TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Fakultät Elektrotechnik Feinwerktechnik Informationstechnik

Entwicklung eines Suchalgorithmusprototypen zur Bewertung von Suchergebnissen verschiedener Kategorien

Studienarbeit im Studiengang Software Engineering

vorgelegt von

Marc Jonas Roser

Matrikelnummer 364 7316

Betreuer: Prof. Dr. Hans-Georg Hopf

Vorgelegt am 02.10.2022

Kurzdarstellung

Das Ziel der vorliegenden Studienarbeit ist es, eine bestehende Datenbank mit multimedialen Inhalten möglichst effizient nach unterschiedlichen Kriterien zu durchsuchen. Suchergebnisse sollen nach bestimmten Kriterien gewichtet, gefiltert und sortiert werden. Vorschläge für eine weiterführende Navigation auf der Suchergebnisseite sollen angeboten werden, Suchergebnisse sollen dazu nach Kontext und Wahrscheinlichkeiten gewichtet angezeigt werden. Das theoretische Fundament dieser Arbeit stellt die wissenschaftliche Betrachtung der Methoden zur Bewertung der Relevanz von Suchergebnissen dar. Die Arbeit untersucht die Möglichkeit, einen Suchbegriff so zu analysieren, dass ein Nutzer die bestmögliche Ergebnisliste bzw. zielgerichtete weiterführende Navigationsmöglichkeiten erhält. Die bestehende Anwendung „Crossload“ wird vorgestellt, um dem Leser einen Kontext zu bieten, in der sich die Entwicklung bewegt.

Abstract

The goal of this student research project is to search an existing database with multimedia content as efficiently as possible according to various criteria. Search results are to be weighted, filtered, and sorted according to certain criteria. Suggestions for further navigation on the search results page are to be offered, and search results are to be displayed weighted according to context and probabilities. The theoretical foundation of this work is the scientific consideration of methods for evaluating the relevance of search results. The work examines the possibility of analyzing a search term in such a way that a user receives the best possible list of results or targeted further navigation options. The existing application „Crossload“ is presented to provide the reader with a context in which the development takes place.

Eidesstattliche Erklärung

Hiermit versichere ich, Marc Jonas Roser, ehrenwörtlich, dass ich die vorliegende Studienarbeit mit dem Titel: „Entwicklung eines Suchalgorithmusprototypen zur Bewertung von Suchergebnissen verschiedener Kategorien“ selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Nürnberg, 02.10.2022

Marc Jonas Roser

Glossar

API

Application Programming Interface.

AWS

Amazon Web Services: Cloud Dienste gehostet von Amazon..

Crossload

Plattform zum Durchsuchen und Anhören einer umfassenden Predigtdatenbank.

Lucene

Open Source Suchbibliothek der Apache Foundation..

Mongo DB

Nicht relationale Datenbank..

SOLR

Open Source Suchframework der Apache Foundation.

Suchmaschine

Eine Anwendung, die gezielt Ergebnisse aus dem Internet für den Nutzer aufbereitet und sortiert. Englisch: „Search Engine“.

Trial-and-Error

Wiederholtes Ausprobieren ohne Erfolgsgarantie mit Änderung der Startparameter um zum gewünschten Ziel zu gelangen..

UI

Benutzeroberfläche, der Teil der Anwendung, der für den Nutzer sichtbar und nutzbar ist. Englisch: „User Interface“..

Inhaltsverzeichnis

1. Einleitung

1.1. Relevanz des Themas

Suchalgorithmen und relevante Suchergebnisse sind derzeit so relevant wie noch nie. Dabei wollen die Benutzer einer Suchmaschine in Sekundenbruchteilen Ergebnisse, die am besten zu ihrem Suchbegriff passen, ohne sich dabei viel Gedanken über die Formulierung eines solchen Begriffes zu machen. Ein Beispiel für einen solchen Algorithmus ist Google, welches seit den frühen 2000ern einen kometenhaften Aufstieg in der Welt der Suchmaschinen hinter sich hat, was anhand der erreichten Werbeeinnahmen sichtbar wird.[1]

Google ist im Vergleich zu anderen Suchmaschinen so stark verbreitet[2], dass mittlerweile sogar der Duden das Verb „googeln“ als eigenen Begriff für die Recherche im Internet führt.[3] Dabei stellt sich für die Entwicklung eigener Produkte die Frage, wie aus einem Suchbegriff, der meist nur aus wenigen Wörtern bis zu einem ganzen Satz besteht, relevante Suchergebnisse gefunden werden können. Dies würde zur Akzeptanz der Nutzer im Hinblick auf die entwickelte Funktionalität führen, da gewünschte Ergebnisse schneller und ohne großen Aufwand gefunden werden können.

1.2. Ausgangssituation

Derzeit besteht bei Crossload[4], einer Plattform zur Durchsuchung einer umfangreichen Predigtdatenbank, welche mit einer Such API auf Basis von Spring Boot und SOLR ausgestattet ist. Diese teilt auf der Suchergebnisseite die Ergebnisse nach Kategorien auf und somit können nur schwer übergreifende Suchanfragen getätigt werden. Zwar werden alle Treffer auf der gleichen Seite angezeigt, doch durch die Aufteilung nach Kategorien werden Ergebnisse gewisser Kategorien über anderen gezeigt, auch wenn niedrig positionierte Kategorien relevantere Ergebnisse enthalten.[5]

Durch eine Verbesserung der Relevanz, sowie einfacheres Suchen und weiterführende Vorschläge kann die Nutzerakzeptanz der Webseite weiter gefördert werden, da schneller bzw. überhaupt gesuchte Inhalte gefunden werden. Gefundene Inhalte werden direkt auf der Crossload angehört, weswegen dadurch die mittlere Nutzungsdauer der Seite gesteigert wird.

[1] Siehe A.1
[2] Siehe A.2
[3] Vgl. Duden [1]
[4] Siehe Crossload.org [2]
[5] Siehe A.3

1.3. Zielsetzung & Vorgehen

Das Ziel der vorliegenden Studienarbeit ist es, für die oben genannte Problemstellung einen Prototyp zur Erweiterung und Verbesserung des bisher genutzten Suchalgorithmus bei Crossload zu entwickeln.

Einleitend wird ein Einblick in die Grundlagen der Relevanz sowie mögliche Methoden und Funktionen zur Bewertung gegeben, sowie auf eine finale Auswertung der gesammelten Methoden eingegangen. Die hier erarbeiteten Grundlagen und Methoden werden im weiteren Verlauf mit in die Entwicklung einfließen. Anschließend folgt eine kurze Einleitung zu SOLR, der genutzten Search Engine von Crossload. Dieser theoretische Teil der Arbeit basiert größtenteils auf einer Literaturrecherche. Google Scholar, relevante Dokumentationen oder die einfache Google Suche stellen dazu die Grundlage dar. Die gefundenen Ergebnisse werden auf Qualität und Themenbezug geprüft.

Anhand der gegebenen Aufgabenstellung werden Anforderungen nach dem SMART Prinzip[6] erarbeitet, die als Grundlage der darauffolgenden Konzeption und Entwicklung dienen sollen. Ebenso werden bereits etablierte Tools genutzt, um die Anforderungen weiter zu verfeinern.

Für die Entwicklung wird dabei das Wasserfallmodell genutzt. Obwohl es in seinen Nachteilen gegenüber z. B. agilen Methoden überwiegt, bietet es doch wenig Aufwand um die eigentliche Entwicklung herum und stellt eine klare Struktur bereit. Diese hilft, schnell ein Produkt, oder in diesem Fall einen Prototyp, fertigzustellen. Das Modell ist auch vorteilhaft, weil die Anforderungen in ihrer Gesamtheit schon bekannt sind, beziehungsweise vor der Entwicklung sein werden und keine weiteren Faktoren hinzukommen. Die dafür verwendeten Phasen, Anforderungserhebung, Entwicklung und anschließendem Test der Anforderungen, werden in nachfolgender Abbildung verdeutlicht.

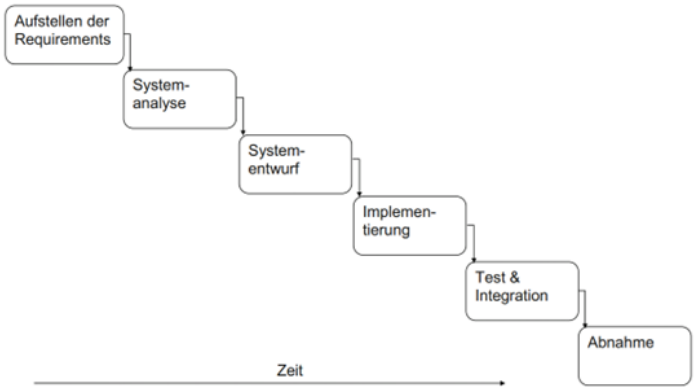

Abbildung (1.1) –Grundform eines Wasserfallmodells ohne Machbarkeitsanalyse. Nach Goll [4].

[6] Vgl. Witte 2019a, S. 67 [3]

Im Rahmen der Konzeption wird auf die bisherige Funktionalität eingegangen, um mögliche Probleme und Verbesserungspotenzial aufzudecken. Anschließend wird eine neue Berechnung des Relevanzscores mithilfe der erarbeiteten Methoden zur Berechnung der Relevanz geplant. Die Konzeption wird abgeschlossen mit der Planung der Entwicklung, mit welcher die gesammelten Anforderungen umgesetzt werden sollen.

Die Implementierung umfasst, die nach Kontext und Wahrscheinlichkeiten gewichtete und gefilterte Suche über eine Datenbank mit Datentypen verschiedener Kategorien bei der zusätzlich Vorschläge zur weiteren Navigation auf der Suchergebnisseite gegeben werden sollen.

Schlussendlich werden die Ergebnisse der Entwicklung zusammengefasst, die Anforderungen und die Implementation bewertet, sowie durch einen Ausblick, wie der Prototyp in einen produktiven Betrieb übergehen könnte, und ein Fazit abgerundet.

Im Lauf der Arbeit werden die folgenden Fragen beantwortet:

- Was ist Relevanz?
- Wie wird Relevanz in Suchmaschinen berechnet bzw. bewertet?
- Welche Anforderungen ergeben sich an ein solches Plugin?
- Mit welchen Methoden kann die aktuelle Implementierung verbessert werden?
- Welche Vorteile, Nachteile oder Hürden bringt die Implementation mit sich?

2. Theoretische Grundlagen

2.1. Relevanz

Relevanz ist allgemein beschrieben eine Beziehung zwischen einem Individuum, dem zeitlichen Rahmen, in welchem dieses eine Information benötigt und einer beliebigen Information.[1] Das bedeutet, dass Relevanz von Person zu Person unterschiedlich ist, da zum einen diverse Informationen nur zu einer bestimmten Zeit notwendig bzw. wichtig sind und der Kontext der benötigten Information sich ständig ändert.

Um zu verstehen, woher die Relevanz stammt bzw. in der Informationstechnik verwendet wird, ist es wichtig die Gewinnung von Informationen aus Objekten (*Information Retrieval*) zu verstehen. Dieser Teil der Wissenschaft beschäftigt sich mit dem präzisen Abruf von Informationen, um den das Informationsbedürfnis (*Information need*) eines Nutzers zu stillen. Das Informationsbedürfnis wird hierbei von einem idealen Inhalt gestillt, stellt also die Spezifikation eines idealen Inhalts dar. Diese Spezifikation geht aber über den reinen textuellen Inhalt der Suche hinaus. Die Relevanz ist dabei die Aktivität bzw. Praxis um diesen idealen Inhalt zu finden.[2]

2.2. Methoden zur Bewertung von Relevanz

Eine Suchmaschine gibt nach Anfrage Webseiten sortiert nach der Relevanz der Ergebnisse abhängig zum gegebenen Suchbegriff des Nutzers. Die Schwierigkeit dabei ist die Bestimmung der Relevanz für eine beliebige Website. Die dafür genutzten Funktionen und Methoden werden allerdings von den Unternehmen geheim gehalten, um einen Missbrauch ihrer Suchmaschine zu verhindern. Dennoch sind die am häufigsten genutzten Merkmale bekannt und in einigen wissenschaftlichen Arbeiten untersucht worden.[3] Da Moderne Suchmaschinen nutzen dutzende oder gar hunderte verschiedener Methoden um Features um die Relevanz der verfügbaren Suchergebnisse zu bewerten, wird im folgenden nur auf einige bekannte Methoden eingegangen.

Zur Einfachheit wird von der Webapplikation Crossload abstrahiert und stattdessen Beispiele aus der Internetsuche verwendet, welche zum Beispiel mit Google, Bing, Ecosia oder anderen Suchmaschinen üblich ist.

[1] Vgl. Bookstein S. 1 [5]
[2] Vgl. Manning, Raghavan, Schütze [6]
[3] Vgl. Zaragoza, Najork, S. 1 [7]

2.2.1. Textuelle Relevanz

Das einfachste Merkmal für die Bewertung der Relevanz ist den kompletten Inhalt nach der textuellen Relevanz zu bewerten. Da natürliche Sprache, die meist für Suchergebnisse genutzt wird, generell ungenau ist, wird mit sogenannten „Matching Functions“ versucht auch ungefähre Übereinstimmungen in einem Fließtext zu finden. Einige der verwendeten Funktionen um die textuelle Relevanz zu bewerten sind dabei:[4]

- Die Anzahl der Treffer für den Suchterm oder Abwandlungen
- Position des Suchterms (früheres Vorkommen, z. B. im Titel)
- Seiten Struktur (für Webseite: Ist der Term eine Überschrift o. ä.)
- Grafisches Layout (für Webseiten: Ist der Term z. B. farblich markiert)
- Levenshtein Distanz[5] (die minimale Anzahl an Operationen, um eine Zeichenkette in eine andere umzuwandeln)

2.2.2. Relevanz durch Attribute

Des Weiteren ist es auch möglich den durchsuchten Objekten Attribute zuzuweisen, um für Schlagwörter relevantere Ergebnisse zu erlangen. Diese können entweder von Nutzern selbst bestimmt werden, wie z. B. bei der Website Flickr[6], um Bilder für bestimmte Themen höher werten zu lassen oder werden von Algorithmen aufgrund von Bilderkennung automatisch zugewiesen.

Ein Beispiel hierfür ist Google, welches eine frei verwendbare Machine Learning API[7] oder eine direkte Integration, in die Google Fotos App anbietet, welche die gemachten Bilder automatisch in verschiedene Kategorien aufteilt.[8]

Möglich gefundene Ergebnisse können auch durch Existenz oder Nichtvorhandensein eines Attributs höher gewichtet werden. Dadurch können zum Beispiel bereits aufbereitete Ergebnisse eine höhere Relevanz erhalten. [9]

[4] Vgl. Zaragoza, Najork, S. 1 [7]
[5] Vgl. Levensthein [8]
[6] Vgl. Liu et.al., S. 1-3 [9]
[7] Vgl. Google ML Dokumentation [10]
[8] Vgl. Google Fotos [11]
[9] Siehe Crossload Search API [12]

2.2.3. Hyperlink Relevanz

Für Suchergebnisse im Internet oder andere miteinander verlinkte Seiten, wie z. B. in internen Dokumentationsseiten, Wikis o. ä., können auch die Hyperlinks, die auf eine andere Seite verlinken genutzt werden die Relevanz eines Ergebnisses zu bestimmen. Ein Hyperlink besteht hierbei aus dem angezeigten Text auf der Quellseite und einem Link zur Zielseite oder auf einen bestimmten Abschnitt dergleichen. Dies ist aber kein automatischer Prozess, sondern jeder Link wird von Menschen gesetzt. Aus diesem Grund kann man hier von „menschlicher Intelligenz“ sprechen.[10]

Um einen Treffer höher zu gewichten, ist eine Option die Anzahl an Verlinkungen auf eine Seite zu zählen und absteigend zu sortieren.[11] Alternativ kann der angezeigte Linktext noch zusätzlich als eine Art Attribut (2.2.2) oder erweiterte textuelle Referenz (2.2.1) gesehen werden, der dann bei der Auswertung einer Suche mitverwendet wird.[12]

2.2.4. Relevanz durch Nutzerverhalten

Um unabhängiger von manuellem Verlinken zwischen Seiten zu werden, haben bekannte Suchmaschinen auch Möglichkeiten entwickelt, die Anzahl der „erfolgreich“ gefundenen Treffer zu zählen und als relevanter zu gewichten. Im Umfeld einer Internetsuche wäre der „erfolgreich“ gefundene Treffer ein Klick auf die entsprechende Website. Diese können entweder live oder durch Auswertung von Log Dateien analysiert werden. Andere Wege um die Anzahl an Besuchen auf einer Website zu messen, umfassen Tracking Methoden, Toolbars oder Werbung. Diese Methode ist überaus erfolgreich, da hier von einer Art Schwarmintelligenz ausgegangen wird, die Nutzern für die gleiche Suche Ergebnisse anzeigt, die schon viele Benutzer davor angeklickt haben. [13]

2.2.5. Performance

Da Suchmaschinen dem Nutzer eine bestmögliche Benutzererfahrung, auch bekannt als User Experience, ermöglichen wollen, sollen die gefundenen Webseiten dies bieten. Eine Möglichkeit dies zu messen ist die Performance einer Website. Dies umfasst die Ladegeschwindigkeit, Speicherverbrauch und benötigte Leistung um die Seite komplett anzuzeigen. Da dies nicht für x-Millionen Treffer bei jeder Suchanfrage getestet werden kann, werden mögliche Suchtreffer vorher indiziert und nach der Performance untersucht. Dadurch entsteht ein Performance-Score, welcher dann für die Relevanz verwendet werden kann.[14]

[10] Vgl. Zaragoza, Najork, S. 2 [7]
[11] Vgl. Marchiori [13]
[12] Vgl. Page, Brin, Motwani und Winograd [14]
[13] Vgl. Joachims, Radlinski, S. 1 [15]
[14] Vgl. Manning, Raghavan, Schütze [6]

2.3. Auswertung der Relevanz von Suchergebnissen

Um letztendlich Ergebnisse mit der höchsten Relevanz zu erhalten wird meist eine Kombination aus mehreren der o.g. Methoden benutzt, um die komplette Relevanz für einen Treffer zu bewerten. Die Herausforderung dabei ist die genaue Gewichtung der einzelnen Methoden um die Relevanz eines Treffers optimal zu bewerten. Für jeden Treffer wird dann ein Relevanzscore berechnet, der sich aus den einzelnen Methoden zusammensetzt. Nach diesem Score wird in einer Liste absteigend sortiert, um das relevanteste Ergebnis als erstes Element zu erhalten.

Sollte sich der Score eines Treffers in der Relation zu anderen Wertungen weit absetzen, kann dieser Treffer dem Nutzer auch direkt vorgeschlagen werden.[15] Dieses Vorschlagen von Ergebnissen kann bereits bei der Eingabe einer Abfrage geschehen, durch sogenannte „Search Completion“.[16]

Für kleinere Anwendungen ist hierbei meist ein manuelles Einstellen nach einem Trial-and-Error Verfahren notwendig, bei denen einige wenige Methoden unterschiedlich gewichtet werden. Dies wird dann von Zeit zu Zeit wiederholt, wenn neue Erkenntnisse aus Tests oder dem produktiven Betrieb zurückkommen.[17]

Große Suchmaschinen Nutzen hierfür allerdings wie bereits erwähnt hunderte Methoden und evaluieren deren Erfolg im produktiven Betrieb durch proprietäre statistische Methoden.[18]

2.4. SOLR

SOLR ist von Crossload verwendete Such Engine, die als Web Schnittstelle dient, um auf einer Datenmenge Suchanfragen mit Apache Lucene auszuwerten.[19]

Apache Lucene, oder auch kurz Lucene genannt, ist eine mächtige Suchbibliothek, die plattformunabhängig von verschiedenen bekannten Apps, wie z. B. Netflix eingesetzt wird. Lucene nutzt für die Indizierung zu durchsuchender Dokumente Textfelder, wie z. B. „title“ für den Titel eines Dokumentes, um sowohl den Inhalt als Volltext sowie auch die Attribute durchsuchen zu können. [20]

SOLR benutzt die hier die Indexing Funktionen von Lucene, um in Echtzeit alle verfügbaren Dokumente zu indizieren um bei einer Suche nur den Index durchsuchen zu müssen. Mit Apache Zookeeper wird dann eine API zur Verfügung gestellt, welche Synchronisierung, Namensregister und die Verteilung der Konfiguration bereitstellt. Inhalte werden anhand

[15] Vgl. Turnbull, Berryman, S. 225-228 [16]
[16] Vgl. Turnbull, Berryman, S. 206-218 [16]
[17] Vgl. Zaragoza, Najork, S. 3 [7]
[18] Vgl. Taylor, Zaragoza, Craswell, Robertson, Burges [17]
[19] Siehe Apache SOLR [18]
[20] Siehe Apache Lucene [19]

von Boostingmechanismen höher oder schlechter bewertet. Als Entwickler gibt man hierfür mögliche Textfelder an, auf denen SOLR automatisch ein Textmatching anwendet (2.2.1). Ebenso ist es möglich eigene Boostingmechanismen zu erstellen, wobei hier dann in Java entwickelt wird.[21]

2.5. Crossload

Crossload ist eine deusche Predigtdatenbank, deren Ziel es ist, mit modernen Technologien und einem ansprechendem User Interface (UI) den Zugang zu Predigten und anderem christlichen Material zu vereinfachen. Hierzu werden teils Predigten aus anderen System importiert, teils Autoren angefragt, welche dann regelmäßig ihre eigenen Predigten selbstständig hochladen. Dadurch sind sowohl ältere Predigten, etwa von Martin Luther, als auch Predigten zu aktuellen Themen und Weltgeschehen verfügbar. Zudem gibt es Schnittstellen zu christlichen Verlagen oder Webseiten wie CLV[22] oder Evangelium 21[23].[24] Auf Crossload gibt es derzeit Predigten mit und ohne Video, Bücher, Bilder, Musik, Hörbücher und andere bzw. noch nicht kategorisierte Inhalte.

Technisch ist Crossload wie folgt aufgestellt:

- **Frontend:** UI entwickelt mit Angular zum Durchsuchen der Datenbank und direktem Streaming der Predigten. Für die Analyse und Statistiken, welche Seiten besucht, welche Inhalte angehört und welche Suchanfragen abgegeben wurden, wird Matomo verwendet. Matomo, ein Open-Source Pendant zu Google Analytics, enthält auch Statistiken zur durchschnittlichen Dauer eines Besuches.[25]
- **Suche:** Auf SOLR basierte REST API mit allen veröffentlichten Inhalten und anderen Metadaten. Wird benutzt, um Last vom redaktionellen Backend zu nehmen.
- **Redaktion:** Aufbereitung und Anlegen von Inhalten verschiedenster Kategorien und anderer Metadaten.
  - **Angular UI**: Redaktionelles Backend zum Pflegen aller Daten von Crossload.
  - **Node.js RESTFUL API**: Schnittstelle zwischen der UI, der Datenbank und AWS.
  - **AWS**: Speicherung von Dateien (Audio, Video, Bilder).
  - **Mongo DB**: Datenbank zur Verwaltung und Speicherung aller Daten.

[21] Siehe Apache SOLR [18]
[22] Siehe CLV [20]
[23] Siehe Evangelium 21 [21]
[24] Vgl. Pfleiderer, Crossload [2]
[25] Siehe Matomo [22]

3. Anforderungen und Problemanalyse

Im folgenden Kapitel sollen alle wesentlichen Funktionen der Erweiterung dargestellt werden. Da es sich um ein kleines Projekt mit einem abgestecktem Rahmen handelt und es kein Team gibt, welches die Entwicklung durchführt, wird das Wasserfallmodell für diese Entwicklung verwendet.

3.1. Vorgehensweise

Zur Erfassung der Anforderungen bzw. Requirements Engineering werden User Stories benutzt.

Diese sind bekannt aus agilen Softwareentwicklungsmodellen, wie z. B. Scrum, entstanden aber durch praktische Erfahrungen in der Softwareentwicklung. Konzeptioniert wurden sie von Dr. Ivar Jacobsen[1] und Ron Jeffries[2]. Mithilfe einfacher Sprache wird aus der Sicht des Stakeholders das Ziel einer Story in einem kurzen Satz zusammengefasst. Anschließend wird dieses Ziel begründet, um die Wichtigkeit und Existenzberechtigung der Story zu begründen. User Stories sind dabei auch Anforderungen nach dem SMART Prinzip[3], da diese nur einen sehr kleinen abgesteckten Teilbereich einer Funktionalität enthalten. Dadurch sind sie einfacher schätzbar, umsetzbar und testbar.

Anhand der ermittelten User Stories werden nach der Entwicklung Akzeptanztests durchgeführt, um den Erfolg des Endproduktes objektiv zu bewerten.

3.2. User Stories

Die Anforderungen umfassen alle Aktionen, welche der Nutzer in der Anwendung durchführen will. Ziel aller Anforderungen ist die übergreifende Suche über mehrere Kategorien nach effizient nach mehreren Kriterien zu durchsuchen und zu bewerten. Anhand dieses Ziels werden User Stories entwickelt, in denen ein Nutzer und andere Personen ihre Anforderungen an das zu entwickelnde Produkt stellen.

Nichtfunktionale Anforderungen werden bei dieser Anforderungserhebung nicht beachtet, da es sich hierbei um die Erweiterung einer bestehenden API handelt und Punkte wie Benutzerfreundlichkeit und User Experience hierbei wenig relevant sind, beziehungsweise das Entwicklungsumfeld durch die bereits bestehende Anwendung vorgegeben ist. Die Priorität der einzelnen User Stories ergibt sich aus der unten gegebenen Reihenfolge.

[1] Vgl. Jacobson, Spence, Kerr 2016 [23]
[2] Vgl. Ron Jeffries [24]
[3] Vgl. Witte 2019a, S. 67 [3]

Ich, als Benutzer, will ...

- ... für ein gegebenes Suchkriterium relevante Suchergebnisse über mehrere Kategorien hinweg erhalten, damit mit einer einzelnen Suche nur eine geringe Teilmenge der Datenbank angezeigt wird.
- ... die erhaltenen Suchbegriffe nach Kontext und Wahrscheinlichkeit gewichtet erhalten, damit diese im späteren Verlauf sortiert werden können. Der Kontext ergibt sich aus möglichen Schlagworten, die im Suchbegriff verwendet worden, womit z. B. eine Kategorie, ein Attribut eines Ergebnisses oder höher gewertet wird. Beispiele wären:
  - ... der Titel eines Buches wird „relativ“ genau als Suchbegriff eingegeben, folglich wird dieses Buch stärker gewichtet.
  - ... der Suchbegriff enthält den Term „Video“, folglich werden alle Videos priorisiert.
- ... die erhaltenen Suchbegriffe anhand des errechneten Gewichts absteigend sortiert zurückgegeben werden, damit das relevanteste Suchergebnis auf der Suchergebnisseite ganz oben steht.
- ... ein mit hoher Wahrscheinlichkeit gesuchte Suchergebnis als Vorschlag angezeigt bekommen, damit auf der Suchergebnisseite eine schnelle Navigation zu diesem Ergebnis möglich ist. Dabei soll nur ein Vorschlag angezeigt werden, wenn er der einzige mit einer hohen Wahrscheinlichkeit, in der Suchergebnisliste vorhanden durch das vorher errechnete Gewicht, und dieses Gewicht eine gewisse Schwelle überschreitet. Dadurch sollen verhindert werden, dass von ähnlich relevante Suchergebnisse nur eines vorgeschlagen wird und ein Suchergebnis vorgeschlagen wird, welches für das gegebene Suchkriterium irrelevant ist.

4. Konzeption

Bevor mit der Entwicklung des Prototyps gestartet werden kann, geht die Planung und Konzeption der Erweiterung voraus. Der Entwurf einer Software ist die Basis für jede Entwicklung. Anfangs wird die momentane Anwendung auf bereits implementierte Funktionalität überprüft und schließlich mithilfe der erarbeiteten Methoden zur Bewertung der Relevanz auf Grundlage der gesammelten Anforderungen verbessert.

4.1. Bisherige implementierte Funktionalität

Bei Crossload werden verschiedene Typen bzw. Kategorien von Inhalten in der von SOLR indizierten Datenbank über eine Spring Boot Anwendung an das Webfrontend zur Verfügung gestellt. Der initiale und derzeit implementierte Gedanke dabei ist, die Inhalte auch in diesen Kategorien zu übertragen und in fester Reihenfolge anzuzeigen. Diese Vorgehensweise hat jedoch einige Nachteile:

- **Relevanz:** Der möglicherweise relevanteste Inhalt wird nicht als erstes angezeigt, da dessen Kategorie relativ weit unten angezeigt wird.
- **Übersicht:** Es ist schwer für den Nutzer eine Übersicht über alle gefundenen Inhalte zu erlangen.
- **User Experience:** Höchstwahrscheinliche Treffer (90-100 % Trefferwahrscheinlichkeit) werden nicht direkt vorgeschlagen.

Diese Nachteile sollen im Verlaufe der Entwicklung verbessert werden. Ebenso sollen auch die bisher genutzten Methoden zur Berechnung der Relevanz verbessert werden. Diese umfassen derzeit:

- Text Matching auf verschiedene Textteile und Attribute. Hier werden verschiedene Attribute in 3 Kategorien (hoch, mittel, niedrig) wie folgend bewertet:
  - **Hoch:** Titel, Serie, Thema, Autor
  - **Mittel:** Untertitel, Schlagwörter, Kategorie, Thema
  - **Niedrig:** Verlag, Standort, Dateiname, Speech to Text, Mitschrift, Suchsnippet
- Matching des Suchterms zu einem Bibelvers.
- Oder falls kein Suchterm gegeben ist, werden Inhalte mit Video oder Bild höher bewertet.

- Filter für mitgegebene Query Parameter: Kategorie, Serie, Event, Thema, Jahreszahl oder Dauer. Inhalte, die nicht zu diesem Filter passen, werden komplett aussortiert.

4.2. Verbesserungen für relevantere Inhalte

Grundsätzlich findet die Anwendung bereits passende bzw. relevante Inhalte durch das Matching der verschiedenen Attribute (2.2.2) und Textteile (2.2.1). Ebenso das Matching bezüglich des Bibelverses oder des initialen Boosting über ein vorhandenes Video oder Bild führt bereits zum gewünschten Ergebnis und eine Änderung würde hier keinen nennenswerten Mehrwert bieten.

Dennoch gibt es einige Ideen, relevantere Inhalte für den Nutzer herauszugeben: zusammengeführte Listen, ein Schlagwortabgleich auf die Kategorien und Vorschläge zur weiteren Navigation.

4.2.1. Zusammengeführte Liste

Als oberstes Ziel wird die Liste der gefundenen Inhalte, momentan aufgespalten in die verschiedenen Kategorien wie z. B. Bild, Video, Predigt, Buch, etc., in eine große Liste überführt. Dadurch können relevante Inhalte, die bisher durch die vordefinierte Sortierung der Kategorien auf der Suchseite nicht als erste aufgelistet wurden, an der Stelle angezeigt werden, an die der Nutzer sie erwartet.

Damit der Nutzer dennoch sieht, welcher Inhalt welche Kategorie, wird anschließend zu der ausführlichen Version des Ergebnisses ein kleiner Text mit dessen Kategorie hinzugefügt. So geht die bisherige Funktionalität nicht komplett verloren und der Nutzer erhält die relevantesten Inhalte direkt an erster Stelle und sieht sofort dessen Kategorie.

4.2.2. Schlagwortabgleich

Eine Möglichkeit, die Relevanz der Suchergebnisse im Sinne der Aufgabenstellung zu verbessern, wäre es, anhand des Suchbegriffes herausfiltern, ob z. B. ein Schlagwort wie „Video“ oder „Bild“ verwendet wurde und somit relevante Inhalte dieser Kategorie höher zu gewichten.

Dafür müssten relevante Schlagwörter ermittelt werden und auch in allen möglichen Varianten untersucht werden, um ein hilfreiches Matching zu erhalten, welches dann anhand dem Attribute „Hauptkategorie“ nachvollzogen werden kann. Eine gewisse Menge an Varianten kann vordefiniert werden, um einen Großteil der Anfragen korrekt abzufangen. Um letztendlich aber eine mehr und mehr vollständige Menge an Varianten und Suchbegriffen zu

erhalten müssen die abgegebenen Suchabfragen untersucht werden. Diese können aber mit Matomo untersucht werden und mit der Zeit angepasst werden.[1]

Für den Start wären folgende Schlagwörter für die vorhandenen Kategorien denkbar:[2]

- **Predigten (mit Video):** Video, Film, Stream, Live, Livestream.
- **Predigten (mit und ohne Video):** Predigt, Vortrag, Mahnwort.
- **Bücher:** Buch, Bücher, Taschenbuch, Sammelband, Reader, Druck, Bestseller.
- **Bilder:** Bild, Darstellung, Zeichnung, Aufnahme, Foto, Fotografie.
- **Musik:** Song, Melodie, Hymne, Stück, Gesang, Klavier, Musik, Orchester.
- **Hörbücher:** Hörbuch, Hörbücher, Audiobook.
- **Sonstige:** Sonstige.

4.2.3. Vorschläge für weitere Navigation

Optimalerweise gibt der Nutzer eine Suchanfrage ein, zu der ein Inhalt eine sehr hohe Relevanz hat und alle anderen Inhalte eine recht niedrige. Sollte dies der Fall sein, so könnte dieser Inhalt in einer Vorschlagsbox über der Ergebnisliste angezeigt werden, damit der Nutzer visuell sieht, dass dies der Inhalt ist, den er höchstwahrscheinlich sucht. Eine Berechnung hierfür ist nicht klar definiert, als ersten Versuch wird überprüft, ob der berechnete Score des relevantesten Inhalts mindestens doppelt so groß ist, wie der des nächsten Inhalts. Dieses Vorgehen muss aber in der Entwicklung und im produktiven Betrieb weiter geprüft werden, um dieses Vorgehen weiter zu verfeinern.

Eine mögliche Schwachstelle hierbei könnten sehr relevante Inhalte direkt am Anfang sein, bei denen die darauf folgenden Inhalte im Vergleich irrelevant sind. Somit könnten auch beide Inhalte vorgeschlagen werden, was aber aus Gründen der Nutzerfreundlichkeit nur auf einen minimiert wird. Dafür müsste die ganze Liste, oder ein Teil z. B. die Top 10, auf die durchschnittliche Relevanz geprüft werden und Inhalte, die sich stark nach oben von diesem Durchschnitt unterscheiden, als Vorschläge genommen werden.

Für die Implementierung wird hierbei zuerst geprüft, ob ein Inhalt vorgeschlagen werden kann. Falls dies nicht zutrifft, wird anschließend mit der Prüfung des Durchschnitts fortgefahren.

[1] Siehe 2.5 [22]
[2] Siehe Duden [25]

5. Entwicklung des Prototyps

Die Umsetzung des Prototyps erfolgt in mehreren Schritten. Zu Beginn wird wie in 4.2.1 beschrieben, die Liste aller Ergebnisse zusammengeführt und absteigend nach der Relevanz sortiert. Anschließend wird der Schlagwortabgleich (4.2.2) implementiert, indem konfigurierbar die Liste der möglichen Synonyme mit dem Suchterm abgeglichen wird und die Ergebnisse geboostet werden, wenn der Suchterm ein Synonym enthält. Zuletzt werden die resultierenden Inhalte nach einem möglichen Vorschlag wie in 4.2.3 beschrieben dem Ergebnis hinzugefügt.

5.1. Zusammengeführte Liste

5.2. Schlagwortabgleich

5.3. Vorschläge für weitere Navigation

6. Evaluation

7. Diskussion der Ergebnisse

7.1. Abgleich der Ergebnisse mit den gestellten Anforderungen

7.2. Identifizierte Probleme

7.3. Bewertung

8. Ausblick & Fazit

A. Anhang

I. Bilder

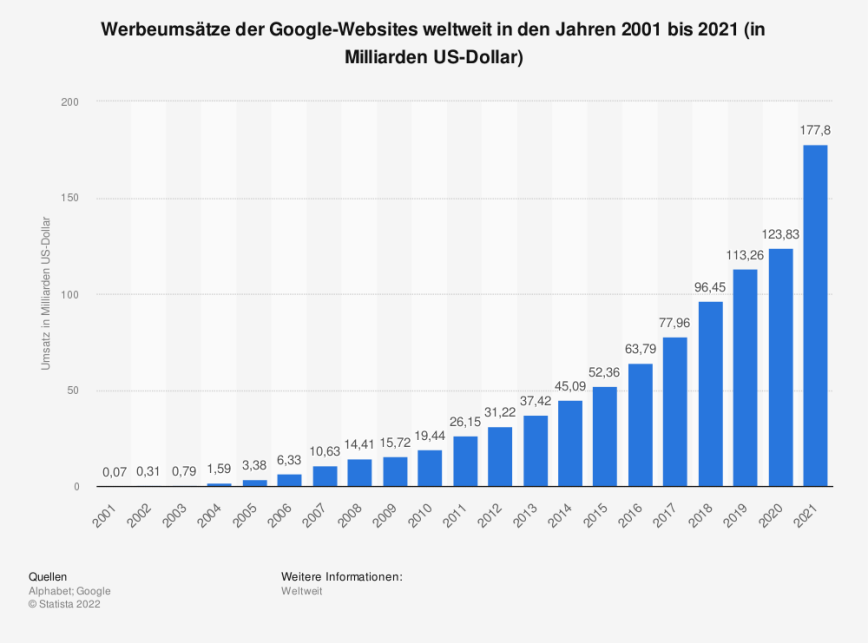

Abbildung (A.1) –Werbeumsätze Google Websites [26]

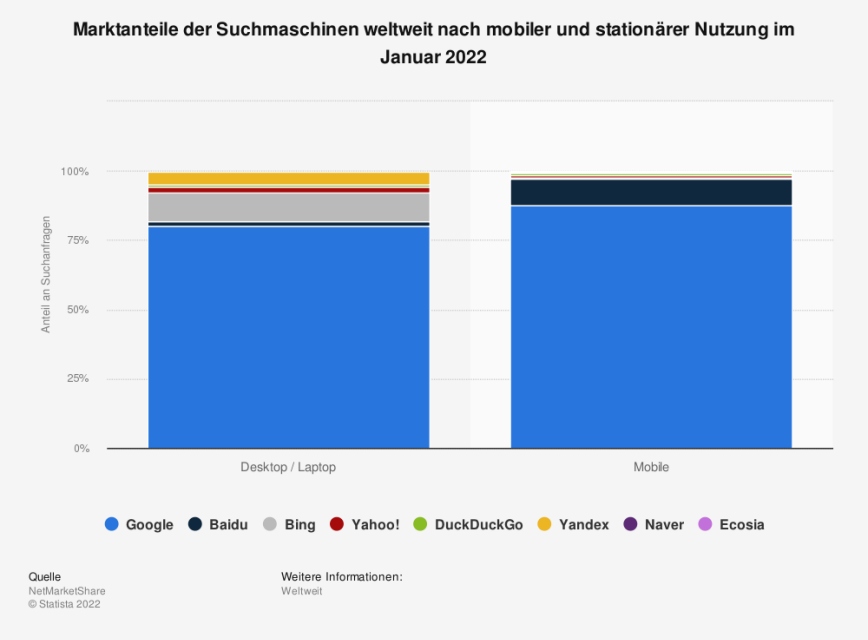

Abbildung (A.2) –Marktanteil Google [27]

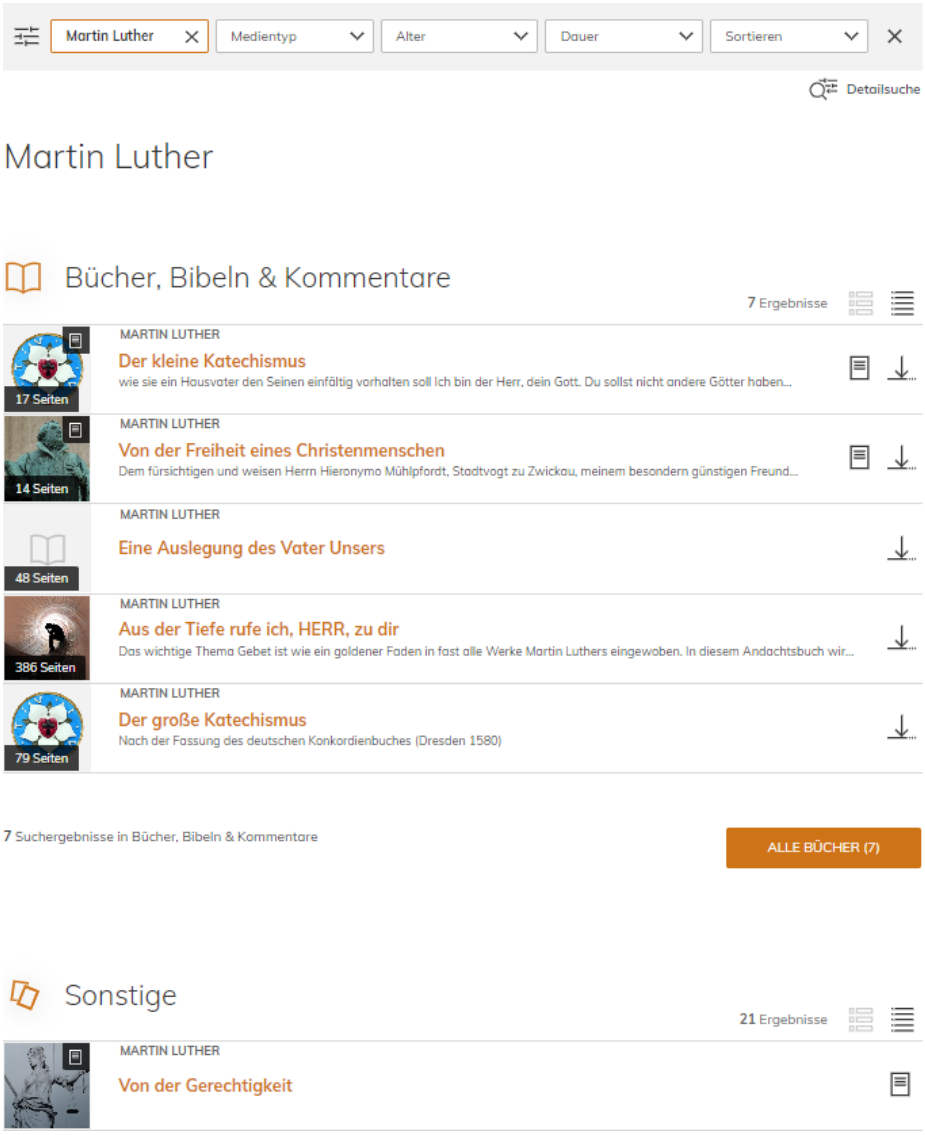

Abbildung (A.3) –Crossload [2]

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Duden, "Duden | Wie schreibt man „googeln"? | Rechtschreibung." https://www.duden.de/rechtschreibung/googeln, 2022.

[2] D. Pfleiderer, "CROSSLOAD." https://crossload.org/info/ueber, Sept. 2022.

[3] F. Witte, *Testmanagement und Softwaretest*. Wiesbaden: Springer Fachmedien, 2016.

[4] J. Goll, *Methoden und Architekturen der Softwaretechnik*. Wiesbaden: Vieweg+Teubner Verlag, 2011.

[5] A. Bookstein, "Relevance," *Journal of the American Society for Information Science*, vol. 30, pp. 269–273, Sept. 2007.

[6] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York: Cambridge University Press, 2008.

[7] H. Zaragoza and M. Najork, "Web Search Relevance Ranking," in *Encyclopedia of Database Systems* (L. Liu and M. T. Özsu, eds.), pp. 4650–4655, New York, NY: Springer New York, 2018.

[8] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet Physics Doklady*, vol. 10, pp. 707–710, Soviet Union, 1966.

[9] D. Liu, X.-s. Hua, M. Wang, H.-j. Zhang, and L. Yang, "WWW 2009 MADRID! Track: Rich Media / Session: Tagging and Clustering Tag Ranking $_*$," 2009.

[10] G. Developers, "Image labeling | ML Kit." https://developers.google.com/ml-kit/vision/image-labeling, 2022.

[11] G. Photos, "Google Photos." https://www.google.com/photos/about/, 2022.

[12] Crossload, "CROSSLOAD / Backend / solr-search · GitLab." https://gitlab.crossload.org/crossload/backend/solr-search, 2022.

[13] M. Marchiori, "The quest for correct information on the Web: Hyper search engines," *Computer Networks and ISDN Systems*, vol. 29, pp. 1225–1235, Sept. 1997.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab / Stanford InfoLab, Nov. 1999.

[15] T. Joachims and F. Radlinski, "Search Engines that Learn from Implicit Feedback," *Computer*, vol. 40, pp. 34–40, Aug. 2007.

[16] D. Turnbull and J. Berryman, *Relevant Search: With Applications for Solr and Elasticsearch*. Manning, 2016.

[17] M. Taylor, H. Zaragoza, N. Craswell, S. Robertson, and C. Burges, "Optimisation methods for ranking functions with multiple parameters," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management - CIKM '06*, (Arlington, Virginia, USA), p. 585, ACM Press, 2006.

[18] Solr, "Apache Solr Reference Guide :: Apache Solr Reference Guide." https://solr.apache.org/guide/solr/latest/, 2022.

[19] A. Lucene, "Welcome to Apache Lucene." https://lucene.apache.org/index.html, 2022.

[20] CLV, "CLV | Bücher, die weiterhelfen." https://clv.de/, 2022.

[21] Evangelium21 e.V., "Startseite | Evangelium21." https://www.evangelium21.net/, 2022.

[22] Matomo, "Matomo - The Google Analytics alternative that protects your data." https://matomo.org/, 2022.

[23] I. Jacobson, I. Spence, and B. Kerr, "Use-Case 2.0: The Hub of Software Development," *Queue*, vol. 14, pp. 94–123, Jan. 2016.

[24] R. Jeffries, "Essential XP: Card, Conversation, Confirmation." https://ronjeffries.com/xprog/articles/expcardconversationconfirmation/, 2022.

[25] D. Synonyme, "Synonyme online finden | Synonymwörterbuch | Duden." https://www.duden.de/synonyme, 2022.

[26] Alphabet, "Werbeumsätze der Google-Websites weltweit in den Jahren 2001 bis 2021." https://de.statista.com/statistik/daten/studie/75181/umfrage/werbeumsatz-der-google-websites-seit-2001/, Feb. 2022.

[27] NetMarketShare, "Marktanteile der Suchmaschinen - Mobil und stationär 2022." https://de.statista.com/statistik/daten/studie/222849/umfrage/marktanteile-der-suchmaschinen-weltweit/, Feb. 2022.