

Extended Boolean Model

Filip Leško, Tímea Čitbajová

Popis projektu

Cieľom projektu bolo vytvorenie rozšíreného boolovského modelu a pomocou neho vyhľadávať dokumenty/texty v database a vytvorenie webovej aplikácie pre zadávanie dotazov od užívateľa a ich následné vyhodnotenie.

Boolovský model

Funkcionalitou modelu je vyhľadať dokument v database na základe dotazu od užívateľa (Dotaz v tvare “term AND/OR/NOT TERM”). Pred vyhľadávaním sa texty spracujú. Najprv sa prevedie odstránenie stopových slov a následne stemmatizácia/lemmatizácia aby sa slovník osekala. Následne sa spracuje dotaz od užívateľa a vyberú sa dokumenty podľa dotazu.

Rozšírený boolovský model

Ide o verziu boolovského modelu, kde sa však pracuje s váhami. Každý term v dokumente má svoju váhu a tá sa pri dotaze spracuje. Výsledkom je následne preferencia dokumentov. Pri počítaní váhy sa vyžije td-idf schéma.

Spôsob riešenia

Všetky dokumenty sú najprv načítané a je na nich prevedené osekánie o stopové slová a následne stemmatizácia/lemmatizácia (Porter stemmer). Pri načítaní jednotlivých termov sa počíta ich frekvencia a váha pomocou td-idf schémy (td – term frequency, idf – inverse document frequency). Termy načítame do 2 štruktúr :

1. Matica term/document list pre sekvenčný prechod
2. Invertovaný zoznam

Po načítaní vstupu od užívateľa sa rozdelí na jednotlivé tokeny po slovách. Tie sa následne prevedú do RPN (reverse polish notation) pomocou shunting-yard algoritmu, tým sa vytvorí strom, ktorým budeme prechádzať.

Vyhodnocuje sa potom postupnosť jednotlivých termov. Výsledok je postupne ukladaný na zásobník. Podľa operácie (AND/OR/NOT) potom aplikujeme jednotlivé vzorce na výpočet relevancie.

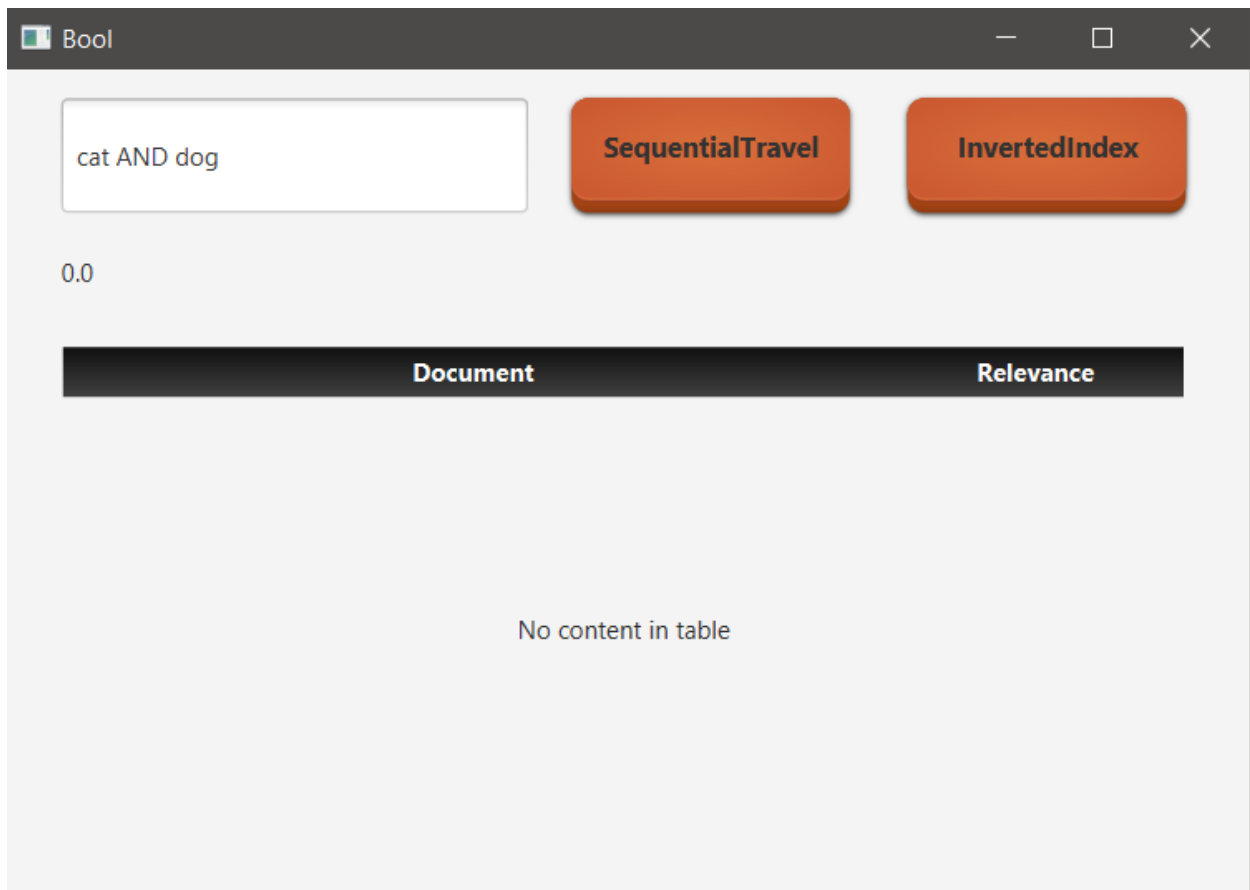
Nakoniec zoradíme dokumenty podľa relevancie a zobrazíme.

Implementácia

Projekt bol implementovaný v jazyku Java. Pre stemmovanie a lematizáciu termov bol použitý Porter stemmer. Na grafickú aplikáciu bola použitá knižnica Javafx.

Príklad výstupu

Na vstup zadávame dotaz v tvare term AND/OR/NOT term prípadne sa dajú použiť zátvorky ako prioritný operator. Medzi jednotlivými slovami a zátvorkami musí byť medzera.



Bool

cat AND dog

SequentialTravel

InvertedIndex

0.0

Document	Relevance
----------	-----------

No content in table

Bool

cat AND dog

SequentialTravel

InvertedIndex

0.1965ms

Document	Relevance
Royal_Society_Corpus_v4.0.1_text_101226.txt	0.5461269296672022
Royal_Society_Corpus_v4.0.1_text_112462.txt	0.5114427391247096
Royal_Society_Corpus_v4.0.1_text_112336.txt	0.3310358575104889
Royal_Society_Corpus_v4.0.1_text_112696.txt	0.30914444527568596
Royal_Society_Corpus_v4.0.1_text_101349.txt	0.2872411127744381
Royal_Society_Corpus_v4.0.1_text_101456.txt	0.2872411127744381
Royal_Society_Corpus_v4.0.1_text_101230.txt	0.2860882226267192
Royal_Society_Corpus_v4.0.1_text_101412.txt	0.28445144205341766
Royal_Society_Corpus_v4.0.1_text_112622.txt	0.28445144205341766

Experimentálna sekcia

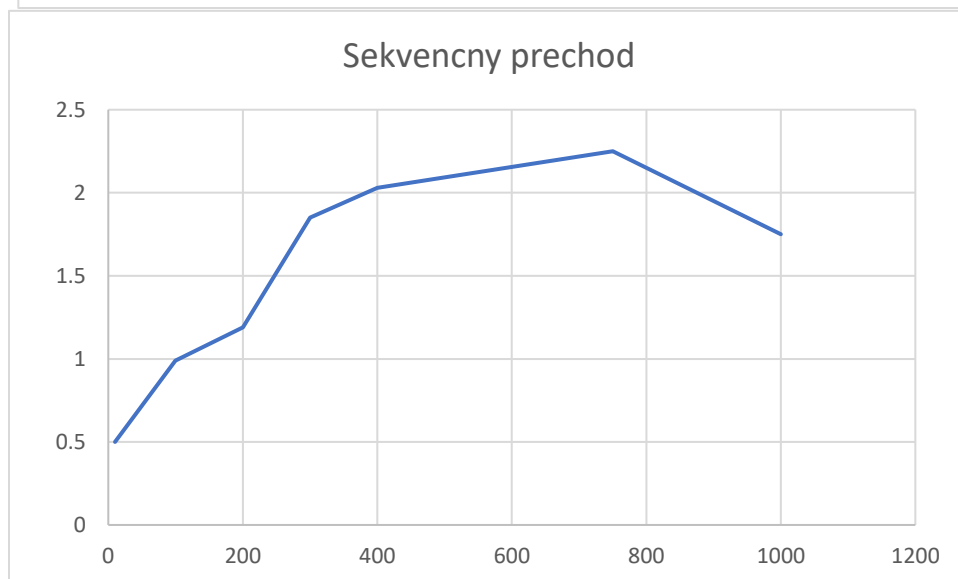
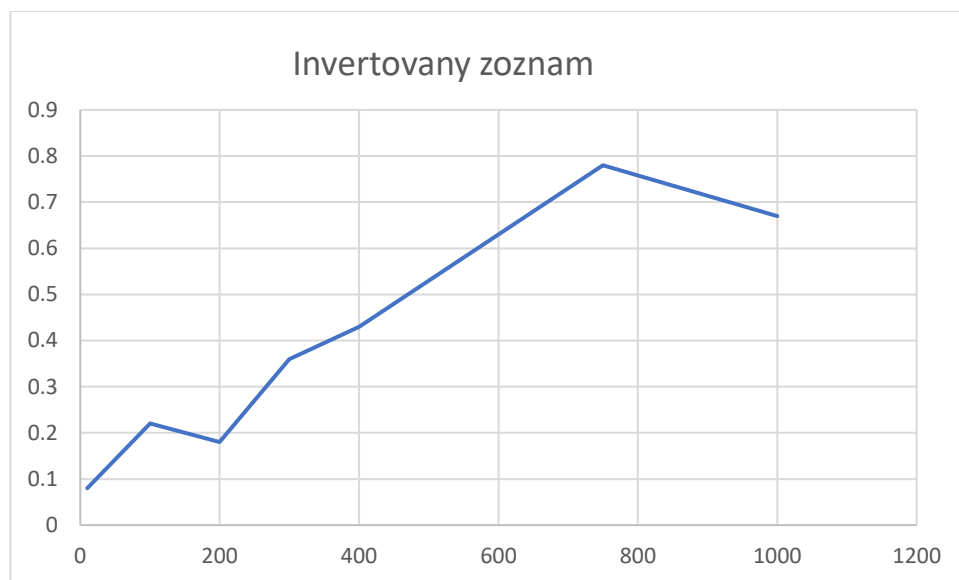
V projekte sme testovali rozdiel medzi časovými zložitostami prechodu invertovaným zoznamom a sekvenčným priechodom. Testovali sme na rôznych textových dokumentoch. Výsledky:

Invertovaný zoznam

Počet dokumentov	Dĺžka vyhodnocovania (ms)
10	0.08
100	0.22
200	0.18
300	0.36
400	0.43
750	0.78
1000	0.67

Sekvenčný prechod

Počet dokumentov	Dĺžka vyhodnocovania (ms)
10	0.5
100	0.99
200	1.19
300	1.85
400	2.03
750	2.25
1000	1.75



Z našich výsledkov vyplýva, že prechod invertovaným zoznamom bol asi 10x rýchlejší.

Diskusia

Aplikácia neposkytuje validáciu dotazov. Taktiež sa pri spustení aplikácie prepočítava kolekcia, čo je neefektívne. Riešením by bolo kolekciu spracovať raz a následne si ju uložiť a pri spustení programu iba načítať.

Ďalším nedostatkom je spracovanie pomocou Javafx. Ideálnym sprostredkovaním by bolo webové rozhranie. To nebolo implementované z dôvodu nedostatku znalostí html/css.

Ďalšie optimalizácie by mohli byť prevedené na celkovej zložitosti kódu.

Záver

Experimenty potvrdili, že invertovaný zoznam efektívnejšie spracováva dotazy. Hlavnou záťažou projektu bolo parsovanie dotazov a vytváranie užívateľského rozhrania.

Zdroje

<http://orion.lcg.ufrj.br/Dr.Dobbs/books/book5/chap15.htm>

<http://irkwan.github.io/irkwan/tugas/2016/06/03/13514104-Extended-Boolean-Model/>

<https://www.geeksforgeeks.org/evaluate-the-value-of-an-arithmetic-expression-in-reverse-polish-notation-in-java/>

<http://www.learn4master.com/algorithms/convert-infix-notation-to-reverse-polish-notation-java>