Universität Regensburg

Philosophische Fakultät III
Sprach- , Literatur- und Kulturwissenschaften
Institut für Information und Medien, Sprache und Kultur (I:IMSK)
Lehrstuhl für Informationswissenschaft

Vertiefungsseminar Information Retrieval
Modul: INF-M 04 (B.A.)
SS 15
Leitung: Florian Meier

Vertiefungsseminar Information Retrieval

Fabian Schatz **1636790**
Mark Engerisser **1625264**
Matthias Bräuer **1632272**

Inhalt

1 Einleitung

Im Rahmen des Vertiefungsseminars Information Retrieval wurden viele Arten der Informationsgewinnung vorgestellt. Unsere Gruppe konzentrierte sich dabei auf den Bereich des Question.Answering, welcher bereits durch größere Projekte wie „Watson“ der Firma IBM oder auch ELISE aus dem Projektseminar „URTalking“ an der Universität Regensburg bearbeitet wird. Question-Answering-Systeme fundieren auf dem Prinzip, dass im Gegensatz zu herkömmlichen-Suchanfragen, wie z.B. Google, Yahoo oder Bing, genau ein Ergebnis zurückgeliefert wird. Diese Eigenschaft schafft gute Voraussetzungen bei einer Quizsendung, wie JEOPARDY!, ein solches System als Mitspieler zu nutzen. Dies wurde bereits 2011, durch das oben genannte System „Watson“ erfolgreich dargestellt. Das in dieser Arbeit vorgestellte Projekt nutzt die frei verfügbare JEOPARDY!-Datenbank aus Frage-Antwort-Paaren und setzt diese in einem hybriden Projekt aus einem herkömmlichen Question-Answering-System und einer spielbaren Oberfläche, welche dem originalen Spielprinzip der Quizshow nachempfunden ist, ein.

2 Vorbereitung

2.1 Dokumentenkollektion

Als grundlegende Dokumentenkollektion für unser Projekt haben wir eine JSON Datei mit Frage-Antwort-Paaren aus vergangenen JEOPARDY!-Sendungen bekommen. Diese Kollektion beinhaltet 216930 Paare, welche in 27995 Kategorien eingeteilt sind. Die einzelnen Objekte sind jeweils mit der Frage, Antwort, Sendungsnummer, Ausstrahlungsdatum, Wert und Kategorie versehen.

Bei genauerer Durchsicht der Daten fielen uns sehr viele Inkonsistenzen in der Formatierung sowie den Werten im „Value“-Feld auf. Ebenso wurde uns bewusst, dass die uns zu Verfügung gestellte Dokumentenkollektion für das eigentlich geplante Question-Answering-System nicht als sinnvolle Grundlage genutzt werden kann.

2.2 Klassisches IR System vs. JEOPARDY!

Bei einem klassischen Question-Answering-System geht es vor allem darum, dass die Eine korrekte Antwort zu einer Frage gefunden wird. Da dies ein relativ komplexes Verfahren ist, gibt es auf Themengebiete spezialisierte Systeme sowie allgemeine Sys-

teme. Den allgemeinen Systemen kann man allerdings durch zusätzliche Metainformationen Einschränkungen vermitteln.

Da JEOPARDY! eine Game-Show ist und somit der Spieler mehr oder weniger die Frage zu einer Antwort formulieren muss, funktioniert das Retrieval hier etwas anders. Es muss also ein gegebenes Frage-Antwort paar zurückgeliefert werden und danach die vom Spieler eingegebene Antwort auf Korrektheit geprüft werden. Somit ist unsere Suchfunktion mehr in den Bereich des Auffindens eines Frage-Antwort-Paares aus einer JEOPARDY!-Show verschoben und weniger ein klassisches Question-Answering-System zur Beantwortung von allgemeinen Informationsbedürfnissen.

3 Umsetzung

3.1 Solr

3.1.1 Allgemein

Um einen Kern erstellen und somit eine Collection indexieren zu können, müssen zwei Konfigurationsdateien erstellt werden. Die erste heißt in der Regel „solrconfig.xml“ und setzt die gewünschten „Request-Handler“, die bestimmen welche „Pfade“ des Servers gesetzt sind und wie diese reagieren. Die Zweite Datei heißt „schema.xml“ und legt fest, welchen Typ Felder einer Collection haben und wie diese zur Indexzeit und zur Queryzeit verarbeitet werden. Grundsätzlich gibt es zwei Komponenten in der „schema.xml“. Die erste Komponente beschreibt alle Felder (dynamisch oder statisch), die in einem Dokument auftreten können und weist ihnen einen Typ zu. Zusätzlich können einige Attribute bestimmt werden, z.B. ob das jeweilige Feld indexiert werden soll oder nicht. Die Feldtypen stellen die zweite Komponente dar. Einem Feldtyp wird eine Klasse zugewiesen, welche den Datentyp („long“, „string“, etc.) bestimmt. Innerhalb des Feldtypen können „Analyzer“ gesetzt werden, welche die Verarbeitung („tokenisieren“, „Stopwörter filtern“, etc.) der Dokumente zur Query- und Indexzeit festlegen.

3.1.2 Implementierung

3.1.2.1 Solrconfig.xml

Für die „solrconfig.xml“ wurde die von „Solr“ bereitgestellte Datei verwendet und Kommentare, sowie für das Projekt offensichtlich unwichtiger Code entfernt. Die wichtigen Komponenten sind nun ein „Update-„ und ein „Select-“ Handler. Der „Update-Handler“ ist für das Aktualisieren des Index zuständig. Der „Select-Handler“ setz einige Defaultwerte, wie z.B. die Anzahl der zurückgegebenen Dokumente (rows=10) für die Suche unter dem Pfad „/select“.

3.1.2.2 Schema.xml

Da der JEOPARDY!-Datensatz einheitlich ist und jedes Dokument exakt gleich aufgebaut ist, wird lediglich für das randomisierte Auswählen von Objekten ein dynamisches Feld benötigt.

Ein Objekt besteht aus folgenden Attributen:

- category
- question
- answer
- value
- air_date
- round
- show_number

Da das Frontend nur Anfragen auf die Felder „category“, „questions“, „answer“ und „value“ sendet, werden die restlichen nur zur Anzeige gespeichert, jedoch nicht indexiert. Um eine sinnvolle „Facet“-Suche für die „TagCloud“ zu gewährleisten und die normale Suche nicht zu beeinflussen, werden die Felder „question“ und „answer“ kopiert und diese zur Index- und Queryzeit von Stopwörtern befreit. Eine Liste mit allen Stopwörtern ist im Ordner „conf/lang“ zu finden.

Wie jedes Feld werden auch die kopierten Fehler durch einen Analyzer tokenisiert und in Kleinbuchstaben überführt.

3.2 SUI/Design

Wir entschieden uns als Gruppe für ein schlichtes und übersichtliches Design des Systems mit einer einfachen Informationsstruktur. Die Landingpage sollte somit lediglich

unsere drei Funktionen, die Tagclouds, die Suche und das Erstellen eines neuen Spiels, enthalten.

Die Tagclouds werden auf einer neuen Seite angezeigt um die Übersichtlichkeit zu wahren und eine vereinfachte Exploration zu ermöglichen.

Wir entschieden uns für die zurückgelieferten Suchergebnisse auf der Mainpage, darzustellen, um ein einfaches stöbern in der Kollektion zu ermöglichen und den Workload für den Nutzer so gering wie möglich zu halten. Die Begrenzung beziehungsweise Erweiterung der zurückgelieferten Ergebnisse auf drei Resultate, ist dem Fakt geschuldet, dass es uns nicht möglich war mit einer hohen Wahrscheinlichkeit das richtige Ergebnis anzuzeigen. Somit überlassen wir dem Nutzer die Auswahl des für ihn relevanten Eintrags.

Das Erzeugen eines neuen JEOPARDY!-Spiels wurde minimal und einfach gehalten. Der Nutzer kann entscheiden ob er ein neues Spiel erstellen möchte oder ob er zunächst eine Erklärung benötigt. Die Auswahl ob er alleine spielen möchte oder mit seinen Freunden ein Mehrspielerspiel erstellen möchte erfolgt auf der nächsten Seite.

3.3 Implementierung des Games

Um dem Spieler jedes Mal eine neue Zusammenstellung von Kategorien und dessen Fragen zu ermöglichen wird zu Beginn eines Spiels eine randomisierte Anfrage an den Server geschickt, der 5 zufällige Kategorien zurücksendet. Nun werden für jede Kategorie, alle Fragen, gruppiert nach dem Wert der Frage, geholt und sortiert. Aufgrund verschiedener Probleme, wie nicht einheitliche Werte im Feld „value“ und die fehlende Möglichkeit, mittels einer Anfrage an den Server nach einer Kategorie zu suchen, die mindestens ein Dokument mit den Werten „$200“, „$400“, „$600“, „$800“, „$1000“ im Feld „value“ enthält, werden im Client 5 Fragen mit aufsteigendem „value“ ausgewählt und in einem Spielfeld (5 x 5) angezeigt. Da es zusätzlich zu den oben genannten Werten nicht nur „$100“, „$300“, „$500“, „$700“ und „$900“ gibt, sondern auch ungerade Werte, wie „$1.111“, und jede Frage mit einem Bild repräsentiert wird, kann es zu Fehlern zwischen gewonnenem/verlorenem Geld und der Anzeige des Werts bei einzelnen Fragen kommen.

Da die Überprüfung einer Antwort auf Richtigkeit zu einer Frage schwierig ist und lediglich als exakter Vergleich implementiert wurde, ist es schwierig Fragen richtig zu beantworten. Um den Spielspaß trotzdem aufrecht zu erhalten und weil das Spielen alleine nicht sehr interessant ist, wurde zusätzlich ein Multiplayer implementiert. Dieser Multiplayer besteht aus 3 Spielern und sieht vor, dass eine Person den Part des Showmasters und somit die Überprüfung der Antwort auf Richtigkeit übernimmt, indem er zusätzlich zur Frage die richtige Antwort angezeigt bekommt. Um zu klären wer Punkte addiert oder abgezogen bekommt, hat der Showmaster die Möglichkeit, auszuwählen welcher Spieler am Zug war. Für den Fall, dass kein Spieler auf die Frage antwortet, bietet das System einen Button an, der die Frage als beantwortet setzt, jedoch keinem Spieler etwas abzieht.

4 Usability Test

Zur Verbesserung unseres Systems führten wir einen Usability Test mit zwei Probanden durch. Der Test stellte die Aufgaben, die Kollektion zu erforschen und interessante Fragen herauszufinden, sowie eine Einzelspieler und eine Mehrspielerpartie zu spielen.

Die Probanden kamen mit unserem System sehr gut zu Recht und äußerten, dass das Design sowie der Umgang mit dem System sehr einfach und intuitiv sind.

Als größtes Problem kam aus den Tests hervor, dass eine verbesserte Verarbeitung der Antworten erfolgen muss, um ein zufriedenstellendes Spielerlebnis gewährleisten zu können.

Das Suchfeature war für die Probanden nicht zufriedenstellend, da lediglich Phrase-Queries möglich sind und somit die Suche nicht wie eine für die Probanden bekannte Suchmaschine genutzt werden konnte.

5 Fazit

Trotz der großen Anzahl an Datensätzen in der Frage-Antwort Datenbank reichen die Informationen kaum aus, um einen alltäglichen Mehrwert der Applikation zu erreichen. Dies begründet sich auf der Tatsache, dass viele Fragen zu spezifisch gestellt und meist nur Randinformationen der Antwort als Frageparameter mitgeben. Ein Beispiel hierfür wäre folgende Konstellation: Der Nutzer fragt das Question-Answering-System

„What is the Phantom of the Opera“. Die wahrscheinliche Antwort des Systems wäre für den gewöhnlichen Nutzer unbrauchbar: „This titel character of a musical sometimes signs notes with the initials O.G., for ‚Opera Ghost‘“Aus diesem Grund lesen größere Projekte wie IBM Watson kontinuierlich neue Informationen ein und wächst in den kommenden fünf Jahren bis zu 800%. Diese Tatsache ermöglicht den Nutzern immer präzisere Antworten auf ihre Problemstellungen. Die Möglichkeiten sind jedoch nicht auf simple Frage-Antwort Systeme begrenzt, sondern lassen auch logische Schlussfolgerungen zu, welche in der Forschung oder zur Bildung einer künstlichen Intelligenz nützlich sind.