UNIVERSITÄT
LEIPZIG

Universität Leipzig

Abteilung Automatische Sprachverarbeitung

Fortgeschrittene Methoden des Information Retrieval

Bericht zum Laborprojekt

Jeremy Puchta – Jonathan Lange – Ali Al-Ali

14. März 2019

Inhaltsverzeichnis

1 Einleitung

Der vorliegende Bericht erläutert den Aufbau sowie die Funktionsweise der Suchmaschine *Historical News Search*, welche im Rahmen des Laborprojektes innerhalb des Moduls *Fortgeschrittene Methoden des Information Retrieval* im Wintersemester 2018 / 2019 erstellt wurde. Ziel des Laborprojektes war es, die vermittelten Vorlesungsinhalte zu vertiefen und diese bei der Erstellung einer domänenspezifischen Suchmaschine umzusetzen. Die vorgestellte Suchmaschine dient der Exploration von historischen Nachrichten und Zeitungsartikeln. Als Anwendungsfälle für eine solche Suchmaschine sind verschiedene Szenarien denkbar. Ein Nutzer kann beispielsweise nach der Berichterstattung zu Personen, Gruppen, Orten oder bestimmten historischen Ereignissen suchen. Eine weitere Möglichkeit der Nutzung stellt die Ahnenforschung dar, bei welcher ein Nutzer in Zeitungsberichten nach Informationen über seine Vorfahren suchen und somit mehr über seine Familienhistorie erfahren kann.

Im folgenden Kapitel wird zunächst der verwendete Technologiestack eingehend beschrieben. Anschließend erfolgt die Vorstellung des verwendeten Datensatzes und die Analyse dessen mit Methoden der deskriptiven Statistik. Die Architektur und der Aufbau der Suchmaschine werden in Kapitel 4 näher dargestellt. Dazu zählt insbesondere die Erstellung des Indizes sowie die Umsetzung des Suchprozesses über alle beteiligten Komponenten des Systems. Kapitel 5 stellt das User Interface sowie die dahinter ablaufenden Prozesse vor. In Kapitel 6 werden zunächst die Evaluationsmethodik sowie die Resultate der Evaluation erläutert. Weiterhin wird erklärt, welche Anpassungen vorgenommen wurden, um die Effektivität der Suchmaschine und die Nutzererfahrung zu verbessern. Abschließend erfolgt in Kapitel 7 eine Zusammenfassung des Projektes und es wird ein Ausblick in Bezug auf eine Weiterentwicklung der Suchmaschine geliefert.

2 Technologiestack

Zur Umsetzung des Laborprojektes wird die Open-Source-Suchmaschine *Elasticsearch* verwendet. *Elasticsearch* ist in Java geschrieben und basiert auf der Java-Bibliothek *Lucene*. Außerdem ist *Elasticsearch* gebaut für den Einsatz auf verteilten Systemen und die Echtzeitverarbeitung von großen Datenmengen, was unter anderem zur Volltextsuche eingesetzt wird und *Elasticsearch* somit zu einer ausgezeichneten Option für die Umsetzung einer Dokumentensuchmaschine macht. *Elasticsearch* stellt sämtliche Funktionen über eine programmiersprachenunabhängige REST-Schnittstelle zur Verfügung. Um die Dokumente durchsuchbar zu machen, legt *Elasticsearch* die Datenstruktur des *Invertierten Index* an, in welcher für jeden Term gespeichert wird in welchem Dokument dieser auftritt.

Im Backend des Systems kommt das Python-Framework *Flask* zum Einsatz. Bei *Flask* handelt es sich um ein Mikroframework, welches eine einfache, jedoch ebenfalls erweiterbare und robuste Möglichkeit bietet APIs zu erstellen. Im Frontend wird mit *Angular* ein Framework verwendet, welches Entwickler durch sein Komponentensystem darin unterstützt modularen Quellcode zu

produzieren, der sich einfach warten lässt. Außerdem bringt es bereits viele Funktionen, die häufig im Frontend zum Einsatz kommen mit, wie zum Beispiel Routing. Zur Vereinfachung des Deployments und zur Optimierung des Entwicklungsprozesses kommt die Containervirtualisierungs – Technologie *Docker* zum Einsatz.

3 Datenakquise und -analyse

Die *Staatsbibliothek zu Berlin* besitzt ein breites Spektrum von historisch bedeutsamen digitalisierten Zeitungen. Diese werden im hauseigenen Zeitungsinformationssystem namens *ZEFYS* kostenfrei bereitgestellt. Die Digitalisate, Volltexte und Metadaten der *Berliner Volks-Zeitung (BVZ)* dienen als Korpus für die im Rahmen des Laborprojektes entwickelte Suchmaschine. Bei der *Berliner Volks-Zeitung* handelt es sich um eine von 1849 bis 1944 veröffentlichte regionale deutsche Tageszeitung aus Berlin. Sie besitzt große Bedeutung für die Forschung im Bereich der Kulturwissenschaften, da sie im Gegensatz zu den meisten linken Parteizeitungen, über ein gutes Feuilleton verfügt. [QUELLE] Der Zeitraum der Digitalisate beläuft sich auf die Jahre 1890 bis 1930, wobei einige Jahre stärker abgedeckt sind als andere. Die Datensets umfassen jeweils strukturierte Metadaten im METS-XML-Containerformat für jede Ausgabe, per OCR erzeugte Volltexte im ALTO-XML-Format mit Wortkoordinaten, binarisierte TIFFs als Grundlage der OCR sowie JPEG2000-Bilder für die Anzeige. Insgesamt umfasst der Datensatz 103.771 digitalisierte Seiten.

3.1 Datenverarbeitung

Mithilfe des ALTO-XML-Formates wird die Möglichkeit geschaffen Digitalisate vollständig und originalgetreu zu replizieren. Aus diesem Grund sind neben Zeitungsinhalten Informationen zum Layout der Seite und die exakten Koordinaten der Wörter enthalten. Dokumente werden von *Elasticsearch* im JSON-Format repräsentiert. Daher ist es erforderlich die im Datensatz vorhandenen XML-Dateien in das JSON-Format zu konvertieren und dabei relevante Informationen zu extrahieren. Relevante Informationen stellen beispielsweise das Veröffentlichungsdatum der Zeitung, weitere Informationen wie die Ausgabe und den gesamten Text einer Zeitungsseite dar. Listing 1 zeigt ein Beispiel für die konvertierte JSON-Repräsentation einer Zeitungsseite. Interessant ist insbesondere der Schlüssel "Text", der ein dreifach geschachteltes Array bestehend aus den Texten einer Seite enthält. Innerhalb des Arrays der ersten Ebene befinden sich die einzelnen Artikel einer Seite, welche wiederum Arrays mit den Wörtern der Zeilen eines Artikels enthalten. Durch diese Repräsentation ist man in der Lage die einzelnen Artikel statt einer gesamten Seite als Ergebnis einer Suchanfrage auszugeben. Die vollständige Auflösung dieser Scanartefakte stellt eine unverhältnismäßig große Herausforderung dar, weshalb sich gegen die Durchführung der Artefaktsauflösung entschieden wurde. Stattdessen erfolgt bei der Auswahl der Referenzmenge für das Repository eine stärkere Einbindung von jüngeren Digitalisaten, wie näher in Kapitel 4.1 beschrieben wird.

```
{
  "Year": "1930",
  "Month": "01",
  "Day": "03",
  "NewspaperNumber": "001",
  "PageNumber": "004",
  "Edition": "0",
  "Issue": "059",
  "Text": "[ // Gesamter Zeitungstext
            [ // Artikel
             ["Nach","Schluss","der","Redaktion","
                eingetroffene","Depeschen."],
             ["Dortmund,","31.","Maerz.","Wie","die
                ","Rheinisch-Westfaelische"]
            ],
            [
             (...)
            ]
           ]"
}
```

Listing 1: JSON-Repräsentation einer Zeitungsseite

3.2 Textstatistiken

Um ein in sich geschlossenes Projekt im öffentlichen Repository zu präsentieren, wird der gesamte Datensatz auf eine geringere Größe reduziert. Tabelle 1 zeigt ausgewählte Textstatistiken für diese reduzierte Dokumentkollektion.

Gesamtanzahl Dokumente	6,144
Gesamtanzahl Worte	44,337,561,540
Vokabulargröße	1,663,267
Wörter mit Vorkommen > 1000	1,461,311
Wörter mit einmaligen Vorkommen	499

Tabelle 1: Textstatistiken für Dokumentkollektion

// ALI SCHAUT SICH ZAHLEN NOCHMAL AN! ANSONSTEN BEGRÜNDUNG FÜR VERZERRTE ZAHLEN SCHREIBEN!

4 Indizierung und Suche

In diesem Kapitel werden die Kernprozesse des Systems näher beschrieben. Dabei handelt es sich um den Indizierungs- und den Suchprozess. Abbildung 1 visualisiert die einzelnen Komponenten des Systems mittels UML-Komponentendiagramm.

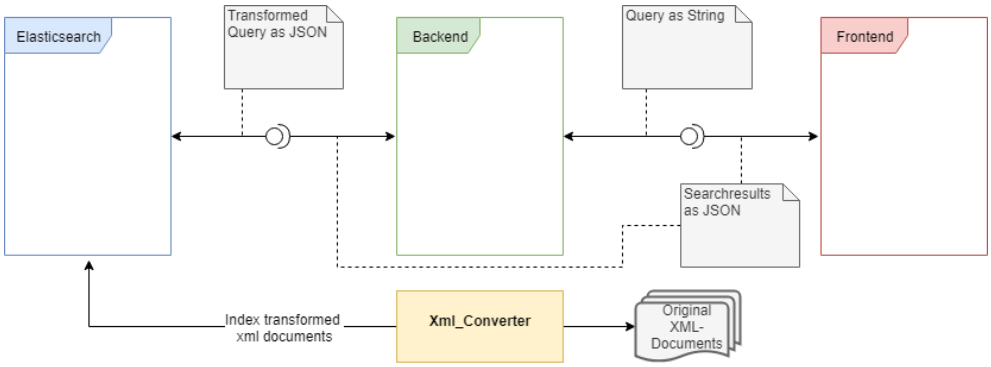

Abbildung 1: Architektur des Systems

4.1 Indizierungsprozess

Die im Rahmen der Datenvorverarbeitung erstellten JSON-Dokumente werden mit *Elasticsearch* mittels der bereitgestellten `bulk API` indiziert. Die `bulk API` dient der Erhöhung der Indizierungsgeschwindigkeit. Jede Seite einer Zeitung wird dabei in der Suchmaschine als eine JSON-Datei repräsentiert und enthält neben dem Veröffentlichungsdatum und weiteren Metadaten (Ausgabe, Seitenzahl, ...) die Texte der Zeitungsartikel. Diese stellen den wichtigsten Bestandteil für die Indizierung dar, da in diesem relevante Terme enthalten sind, die von einem Nutzer gesucht werden können. Wie jedoch bereits in den vorherigen Kapiteln dargelegt wurde, ist die Qualität der Terme durch eine Vielzahl von OCR-Scanartefakten beeinträchtigt.

4.2 Suchprozess

Ein Nutzer verfasst eine Suchanfrage, welche vom Client entgegengenommen und an das Backend weitergeleitet wird. Das *Flask*-Backend nimmt die Anfrage entgegen, konvertiert diese in eine *Elasticsearch*-konforme Syntax und leitet diese an *Elasticsearch* weiter. *Elasticsearch* transformiert diese Anfrage wiederum in das gleiche Format wie die bereits indizierten Dokumente. Anschließend erfolgt der Vergleich der Suchterme mit den Termen der Dokumente im invertierten Index. Auf Basis dieses Vergleichs wird ein Ranking der Dokumente nach absteigender Relevanz an das Backend zurückgegeben, welches dieses wiederum an das Frontend weiterleitet. Im Frontend wird das Ranking der Dokumente dem Nutzer präsentiert.

5 Vorstellung der Suchmaschine

Nachdem der Indizierungs- und Suchprozess des Systems in Kapitel 4 erläutert wurde, erfolgt in diesem Kapitel die Vorstellung der Suchmaschine, insbesondere des User Interfaces. Dabei wird zu Das User Interface besteht aus drei verschiedenen Sichen, die sich wie folgt darstellen:

1. Landing Page mit Eingabefeld für eine Suchanfrage
2. Seite zur Auflistung der Suchergebnisse

3. Seite zur Darstellung von Detailinformationen zu einer Zeitungsseite inklusive des zugehörigen Zeitungsscans

Bei der Beschreibung der Sichten erfolgt eine Erklärung der Prozesse die im System ablaufen.

5.1 Landing Page

Bei der *Landing Page* handelt es sich, wie der Name schon sagt, um die Seite, die ein Nutzer sieht sobald dieser die Suchmaschine verwendet. Auf dieser befindet sich neben einem thematisch passenden und ansprechenden Hintergrundbild ein Eingabefeld, in welcher die Nutzer eine Suchanfragen stellen können. Abbildung 1 zeigt die *Landing Page* mit dem Eingabefeld für Suchanfragen.

Abbildung 2: Landing Page der Historical News Search

5.2 Auflistung der Suchergebnisse

Nachdem ein Nutzer seine Suchanfrage in das Eingabefeld auf der *Landing Page* eingegeben hat, wird diese an die `search`-Methode im Backend übergeben. Wie bereits in Kapitel 3.2 beschrieben sucht diese in der Dokumentkollektion nach Zeitungsseiten, welche die Terme aus der Suchanfrage enthalten und gibt diese zurück. Die zurückgegebenen Zeitungen werden, nach ihrem Gewicht sortiert, in einer Liste von *Cards* dargestellt. Dabei wird eine Zeitungsseite von je einer *Card* repräsentiert. Diese enthalten neben einer Überschrift und dem Veröffentlichungsdatum ein kurzes Snippet, um dem Nutzer eine Vorschau auf die Artikel der Zeitungsseite zu bieten und diesen somit bei der Suche nach relevanten Ergebnissen zu unterstützen. Weiterhin ist zur Gewährleistung der Übersichtlichkeit des User Interfaces eine *Paginierung* implementiert. Jede Seite enthält besitzt zehn Resultate. Abbildung 2 illustriert die Auflistung der Suchergebnisse für eine Suchanfrage nach `max schmeling`.

// ABBILDUNG EINFÜGEN!

5.3 Detailansicht

Klickt ein Nutzer auf eine der *Cards* aus der Ergebnisliste, werden ihm sämtliche Artikel angezeigt, die auf der entsprechenden Zeitungsseite stehen.

// WIRD NACH UMBAU DER DETAILANSICHT FORTGEFÜHRT

// ABBILDUNG EINFÜGEN SOBALD FERTIGGESTELLT

6 Evaluation

Zur Effektivitätsmessung der *Historical News Search* wird das Effektivitätsmaß *Precision@k* sowie die daraus resultierenden Maße *Average Precision* und *Mean Average Precision* kalkuliert. Der *Precision*-Wert stellt den Anteil der relevanten Dokumente am Suchergebnis dar. Bei *Precision@k* handelt es sich um den *Precision*-Wert zu einem bestimmten *Recall*-Level. Grundlage für die Ermittlung, ob es sich bei einem Dokument um ein relevantes Dokument für eine gegebene Suchanfrage handelt, sind die von den Gruppenmitgliedern bereitgestellten Relevanzurteile. Die Relevanzurteile definieren, ob ein Dokument für ein Informationsbedürfnis relevant ist und bewegen sich im binären Spektrum zwischen *Relevant* und *Nicht relevant*. Ein Informationsbedürfnis wird in Form eines *Topics* modelliert. Ein *Topic* besteht aus drei Elementen, welche in Tabelle 2 exemplarisch für zwei gewählte *Topics* gezeigt werden.

	Topic
Suchanfrage	aaaaa
Beschreibung	bbbbb
Narrativ	ccccc

Tabelle 2: Modellierung eines Topics

Im Folgenden wird kurz die Vorgehensweise bei der Evaluation vorgestellt und erklärt, welche Schritte zur Verbesserung der Effektivität der Suchmaschine unternommen wurden.

6.1 Methodik

Die Evaluation durchläuft zwei Iterationen. In den beiden Evaluationsiterationen werden für die rund 40 definierten Topics die jeweiligen *Average Precision*-Werte berechnet. Grundlage für die Berechnung stellen die bereitgestellten binären Relevanzurteile dar. Aus den *Average Precision*-Werten wird wiederum das arithmetische Mittel kalkuliert, um die *Mean Average Precision* zu erhalten. Diese stellt das bedeutenste *Effektivitätsmaß* für die Evaluation des Systems dar, da dieses den Vergleich zweier Systeme auf ein Maß herunterbricht.

In der ersten Iteration erfolgt die initiale Evaluation der Suchmaschine mit sämtlichen standardmäßigen Einstellungen von *Elasticsearch*. Anschließend erfolgt das Refinement des Systems, bestehend aus der Anpassung von Parametern sowie der Implementierung zusätzlicher Funktionen. Die Maßnahmen,

die im Refinement zur Steigerung der Effektivität des Systems unternommen werden sind folgenden Kapitel näher beschrieben. Nach Abschluss des Refinements erfolgt die zweite Evaluationsiteration mit abschließendem Vergleich der beiden Systeme.

6.2 Refinement

Die Optimierung des Retrieval-Prozesses erfolgt durch mehrere Maßnahmen, welche im Folgenden näher erläutert werden. Während der durchgeführten, ersten Evaluationsiteration ist festzustellen, dass Stoppwörter zu gleichem Maße in die Berechnung des Scores eines Dokumentes eingehen, wie die anderen Terme. Dadurch verlieren andere, wichtigere Terme an Gewicht bei der Erstellung des Scores, was zu einer schlechteren Platzierung von relevanten Dokumenten führt. Daher bietet sich die Optimierungsmethode des *Stoppings* an, um eine Verbesserung der Retrievalperformanz zu erreichen.

Unter **Stopping** versteht man das Entfernen von Stoppwörtern aus der Menge der Indexterme. Stoppwörter bezeichnen Wörter, welche häufig anzutreffen sind und keine Relevanz für die Erfassung des Dokumenteninhalts besitzen. *Elasticsearch* bietet die Möglichkeit vordefinierte sowie selbsterstellte Stopwortlisten zu verwenden. Zur Implementierung des *Stoppings* wird die vorgefertigte Stoppwortliste für die deutsche Sprache sowie zusätzlich eine eigene Stoppwortliste mit ergänzenden Wörtern verwendet.

Zur gezielten Suche nach bestimmten Attributen wird außerdem eine eigene **Query Syntax** eingebaut. Bei den Attributen handelt es sich um die in Listing 1 gezeigten Attribute der JSON-Dokumentrepräsentationen. Ein Nutzer ist beispielsweise in der Lage durch Eingabe der Suchanfrage `year:1930 month:01` nach Zeitungsartikeln vom Januar 1930 zu suchen. Durch diese Funktion ist eine gezieltere Suche insbesondere für erfahrene Nutzer möglich.

Zur Verbesserung der Nutzererfahrung wird ein **Highlighting** der in der Suchanfrage enthaltenen Terme implementiert. Weiterhin werden dem Nutzer während der Erstellung der Suchanfrage **Vorschläge zur Vervollständigung** angeboten. Diese Funktionen verhindern orthografische Fehler bei der Eingabe und gewährleisten eine schnellere Erkennung von relevanten Dokumenten durch den Nutzer.

7 Zusammenfassung und Ausblick

Literatur