhalten je nach Dokumentgröße mehr Platz für die Wörter.

Im Gegensatz wurden auch einige negative Punkte im Bezug auf die Verwendung und das Konzept genannt.

- Die Dokumentgröße ist schwierig einzuschätzen.
- Wörter werden einfach übersprungen, wenn sie kein Platz haben.
- Detailinformationen lassen sich nur schwierig finden.
- Die Gesamtgröße ist nicht gut.
- Ein Vergleich von gegenüberliegenden Dokumenten ist nicht möglich.
- Fehlende Such- und Sortierfunktion.
- Es ist nicht klar, wie die Dokumente verteilt sind.
- Es gibt keinen Bezug der Schriftgröße in den einzelnen Teilbereichen.
- Layout wirkt wie eine notdürftige Lösung.
- Ein visuelles Vergleichen der Wörter ist nur schwer möglich.
- Die Positionen der Wörter haben in ihrem Bereich keine Bedeutung.

Wie an den Aufzählungen zu erkennen ist, wurde die ConcentriCloud in vielen Punkte eher negativ als positiv bewertet. Das lag in den meisten Punkten an einem visuell nicht ansprechenden Layout sowie der nicht intuitiven Einteilung der Schnittmengen. Ebenso war auch das Auslassen von Wörtern ein negatives Kriterium.

7.2. MultiCloud

Bei der MultiCloud wurden viele positiven Aspekte genannt. Diese werden im Folgenden aufgelistet.

- Es ist vielseitig konfigurierbar
- Das PieChart zum Vergleich von Wörtern war nützlich und anschaulich.
- Die Bedienung ist viel leichter und benutzerfreundlicher.
- Die Webentwicklung ist zukunftssicher und leicht zugänglich.
- Die Farbcodierung mit der Sättigung war sehr positiv.
- Die generierte Word-Cloud ist sehr ästhetisch.
- Die Suchfunktion wurde als gut empfunden (mit kleinen negativen Punkten)
- Die Dokumentselektion ermöglicht beliebige Schnitte und macht es dadurch einfacher und verständlicher.
- Direkt erkennbar wie oft ein Wort in einer Schnittmenge ist und wie stark es dazugehört (Sättigung, PieChart, Position).
- Brushing und Linking wurde gut umgesetzt.
- Macht Spaß.
- Das Hovern über Dokumente.
- Das Tooltip beim Hovern über ein Wort ist positiv.
- Die Farben sind gut.
- Die Platzausnutzung ist effizienter als bei ConcentriCloud.
- An jeder „Ecke“ sind Verteilungsinformationen zu finden.
- Veranschaulichung von Schnittmengen ist sinnvoller.

Jedoch wurden nicht nur positive, sondern auch einige negative Punkte genannt. Gleichzeitig wurde auch direkt über mögliche Verbesserungsmöglichkeiten diskutiert. Die negativen Punkten waren:

- Farbcodierung funktioniert nicht bei vielen Dokumenten
- Durch die Bedeutung der Position wirkt der etwas vorhandene Whitespace im Zentrum komisch.

- Bei Selektion von Dokumenten ist nicht direkt klar, welches Wort am häufigsten auftritt (keine visuelle Veränderung).
- Gesamtgröße der Dokumente lässt sich schlechter Vergleichen als bei ConcentriCloud.
- BarCharts wären wahrscheinlich besser als PieCharts.
- In allen Dokumenten auftauchende Wörter sind nicht so klar wie bei ConcentriCloud (Zentrum).
- Eine Legende von Dokumentnamen wäre eine große Erleichterung.
- Zuordnung von gegenüberliegenden Dokumenten eventuell schwierig, da sie auch in der Mitte gezeichnet werden.
- Ungünstige Dokumentkombinationen könnten die Zuordnung erschweren.
- Kleinere Bugs.

Bei den Diskussionen zu möglichen Verbesserungsvorschlägen sind einige sehr interessante Punkte gefallen. Dazu zählen die folgenden:

- Layout entsprechend der Dokumentkombinationen wählen.
- Nach Selektion von Dokumente die Wörter neu skalieren.
- BarCharts verwenden.
- Dokumentgrößen auch mithilfe eines Diagramms darstellen.
- Dokumente beliebig anordnen ermöglichen.

Mit Bezug auf die negativen und positiven Punkte im direkten Vergleich zu ConcentriCloud ist zu sehen, dass die MultiCloud eine Optimierung darstellt und so auch von den Probanden empfunden wurde.

8. Zusammenfassung und Ausblick

Die Nachteile der Visualisierungen von RadCloud und ConcentriCloud wurden optimiert und mithilfe der umgesetzten Anwendung dargestellt. Dabei wurde darauf geachtet, dass eine Möglichkeit besteht alle Wörter in der Word-Cloud darzustellen und gleichzeitig den Whitespace und die Positionen der Wörter zu optimieren.

Das gleichzeitige Optimieren von den Nachteilen der Referenzvisualisierungen hat sich als eine Herausforderung herausgestellt. Beide Faktoren beeinflussen sich gegenseitig und machen es dadurch schwer, ein gut verteiltes Layout mit gleichzeitiger Beachtung auf die Platzierung aller verfügbarer Wörter zu erzeugen. Bei der Darstellung von allen Wörtern kann der Ansatz an seine Grenzen gebracht werden. Sollen viele Dokumentwörter und Schnittwörter dargestellt werden, ist dies nicht mehr mit dem Toleranzradius möglich. Ein Kompromiss aus höherer Toleranz oder Entfernung einiger Wörter aus der Visualisierung muss geschlossen werden.

Trotz der Probleme und Beschränkungen erweist sich der Ansatz als hilfreich und verbessert die vorhanden Konzepte erheblich. Das Layout ist visuell ansprechend und macht den Anwendern Spaß. Die generierten Word-Clouds füllen den verfügbaren Platz gut aus und sind durch die Farbcodierung leicht zu verstehen. Zusätzlich unterstützen die Sättigung und die Position das Zuordnen von Wörtern zu Dokument. Die Tooltips mit dem Kreisdiagramm geben ein Überblick über die Verteilung und Häufigkeit der Wörter. Insgesamt bietet die MultiCloud Verbesserungen in fast jedem Bereich zu den Visualisierungen ConcentriCloud oder ConcentriCloud.

Mit den Anwendungsfällen wurde gezeigt, wie die Visualisierung verwendet werden kann, um mehrere Dokumente zu analysieren. Dabei kann der Fokus auf Schnittmengen, Dokumentwörter oder einer Kombination gesetzt werden. Dabei hilft die präzise Position der Wörter und die Interaktion mit der Visualisierung Wörter zu Dokumenten zuzuordnen und gleichzeitig zu vergleichen.

Ausblick

Weitere Möglichkeiten die Visualisierung zu erweitern, besteht aus einer automatisierten Zerlegung eines Dokuments. Bisher ist der vorgestellte Ansatz darauf beschränkt, mindestens zwei Dokumente zu verwenden, um die Visualisierung zu generieren. Eine Erweiterung könnte daher eine Zerlegung des Dokuments mit anschließendem Clustering der Themen sein. Dadurch würden mehrere Dokumente simuliert werden und der Ansatz könnte diese simulierten Dokumente ohne Probleme verarbeiten.

Eine weitere Möglichkeit die Visualisierung zu verbessern, besteht in dem nicht sehr performanten Positionierungsalgorithmus. Hier besteht viel Optimierungspotential, da nicht alle Pixel des Bildes betrachtet werden müssten, um ein Wort zu platzieren. Damit hergehend könnte möglicherweise eine Visualisierung mit direkter Antwort erzeugt werden. Statt für die Generierung der Form einige Sekunden zu warten, könnte dies innerhalb einer Sekunde passieren und so direkte Änderungen ermöglichen.

Die natürliche Sprachverarbeitung kann ebenso optimiert und erweitert werden. Bisher wurde NLP nur verwendet, um die grammatikalischen Strukturen herauszufinden und dementsprechend Filter anzubieten. Des weiteren kann man eine Lemmatisierung einbauen, um gleiche Wörter in unterschiedlichen Formen zu aggregieren und eine bessere Word-Cloud zu erzeugen. Eine weitere Möglichkeit ist die Benutzung von NER, um benannte Entitäten zu annotieren. Damit bestehe die Möglichkeit Word-Clouds zu generieren, die nur aus Personen und Orten bestehen.

Durch die Studie hat sich herausgestellt, dass sich Kreisdiagramme nur bedingt für vergleichende Anweisungen eignen. Daher würde es einen Vorteil bringen die radialen Elemente auszutauschen und stattdessen Balkendiagramme zu verwenden. Damit lässt sich der Vergleich der Elemente viel leichter umsetzen.

A. Aufgaben

Finden Sie eines der relevantesten Wörter aller vorkommenden Wörter

harry, hermione, dumbledore, snape, ron, malfoy, professor, ...

Finden Sie eines der relevantesten Wörter in Dokument HP5 (nicht dokumentspezifisch)

umbridge, defence

Finde Wörter die nur zu Dokument HP5 gehören

oracle, memos, probation, karkus, ...

Wie oft kommt “Harry” in HP1 vor?

1175

Suche nach dem Wort “weasley” (voldemort) (Vorwissentest)

Beobachten

Welche relevantesten Wörter gehören zu Dokument HP1, HP3, HP6

harry, ...

In welchem Dokument kommt das Wort “sirius” am meisten vor?

HP5 (Farbe + PieChart, Hovering hebt größte Zugehörigkeit auch hervor)

Welches Dokument hat die meisten Worte?

Eine Lösung (HP4)

B. Befragung

Studie zu Multidokumenten Word-Clouds

Expertenmeinung zum Thema Multidokumenten Visualisierung mithilfe von Word-Clouds. Dabei werden Experten im Bereich der Visualisierung an der Universität Stuttgart befragt. Die Befragung umfasst eine kurze Einschätzung der Vorkenntnisse mit anschließender Aufgabenstellung und einer abschließenden Bewertung.

Allgemeine Informationen

1. **Geschlecht**
   Markieren Sie nur ein Oval.
   - ◯ Männlich
   - ◯ Weiblich

2. **Alter des Probanden**
   .....

Visualisierungsfragen

3. **Wie gut kennen Sie sich im Bereich von Word-Clouds aus?**
   Markieren Sie nur ein Oval.

	1	2	3	4	5	6	7	8	9	10	
Gar nicht	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Sehr gut

4. **Wie gut kennen Sie sich im Bereich von Multidokumenten-Visualisierungen aus?**
   Markieren Sie nur ein Oval.

	1	2	3	4	5	6	7	8	9	10	
Gar nicht	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Sehr gut

5. **Welche der beiden Visualisierungen war komfortabler zu Benutzen? Warum?**
   .....
   .....
   .....
   .....
   .....

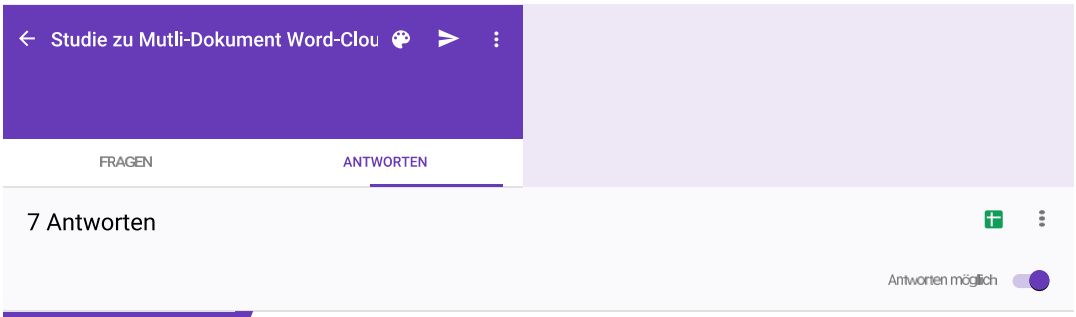
6. **Was fanden Sie positiv an ConcentriCloud?**

7. **Was fanden Sie negativ an ConcentriCloud?**

8. **Was fanden Sie gut an MultiDoc?**

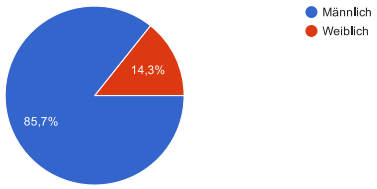
9. **Was fanden Sie schlecht an MultiDoc?**

C. Antworten



Allgemeine Informationen

Geschlecht (7 Antworten)

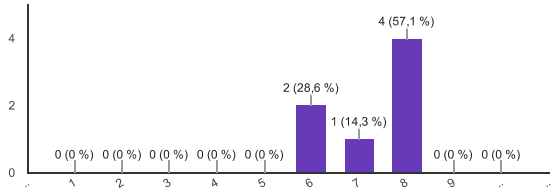


Alter des Probanden (7 Antworten)

27
28
30
31
31
31
32

Visualisierungsfragen

Wie gut kennen Sie sich im Bereich von Word-Clouds aus? (7 Antworten)



Wie gut kennen Sie sich im Bereich von Multidokumenten-Visualisierungen aus? (7 Antworten)

4 (57,1 %)

Welche der beiden Visualisierungen war komfortabler zu Benutzen? Warum? (7 Antworten)

MultiCloud (Viele Verbesserungspotentiale)
MultiCloud, nicht so viele Unterlemente, wie etwas zusammengehört (Schnitte muss man sich genauer angucken), intuitiver zu bedienen,
Hochschieben! MultiDoc. Manche Aufgaben waren nicht mit ConcentriCloud zu lösen. Verteilungsinformationen würde eher hier auftauchen.
Keine, aber Webanwendung durch Webentwicklung komfortabler durch Suchfunktion + Interaktion,
MultiDoc, übersichtlicher, konfigurierbar, BarCharts visuell, leichter zu bedienen, Benutzerfreundlich
MultiCloud, Suchfunktionen, kommt auf Aufgabenstellungen an, Explorations- und Suchfunktionen. Aber Frequenzsuche ist ConcentriCloud. PieChart für klare Aufteilung. Ausrichtung besser, Begriffe farbcodiert, Tendenz, Selektion ermöglicht Schnitte
MultiDoc, weil zooming, HTML Elemente.

Was fanden Sie positiv an ConcentriCloud? (7 Antworten)

Zentrum ist gut, vor allem bei den Harry Potter mit den Charaktern
Strukturiert durch verschiedene Bereiche, auffinden der insgesamt Häufigsten Wörter
Schnitt von allen in der Mitte, Auflistung aller Wörter mit Frequenzen, Schnitte besser zu sehen,
Frequenzen mit Einblick aufgrund von Tabelle, leicht Werte ablesen
Dokument bekommt mehr Platz für Wörter nach Größe (Pro/Contra)
Wörter im Zentrum
Border von Schnittmengen, für Leute die nicht so viel Ahnung haben -> bessere Zuordnung

Was fanden Sie negativ an ConcentriCloud? (7 Antworten)

Dokumentgröße schwierig einzuschätzen,
Wörter überspringen
Details Informationen zu finden ist aufwendig, nicht intuitiv wie die Abschnitte gelöst sind,
Gesamtgröße nicht gut, Vergleiche von gegenüberliegenden Dokument ist nicht möglich,
Kein Suchen, fehlende Sortierfunktion,
Fehlende Suchfunktion, Kombination beliebiger Dokumente nicht vorteilhaft eventuell durch Implementierung erweiterbar ohne Konzept zu verändern
absolute Vergleiche Schwierig mit Radiallayouts, kein Bezug der Größe von der Mitte zu den äußeren Bereiche, viele Layoutprobleme -> notdürftige Lösung, Schön vergleich schwierig, Suchfunktion fehlt (mit vorhandenem Vorwissen), Position der Wörter hat keine Rolle, Dokumentgröße durch Fläche schwierig, Dokumente beliebig anordnen

Was fanden Sie gut an MultiDoc? (7 Antworten)

konfigurierbar, BarCharts visuell, leichter zu bedienen, Benutzerfreundlich, webbasiert, leicht zugänglich, Farbsättigung ist auch hilfe
Schön, Suchfunktion, einfacher zu spezfizieren von Dokumentselektionen
Sättigung, wie viel ein Wort in der Schnittmenge ist und wie stark, Filtern funktioniert gut, Brushing + Linking, visuell ansprechend, macht spaß, Klickbare Dokumente, beliebige SChnitte, Hovering Dokument,
Suchfunktion, Farbcodierung (Problem bei vielen Dokumenten), Sättigung (graufarbe) wenn verteilt, Wortverteilung im Raum aber whitespace in der Mitte komisch,
Farbliche Codierung, Häufigkeit durch Kreisgröße gut, suchfunktion, Hovering gut mit Tooltip, sieht visuell ansprechender aus
Farben sind gut, Platz effizienter, Sättigung positiv erleichtert Bindung zu einem bestimmten Dokument, an jeder Ecke Verteilungsinformationen (z.B. Hovering)
Beliebiger Vergleich von Dokumenschnitten, Veranschaulichung von Schnittmengen ist sinnvoller,

Was fanden Sie schlecht an MultiDoc? (7 Antworten)

Bei Selektion welches Wort am häufigsten auftritt (keine Änderung ersichtlich),
Gesamtgröße noch schlechter (Radius sehr schlecht), Radial (PieCharts -> BarChart eventuell besser),
In allen Dokument vortauchende Wörter nicht so klar
Skalierung von Punkten besser wählen, Legende von den Dokumentnamen damit klar ist zu was welchem Dokument gehört,
Zuordnung von gegenüberliegenden Dokument eventuell schwierig, ungünstige Kombinationen erschweren die Zuordnung, Layout entsprechend wählen, absolute Vergleiche Schwierig mit Radiallayouts, BarCharts besser nach der Größe sortiert, Dokumentgröße durch Fläche schwierig, Dokumente beliebig anordnen, Whitespace in der Mitte
Dokumentnamen anzeigen, Dokumentgröße über Kreisradius nicht optimal, z.B. PieChart zum Vergleich,
Größe der Kreise nah beianeinander -> durch Linien andeuten (Linienlänge entspricht Größe, kleinere Bugs, Suchmodus nicht extra aktivieren (wirkt sehr "workaroundig")

Literaturverzeichnis

[BGN08a] S. Bateman, C. Gutwin, M. Nacenta. Seeing Things in the Clouds: The Effect of Visual Features on Tag Cloud Selections. In *Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia*, HT '08, S. 193–202. ACM, New York, NY, USA, 2008. URL http://doi.acm.org/10.1145/1379092.1379130. (Zitiert auf den Seiten 11 und 14)

[BGN08b] S. Bateman, C. Gutwin, M. Nacenta. Seeing Things in the Clouds: The Effect of Visual Features on Tag Cloud Selections. In *Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia*, HT '08, S. 193–202. ACM, New York, NY, USA, 2008. doi:10.1145/1379092.1379130. URL http://doi.acm.org/10.1145/1379092.1379130. (Zitiert auf Seite 12)

[BLB+14] M. Burch, S. Lohmann, F. Beck, N. Rodriguez, L. D. Silvestro, D. Weiskopf. RadCloud: Visualizing Multiple Texts with Merged Word Clouds. In *18th International Conference on Information Visualisation*, IV '13, S. 108–113. IEEE, 2014. (Zitiert auf den Seiten 24 und 25)

[CCP09] C. Collins, S. Carpendale, G. Penn. Docuburst: Visualizing Document Content Using Language Structure. In *Proceedings of the 11th Eurographics / IEEE - VGTC Conference on Visualization*, EuroVis'09, S. 1039–1046. The Eurographs Association & John Wiley & Sons, Ltd., Chichester, UK, 2009. URL http://dx.doi.org/10.1111/j.1467-8659.2009.01439.x. (Zitiert auf Seite 21)

[Cho03] G. G. Chowdhury. Natural language processing. *Annual review of information science and technology*, 37(1):51–89, 2003. (Zitiert auf Seite 16)

[CSBT09] Y.-X. Chen, R. Santamaría, A. Butz, R. Therón. TagClusters: Semantic Aggregation of Collaborative Tags Beyond TagClouds. In *Proceedings of the 10th International Symposium on Smart Graphics*, SG '09, S. 56–67. Springer-Verlag, Berlin, Heidelberg, 2009. URL http://dx.doi.org/10.1007/978-3-642-02115-2_5. (Zitiert auf den Seiten 20 und 21)

[CVW09] C. Collins, F. B. Viegas, M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, S. 91–98. IEEE, 2009. (Zitiert auf Seite 22)

[DMS06] T. Dwyer, K. Marriott, P. J. Stuckey. Fast Node Overlap Removal. In *Proceedings of the 13th International Conference on Graph Drawing*, GD'05, S. 153–164. Springer-Verlag, Berlin, Heidelberg, 2006. URL http://dx.doi.org/10.1007/11618058_15. (Zitiert auf den Seiten 33 und 35)

[EBC+13] A. Endert, R. Burtner, N. Cramer, R. Perko, S. Hampton, K. A. Cook. Typograph: Multiscale spatial exploration of text documents. In X. Hu, T. Y. Lin, V. Raghavan, B. W. Wah, R. A. Baeza-Yates, G. Fox, C. Shahabi, M. Smith, Q. Yang, R. Ghani, W. Fan, R. Lempel, R. Nambiar, Herausgeber, *Proceedings of the 2013 IEEE International Conference on Big Data, 6-9 October 2013, Santa Clara, CA, USA*, S. 17–24. IEEE, 2013. URL http://dx.doi.org/10.1109/BigData.2013.6691709. (Zitiert auf Seite 26)

[FSI10] J. Feinberg, J. Steele, N. Iliinsky. Wordle. In *Beautiful Visualization*, S. 37–58. O'Reilly, 2010. (Zitiert auf Seite 20)

[Hea96] C. G. Healey. Choosing Effective Colours for Data Visualization. In *Proceedings of the 7th Conference on Visualization '96*, VIS '96, S. 263–ff. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996. URL http://dl.acm.org/citation.cfm?id=244979.245597. (Zitiert auf Seite 45)

[HUA08] A. HUANG. *Similarity measures for text document clustering*. In: Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand. 2008. S, 2008. (Zitiert auf Seite 28)

[KLM15] P. Kuznecov, V. Link, E. Marbach. Vergleich und Bewertung von Word-Cloud-Visualisierungen, 2015. Universität Stuttgart: Institut für Visualisierung und Interaktive Systeme, Fachstudie Nr. 210. (Zitiert auf Seite 26)

[LBSW12] S. Lohmann, M. Burch, H. Schmauder, D. Weiskopf. Visual Analysis of Microblog Content Using Time-varying Co-occurrence Highlighting in Tag Clouds. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, S. 753–756. ACM, New York, NY, USA, 2012. URL http://doi.acm.org/10.1145/2254556.2254701. (Zitiert auf Seite 26)

[LHB+15] S. Lohmann, F. Heimerl, F. Bopp, M. Burch, T. Ertl. ConcentriCloud: Word Cloud Visualization for Multiple Text Documents. 2015. (Zitiert auf den Seiten 23 und 24)

[LRKC10] B. Lee, N. H. Riche, A. K. Karlson, S. Carpendale. SparkClouds: Visualizing Trends in Tag Clouds. *IEEE Trans. Vis. Comput. Graphics*, 16(6):1182–1189, 2010. URL http://dx.doi.org/10.1109/TVCG.2010.194. (Zitiert auf den Seiten 15 und 26)

[OSR+14] D. Oelke, H. Strobelt, C. Rohrdantz, I. Gurevych, O. Deussen. Comparative Exploration of Document Collections: A Visual Analytics Approach. *Comput. Graph. Forum*, 33(3):201–210, 2014. URL http://dx.doi.org/10.1111/cgf.12376. (Zitiert auf Seite 26)

[PMC+13] P. Pagliosa, R. M. Martins, D. Cedrim, A. Paiva, R. Minghim, L. G. Nonato. MIST: Multiscale Information and Summaries of Texts. In *Proceedings of the 2013 XXVI Conference on Graphics, Patterns and Images*, SIBGRAPI '13, S. 91–98. IEEE Computer Society, Washington, DC, USA, 2013. URL http://dx.doi.org/10.1109/SIBGRAPI.2013.22. (Zitiert auf Seite 26)

[PTT+12] F. V. Paulovich, F. M. B. Toledo, G. P. Telles, R. Minghim, L. G. Nonato. Semantic Wordification of Document Collections. *Comput. Graph. Forum*, 31(3):1145–1153, 2012. URL http://dx.doi.org/10.1111/j.1467-8659.2012.03107.x. (Zitiert auf Seite 26)

[VWF09] F. B. Viegas, M. Wattenberg, J. Feinberg. Participatory Visualization with Wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1137–1144, 2009. URL http://dx.doi.org/10.1109/TVCG.2009.171. (Zitiert auf Seite 20)

Alle URLs wurden zuletzt am 10. 12. 2015 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift