Ähnlichkeitsanalyse von Dokumenten

Veranstaltung: Wissensmanagement

Semester: Sommersemester 2017

Teilnehmer: Jan-Hendrik Hemmje, Johannes Giere

Stand: 01.07.2017

Ziel des Projektes

Das Projekt hat zum Ziel, Textdokumente zu analysieren und ihre Ähnlichkeit untereinander festzustellen und zu präsentieren. Zu diesem Zweck sollen die Dokumente in Terme aufgeteilt und die Ähnlichkeit mithilfe ihres TF-IDF-Wertes berechnet werden. Da Dokumente verschiedenster Dateitypen analysiert werden sollen, wird die Indizierungssoftware „Apache Solr“ verwendet. Diese Software basiert auf „Apache Lucene“ und verwendet zusätzlich „Apache Tika“, welches das Parsen von (unter anderem) TXT-, PDF- & Word-Dateien übernimmt. Aus diesen Gründen ist „Apache Solr“ optimal für dieses Projekt. Die berechneten Ähnlichkeiten sollen in einer Grafik dargestellt werden und über die Eingabe eines Suchwortes sollen ähnliche Dokumente gefunden werden.

Ausführen der Software

Das Projekt basiert auf zwei Programmen. Es wird Apache Solr und die mitgelieferte – von uns erstellte – Java-Anwendung benötigt. Vor dem ersten Start der Java-Anwendung ist es notwendig Solr zu konfigurieren. Solr kann von der Webseite http://lucene.apache.org/solr/ bzw. von https://archive.apache.org/dist/lucene/solr/6.5.0/ heruntergeladen werden.

Es wird vorausgesetzt, dass Linux als Betriebssystem und Solr in der Version 6.5.0 verwendet wird. Neuere Versionen von Solr könnten auch funktionieren, jedoch kann dies nicht garantiert werden. Mit der Java-Anwendung wird ein Unterordner „scripts“ mitgeliefert. Die dort hinterlegten Skripte automatisieren die Konfiguration von Solr. Sollte Solr nicht im Pfad „/opt/solr-6.5.0“ entpackt worden sein, so muss dies in den Skripten angepasst werden. Die Skripte müssen auf der Maschine ausgeführt werden, auf der Solr bereitgestellt wird.

Die Skripte starten Solr, legen einen neuen Feldtypen an und ersetzen den Typ eines bestehenden Feldes mit dem neuen Typ. Zu Beginn wird Solr mit dem Skript „startAsCloud.sh“ gestartet. Danach muss über das Web-Interface von Solr eine Collection angelegt werden. Dafür wird der Link „http://localhost:8983/solr“ aufgerufen und über den Punkt „Collections“ → „Add Collection“ eine Collection hinzugefügt. Die Collection muss „wm“ genannt werden und als „config set“ muss „gettingstarted“ ausgewählt werden. Die restlichen Einstellungen können auf dem Standardwert belassen werden. Im Anschluss wird „addNewFieldType.sh“ ausgeführt, gefolgt von „replaceFieldType.sh“. Die Ausführung hat funktioniert, wenn als „status“ 0 zurückgegeben wird.

Anschließend kann die Java-Anwendung konfiguriert werden. Sollte Solr nicht auf der selben Maschine oder dem Port 9983 bereitgestellt werden, so muss die Variable „CONNECTION_STRING“ in der Klasse „src/main/java/solr/SolrConnection.java“ angepasst werden. Im nächsten Schritt sollten die Maven-Pakete installiert werden. Ist dies erfolgt, so kann die Anwendung über die Klasse „view.App“ gestartet werden oder mit Maven der Quellcode kompiliert werden.

Benutzung der Software

Nach dem Start des Programms können links oben per „Datei auswählen“ mehrere PDF- oder TXT-Dateien ausgewählt werden. Diese erscheinen dann links in der Liste. Per Klick auf „Importieren“ werden die ausgewählten Dateien dann in Solr importiert. Der Import-Vorgang ist abgeschlossen, wenn die ausgewählten Dateien wieder aus der Liste verschwunden sind. Je nach Anzahl der Dateien kann der Vorgang einige Sekunden bis Minuten in Anspruch nehmen.

Per Klick auf „Lade Abstandsgraphik“ wird die Distanzmatrix berechnet und dargestellt. Bei einem Hover über die Punkte wird der dazugehörige Titel eingeblendet.

Um zu einem bestimmten Dokument eine Liste der ähnlichsten Dokumente zu erhalten, kann im Drop-Down-Menü unter „Dokumentenähnlichkeit“ ein Dokument ausgewählt werden und per Klick auf „Suchen“ die Liste der ähnlichsten Dokumente geladen werden. An erster Stelle taucht dabei das ausgewählte Dokument auf, da hier die Übereinstimmung bei 1,0 liegt.

Links unten unter „Suchterm“ kann nach einem Term gesucht werden. Die nebenstehende Liste ist ebenfalls nach der Übereinstimmung sortiert, sodass Dokumente, in denen der Term oft auftaucht weiter oben in der Liste stehen.

Verwendete Algorithmen

Damit die Ähnlichkeit von Dokumenten berechnet werden kann, wird ein invertierter Index erstellt. Über den Index kann zu jedem Term ein IDF-Wert und TF-IDF-Wert berechnet werden. Über das Kosinus-Maß wird die Ähnlichkeit zwischen Dokumenten und User-Queries berechnet. Als Werte werden hier die TF-IDF-Werte verwendet.

Um die Dokumente in einer Distanzmatrix darzustellen, werden die berechneten Ähnlichkeiten in Distanzen transformiert und mithilfe der Bibliothek „mdsj.jar“ in einer Distanzmatriz abgebildet. Die Distanzmatrix wird anschließend in einen Graphen überführt.

Probleme

Die Parsing-Funktionalität des Tika-Servers, welche in Solr integriert ist, bereitet erhebliche Probleme bei Dateien, welche keine TXT-Dateien sind. PDF- und Word-Dateien werden von Solr empfangen und im ersten Schritt an Tika weitergeleitet. Tika soll die Dokumente parsen, damit Solr die Terme extrahieren kann. Leider werden die Dokumente ungenügend geparst, sodass Wortkonstrukte entstehen, die über alle Dokumente einzigartig sind. Dies bedeutet, dass es sehr viele Terme gibt, die im Ursprungsdokument gleich sind, durch das Parsen jedoch so verändert werden, dass das Endprodukt des Terms einzigartig ist. Zum Beispiel werden von Termen Buchstaben entfernt und dieser Entfernungsprozess ist nie einheitlich.

Da auf dem Feldtypen Stemming aktiviert ist, wurde vermutet, dass das Stemming das Problem verursacht. Da jedoch Solr TXT-Dateien ohne die Hilfe von Tika parsen kann, wurden die genutzten PDF-Dateien mit Calibre in TXT-Dateien konvertiert und an den Solr-Server zur Indizierung gesendet. In den erstellten Termen trat das Problem nun nicht mehr auf.

Daher wird geschlussfolgert, dass der Parsing-Prozess von Tika nicht zufriedenstellend funktioniert. Aus diesem Grund wird geraten, für eine produktive Nutzung der Anwendung, nur .txt-Dateien zu verwenden.