3. Übung zur Vorlesung "Internet-Suchmaschinen" im Sommersemester 2009

Prof. Dr. Gerd Stumme, M.Sc. Wi-Inf. Beate Krause

20. Mai 2009

1 Tokenizing

1. Skizzieren Sie in Pseudocode oder einer Programmiersprache Ihrer Wahl, wie man aus einem HTML-Dokument den reinen Text extrahieren kann.

   Welche Probleme hat Ihre einfache Lösung noch?

2. Berücksichtigt Ihr Verfahren auch

   - `ALT`-Tags von Bildern?
   - `TITLE`-Tags von Hyperlinks?
   - Keywords in `META`-Tags?
   - Kommentare und Skripte (sollen ausgeblendet werden!)

   Wie aufwendig ist es, dies alles nachzurüsten?

3. Man will die oben genannten Features haben und zusätzlich auch noch Strukturinformation verarbeiten, also etwa die erste `<h1>`-Überschrift besonders gewichten, Hyperlinks extrahieren, usw.

   Finden Sie eine elegantere Möglichkeit, dies alles anzubieten, ohne von Hand einen entsprechenden Tokenizer zu bauen?

4. Bei der Anwendung des Porteralgorithmus werden die folgenden Wortpaare auf die selbe Wurzel abgebildet:

   - abandon - abandoned
   - university - universe
   - volume - volumes

   Welche dieser Anwendungsfälle sind sinnvoll, welche könnten für die Suche kritisch sein?

2 Indexstrukturen

Wir nehmen an, daß wir einen invertierten Index für folgenden Korpus gebaut haben:

- 1.000.000.000 Dokumente
- 2.000.000 Terme
- jeder Term komme im Mittel in 100.000 Dokumenten vor
- jeder Term und jeder Listeneintrag sei 16 Byte groß.

Der Index stehe als Aneinanderreihung der Terme und Dokument-Term-Gewichte auf dem Sekundärspeicher.

Weiterhin haben wir den Index auf einem RAID-Array mit

- 4 kB Blockgröße, 64 Bit breite Blocknummern
- 8 ms mittlerer Zugriffszeit
- 50 MB/s Übertragungsrate bei linearem Zugriff

1. Schätzen Sie ab, wie lange das Auffinden eines Term-Vorkommens bei linearer Suche im Mittel dauern würde.
2. Kennen Sie eine Indexstruktur, um eine solche Suche auf dem Hintergrundspeicher zu beschleunigen? Schätzen Sie die Kosten mit einer solchen Datenstruktur ab.

3 Evaluation

Ein Student soll ein Referat über den höchsten Vulkan der Welt, Ojos del Salado, schreiben. Dafür nutzt er zwei Suchmaschinen, und erhält jeweils eine Liste aus 30 URLs. Für diese erstellt er folgende Relevanzlisten (+ repräsentiert eine relevante URL, - repräsentiert eine nicht relevante URL):

$\triangle_1 = (+|+|-|+|-|-|+|+|-|-|-|-|-|-|-|-|+|-|+|-|-|-|-|+|+|+|-|+|-|+)$

$\triangle_2 = (-|+|+|-|-|-|+|+|-|+|+|-|+|-|-|+|-|-|+|-|-|+|-|-|+|-|+|-|-|-|-|-|+)$

1. Zeichnen Sie für beide Systeme den Precision-Recall Graphen sowie die Interpolation der Graphen. Gehen Sie davon aus, dass für jede Liste insgesamt 12 Dokumente relevant sind.
2. Welche Ranking-Liste ist in welchem Anwendungsszenario besser?
3. Das F-Measure wird als harmonisches Mittel von Precision und Recall definiert. Welchen Vorteil hat die Benutzung des harmonischen Mittels gegenüber des arithmetischen Mittels?

4 Praxisübung

Abgabe: 03.06.2009

Implementieren Sie mit Hilfe eines HTML-Parsers (z. B. JTidy oder NekoHTML) eine Unterklasse `HTMLDocument` von `Document`. Ein `HTMLDocument` soll anstelle eines Textfiles ein HTML-Dokument lesen und den Text extrahieren können. Benutzen Sie dazu den DOM-Parser von JTidy!

Dabei sollen die Sonderfälle von diesem Übungsblatt (ALT-Tags usw.) berücksichtigt werden.

Bereinigen Sie den Text um nichtalphabetische Zeichen, bauen Sie einen Porter-Stemmer ein, und entfernen Sie die Stopwords aus der unten genannten Liste!

JTidy: `http://jtidy.sourceforge.net`

NekoHTML: `http://people.apache.org/~andyc/neko/doc/html/`

Stemmer: `http://www.tartarus.org/~martin/PorterStemmer/`

Stopwords: `http://www.unine.ch/info/clef/englishST.txt`