

# Obsah

<b>1 Úvod.....</b>	<b>3</b>
<b>2 Príprava dokumentov.....</b>	<b>4</b>
2.1 Reprezentácia dokumentu.....	4
2.1.1 Booleovský model.....	4
2.1.2 Multimnožinový model.....	4
2.1.3 Vektorový model.....	5
2.2 Techniky zlepšovania reprezentácie dokumentov.....	6
2.2.1 Transformácia na korene slov.....	6
2.2.2 Vynechávanie stop slov.....	8
2.2.3 Normalizácia dĺžky dokumentu.....	8
2.2.4 Váhovanie slov podľa logických častí dokumentu.....	9
2.3 Postup prípravy dokumentov.....	10
<b>3 Zhlukovanie dokumentov.....</b>	<b>11</b>
3.1 EM algoritmus.....	12
3.2 Aspektový model.....	14
3.2.1 Zohľadnenie odkazov medzi dokumentami.....	16
3.3 Zhlukovací algoritmus k-means.....	17
3.3.1 Zohľadnenie odkazov medzi dokumentami.....	18
<b>4 Špecifikácia požiadaviek.....</b>	<b>19</b>
<b>5 Návrh riešenia.....</b>	<b>20</b>
5.1 Zberač.....	20
5.2 Transformačný modul.....	20
5.3 Zhlukovací modul.....	20
5.4 Implementačné prostredia.....	21

<b>6 Opis riešenia.....</b>	<b>22</b>
6.1 Zber a transformácia dát.....	22
6.2 Úprava dát pre experimenty.....	22
6.3 Zhlukovanie.....	24
<b>7 Výsledky experimentov.....</b>	<b>25</b>
7.1 Priebeh maximalizácie vierohodnosti modelu.....	25
7.2 Zmeny počtu hľadaných aspektov.....	26
7.3 Kumulatívna entropia ako metrika kvality zhlukovania.....	28
7.4 Overenie správnosti výsledkov zhlukovania.....	30
7.5 K-means verus EM algoritmus.....	31
7.6 Diskusia.....	32
<b>8 Zhodnotenie.....</b>	<b>34</b>
<b>9 Možnosti rozšírenia.....</b>	<b>35</b>
<b>10 Použitá literatúra .....</b>	<b>36</b>
<b>11 Príloha A - Technická dokumentácia.....</b>	<b>38</b>
11.1 Dátové štruktúry.....	38
11.2 Zberač.....	41
11.3 Transformačný modul.....	42
11.4 Implementácia EM algoritmu.....	43
11.5 Implementácia algoritmu k-means.....	43
11.6 Popis funkcií v prostredí R.....	44
11.6.1 Funkcie na získavanie a prípravu dát.....	44
11.6.2 Funkcie na zhlukovanie.....	44
11.6.3 Funkcie na analýzu výsledkov.....	45
<b>12 Príloha B – Vybrané výsledky zhlukovania.....</b>	<b>46</b>
<b>13 Príloha C - Obsah priloženého CD.....</b>	<b>47</b>

# 1 Úvod

Hlavným zdrojom informácií je v súčasnosti pre moderného človeka nepochybne Internet. Ten je okrem iného tvorený úžasným množstvom hypertextových dokumentov t.j. dokumentov s odkazmi na iné dokumenty. Toto obrovské úložisko informácií je však tak málo organizované, že sa stáva pre človeka hľadajúceho konkrétne informácie takmer nepoužiteľné. Logicky teda vzniká požiadavka tento chaotický systém zorganizovať a to najlepšie automaticky.

Klastrovaním alebo zhlukovaním vo všeobecnosti rozumieme hľadanie skupín objektov s podobnými vlastnosťami. Táto práca sa obmedzuje na zhlukovanie dokumentov, ktoré majú podobný význam. O dokumentoch s rovnakým významom hovoríme, že patria do rovnakých tém. Identifikácia tém sa však z automatizačného hľadiska môže zdať ako veľmi problematická. Pochopenie súvislostí medzi slovami a dokumentami naozaj nie je ľahko algoritmizovateľná úloha. Jednej z techník, ako takéto témy identifikovať, sa venuje aj táto práca. Ide konkrétne o pravdepodobnostnú latentnú sémantickú analýzu, ktorá v praxi dosahuje dobré výsledky. Na základe tém je potom možné dokumenty zoradiť do skupín s podobným významom.

Cieľom tejto práce je vytvoriť softvérový balík na automatické zhlukovanie hypertextových dokumentov. Tento balík je potom potrebné formou experimentov overiť na netriviálnej množine dát, za ktorú bola zvolená časť voľne dostupnej, celosvetovej encyklopédie Wikipedia<sup>1</sup> a porovnať s výsledkami iného štandardného zhlukovacieho algoritmu.

---

<sup>1</sup> <http://www.wikipedia.org/>

## 2 Príprava dokumentov

### 2.1 Reprezentácia dokumentu

Algoritmy pre spracovávanie dokumentov, vyhľadávanie v nich, automatickú klasifikáciu či zhlukovanie používajú rôzne reprezentácie dokumentov. Táto kapitola opisuje reprezentácie dokumentov, ktoré predstavujú v súčasnosti základ pre problematiku automatického spracovania textov.

Reprezentáciu dokumentov najčastejšie tvoria jednotlivé slová vyskytujúce sa v danom dokumente. Tie sú vybrané buď človekom alebo automaticky. Pre každý dokument sa však vždy vytvorí množina slov, ktorá ho reprezentuje, pričom jednotlivé slová nazývame reprezentantmi dokumentu. Množina všetkých slov všetkých dokumentov sa nazýva **slovník**.

#### 2.1.1 Booleovský model

Asi najjednoduchším spôsobom reprezentácie dokumentu je práve booleovský model. Dokument je v ňom reprezentovaný ako vektor jednotiek a núl a každej pozícií v tomto vektore zodpovedá jedno slovo slovníka. Ak je teda vo vektore jednotka, znamená to, že sa zodpovedajúce slovo v slovníku v danom dokumente nachádza. Naopak, ak je vo vektore nula, dané slovo sa v dokumente nenachádza.

#### 2.1.2 Multimnožinový model

Zrejmým nedostatkom booleovského modelu je práve to, že nezohľadňuje frekvenciu výskytu slov v danom dokumente. Je totiž zrejmé, že dokument, v ktorom sa nejaké slovo vyskytuje veľa krát, bude lepším reprezentantom dokumentu ako slovo, ktoré sa v ňom vyskytuje iba raz. Dôsledkom tohto zjednodušenia môže byť nedostatočne presná reprezentácia dokumentu.

Tento problém sa snaží riešiť reprezentácia zvaná multimnožina slov, v ktorej nie je dokument reprezentovaný iba binárne zakódovanou prítomnosťou a absenciou slov zo slovníka, ale vektorom prirodzených čísel. Tieto čísla vyjadrujú počet výskytov (frekvenciu) slova v danom dokumente. Ak je teda vo vektore napríklad číslo päť,

znamená to, že zodpovedajúce slovo zo slovníka sa v dokumente nachádza práve päťkrát.

### 2.1.3 Vektorový model

Ani predchádzajúci model však nie je vždy úplne vhodný. Predpokladá sa v ňom totiž, že každé slovo má rovnaký podiel na reprezentácii dokumentu. Toto však vo všeobecnosti nemusí byť pravda. Je to spôsobené tým, že slovo, ktoré sa nachádza iba v jednom dokumente, je určite lepším reprezentantom dokumentu ako slovo, ktoré sa nachádza vo všetkých dokumentoch. V konečnom dôsledku to znamená, že slovo, ktoré sa nachádza vo všetkých dokumentoch, reprezentuje každý dokument rovnako a preto nemá z hľadiska reprezentácie jedného konkrétneho dokumentu žiadnu váhu<sup>2</sup>.

Najjednoduchším riešením tohto problému je započítať do váhy slova v dokumente aj jeho váhu v rámci celého slovníka.

Nech  $K=(k_1, k_2, \dots, k_n)$  je slovník všetkých slov. Potom dokument  $d_i$  môžeme reprezentovať ako vektor váh  $W_i=(w_{i1}, w_{i2}, \dots, w_{in})$ , ktoré zodpovedajú jednotlivým slovám z množiny  $K$ . Pričom

$$w_{ij}=tf(i, j) \cdot idf(j) \quad (1)$$

Kde  $tf(i, j)$  je frekvencia slova  $k_j$  v dokumente  $d_i$  a  $idf(j)$  je inverzná frekvencia slova  $k_j$  v dokumentoch. [Nav02]

Spôsobov, ako vypočítať  $idf$  existuje mnoho<sup>3</sup>, preto spomeňme len najpoužívanejší z nich.

$$idf(j)=\log\left(\frac{N}{df(j)}\right) \quad (2)$$

kde  $N$  je počet všetkých dokumentov a  $df(j)$  je počet dokumentov, v ktorých sa vyskytuje slovo  $k_j$ . [Joa97]

<sup>2</sup> V teórii kódovania sa používa častejšie pojem *informačný zisk*.

<sup>3</sup> Vo všeobecnosti  $tf$  patrí medzi tzv. spôsoby lokálneho váhovania a  $idf$  medzi spôsoby globálneho váhovania. [KO98]

Treba tiež zdôrazniť fakt, že pri každej z týchto množinových reprezentácií sa úplne zanedbáva poradie slov, pretože ide v podstate vždy len o množiny výskytov slov v dokumentoch bez ohľadu na to, či už sú váhované alebo nie. Napriek tomuto na prvý pohľad zdanlivo zásadnému zjednodušeniu sa takáto reprezentácia dokumentov v konkrétnych aplikáciach klasifikácie či vyhľadávania dokumentov podľa významu ukazuje ako postačujúca. Späťne teda môžeme tvrdiť, že na význam (sémantiku) dokumentu má hlavný vplyv množina slov dokumentu a ich poradie je až druhoradé.

## 2.2 Techniky zlepšovania reprezentácie dokumentov

Táto kapitola obsahuje ďalšie techniky, ktoré sa používajú na zlepšenie kvality reprezentácie dokumentov.

### 2.2.1 Transformácia na korene slov

Vo väčšine jazykov môžu mať slová rôzne prípony, čiže viacero tvarov. Málokedy však majú slová v rôznych tvaroch naozaj rôzny význam. Ukazuje sa, že tieto rôzne tvary majú približne rovnakú váhu reprezentácie v dokumentoch.

Toto tvrdenie sa dá demonštrovať na jednoduchom príklade. Zoberme dva dokumenty, ktoré budú predstavovať dve jednoduché vety.

*Petrova čiapka je zelená. Peter má zelenú čiapku.*

V oboch týchto vetách sa hovorí o fakte, že Peter má zelenú čiapku. Ak by sme však reprezentovali tieto vety klasickým spôsobom, dostali by slovník s ôsmimi rôznymi slovami a tak osem rôznych reprezentantov.

$K_1 = (\text{Petrova}, \text{čiapka}, \text{je}, \text{zelená}, \text{Peter}, \text{má}, \text{zelenú}, \text{čiapku})$

Pričom reprezentácia v booleovskom modeli bude:

$V_1 = (1, 1, 1, 1, 0, 0, 0, 0)$  a  $V_2 = (0, 0, 0, 0, 1, 1, 1, 1)$

Ak však slová, ktoré sú v inom tvare, budeme pokladať za jedného reprezentanta, dostaneme reprezentantov len päť. Reprezentácia viet v tomto modeli bude nasledovná:

$$K_2 = (\text{Peter}, \text{čiapka}, \text{je}, \text{zelená}, \text{má})$$

$$W_1 = (1, 1, 1, 1, 0) \text{ a } W_2 = (1, 1, 0, 1, 1)$$

Z vektorov  $W_1$  a  $W_2$  je zrejmé, že táto reprezentácia lepšie vystihuje význam vety. Obe vety hovoria v podstate o tom istom a aj ich váhové vektory sa líšia<sup>4</sup> minimálne.

Je veľmi výhodné považovať takéto rôzne tvary slov za jedného reprezentanta dokumentu. Reprezentácia sa tak dostáva bližšie k pravému významu dokumentu a nerozlišuje dokumenty zbytočne podľa toho, v akom tvare sa slová nachádzajú. Ďalšou výhodou je zníženie počtu reprezentantov v slovníku.

Jedným z najznámejších spôsobov ako prakticky realizovať nezávislosť na tvare slova je takzvaná transformácia na korene slov (angl. *stemming*<sup>5</sup>). Každé slovo dokumentu je pred pridaním do slovníka transformované na koreň (napríklad Porterovým algoritmom [Por97]) a až takýto koreň slova potom predstavuje reprezentanta dokumentu.

Jedným z problémov, ktorý sa však takýmto spôsobom nepodarí riešiť, je takzvaná **synonymita** slov. Ide o to, že viaceré slová môžu mať rovnaký význam, avšak nelíšia sa iba tvarom slova, ale môže ísť o úplne odlišné slová, napríklad *mať* a *vlastniť*. Problém vzniká ten istý, opäť budú v slovníku dvaja rôzni reprezentanti, ktorí menia význam dokumentov len minimálne.

---

4 Existujú rôzne metriky pre porovnanie podobnosti vektorov. Napríklad vzdialenosti Manhattan, Minkovski alebo kosínusová podobnosť.

5 *Stem* síce v angličtine znamená koreň alebo kmeň, ale v praxi ide väčšinou len o identifikátor, ktorý nemusí mať pre človeka žiadny význam. Naproti tomu existuje *lematizácia*, ktorá sa zaoberá hľadáním základného tvaru slova (prvý pád, jednotné číslo).

### 2.2.2 Vynechávanie stop slov

Jazyky obsahujú veľa slov, ktoré z hľadiska významu dokumentu nemajú takmer žiadnu reprezentatívnu váhu. Ide najmä o predložky, spojky, zvrtné zámená, atď. Takéto slová nazývame stop slová.

Je vhodné takéto slová odstrániť zo slovníka, pretože ak pre reprezentáciu významu dokumentu nemajú takmer žiadnu váhu, sú prakticky zbytočné. Keby však bola k dispozícii dostatočne veľká množina dokumentov, tak by takéto slová mali mať teoreticky veľmi nízke  $idf(j)$  (vzťah 2).

Keďže množina stop slov je pre jeden jazyk skoro fixná, je z praktického hľadiska vhodné tieto slová zo slovníka odstrániť automaticky a nezapočítavať ich tak do reprezentácie dokumentu.

Množiny stop slov pre rôzne jazyky sú väčšinou voľne dostupné na Internete.

### 2.2.3 Normalizácia dĺžky dokumentu

Ďalším problémom, ktorý vzniká pri reprezentácii dokumentov, je zohľadnenie dĺžky dokumentu. Je zrejmé, že ak sa v dvoch rôznych dokumentoch nachádza nejaké slovo napríklad desať krát, tak väčší význam má toto slovo pre ten kratší z nich. Riešením tohto problému je normalizácia dĺžky dokumentu. Asi najtriviálnejší spôsob, ako to dosiahnuť, je vydeliť funkciu  $tf(i, j)$  celkovým počtom slov v danom dokumente. Pre novú funkciu teda platí:

$$tf(i, j)' = \frac{tf(i, j)}{\sum_{j'=1}^n tf(i, j')} \quad (3)$$

Existujú aj iné, sofistikovanejšie metódy, založené napríklad na počítaní unikátnych reprezentantov dokumentu či maximálnej frekvencie reprezentantov v dokumente [SBM96].

#### 2.2.4 Váhovanie slov podľa logických častí dokumentu

Dokumenty ako také sú len zriedkavo úplne neštrukturované. Väčšinou sú rozdelené na isté logické časti.

Zoberme ako príklad vedecké práce, ktoré vo všeobecnosti obsahujú nadpis, abstrakt, samotné telo práce a bibliografické odkazy. Je rozumné predpokladať, že každá z týchto častí má inú váhu, vzhľadom na význam dokumentu. Slová nachádzajúce sa v nadpise budú mať logicky oveľa väčší význam ako napríklad slová v bibliografických odkazoch.

Taktiež značkovací jazyk HTML obsahuje značky (angl. *tags*), ktoré do istej miery definujú logickú štruktúru dokumentu. Obsahuje značky pre nadpisy (`title`, `h1`, `h2`, `h3`, ..., `h6`), citácie (`cite`, `blockquote`), rôzne úrovne zdôraznenia textu (`strong`, `em`) a iné.

Lepšiu reprezentáciu významu dokumentu môžeme potom získať tak, že pre každú takúto logickú časť dokumentu určíme koeficient, ktorým sa vynásobí pôvodná hodnota  $tf(i, j)$ . Môžeme tak priradiť napríklad väčšiu váhu slovám v nadpisoch ako slovám v bibliografických odkazoch. Koeficienty pre logické časti dokumentu sa určujú väčšinou empiricky.

## 2.3 Postup prípravy dokumentov

Táto kapitola zhŕňa postup prípravy dokumentov pre ďalšie spracovanie, ktorá sa skladá z týchto dvoch fáz:

1. fáza filtrácie
2. fáza ohodnotenia

V prvej fáze sú najskôr z dokumentu odstránené interpunkčné znamienka ako bodky, čiarky, bodkočiarky a pomlčky. Zároveň sa všetky veľké písmená prevedú na malé písmená a dokument sa rozdelí na slová. Proces rozdelenia dokumentu na slová sa nazýva **tokenizácia**. Z týchto slov sa následne vynechajú tie, ktoré sú v množine stop slov. Ostatné slová sa prevedú na korene. Takáto množina reprezentantov (koreňov slov) dokumentu prechádza do druhej fázy.

Vo fáze ohodnotenia sa vytvára váhový vektor dokumentu. Najprv sa vypočítajú frekvencie slov v dokumentoch, kde treba brať do úvahy aj koeficient logickej časti dokumentu, v ktorej sa daný reprezentant nachádza. Následne je vykonaná normalizácia dĺžky dokumentu, ktorá tieto koeficienty finálne upraví. Takýto váhový vektor je potom uložený na ďalšie spracovanie.

### 3 Zhlukovanie dokumentov

Pod zhľukovaním alebo klastrovaním všeobecne rozumieme hľadanie skupín objektov s podobnými charakteristikami. Aplikované na opisovanú problematiku dokumentov to znamená hľadanie skupín dokumentov s podobným významom. Zhľukovanie má veľký význam pri automatickej kategorizácii či inteligentnom vyhľadávaní.

Algoritmy slúžiace na zhľukovanie je možné rozdeliť podľa princípu, na ktorom sú založené, do dvoch najznámejších skupín:

1. **štatistické** alebo **pravdepodobnostné** – výsledkom týchto algoritmov sú väčšinou tabuľky pravdepodobností, vyjadrujúce, s akou pravdepodobnosťou daný objekt patrí do daného zhľuku.
2. **vzdialenostné** – porovnávajúce objekty rôznymi vzdialenostnými metrikami. Tento typ algoritmov sa snaží nájsť také rozdelenie objektov do zhľukov, aby vzdialenosti objektov v rovnakom zhľuku boli minimálne a naopak vzdialenosti objektov z rôznych zhľukov maximálne.

Výsledok zhľukovania je možné kategorizovať podľa charakteru zhľukov:

1. **neprekrývajúce sa zhľuky** – každý objekt je práve v jednom zhľuku.
2. **prekrývajúce sa zhľuky** – objekt môže byť aj vo viacerých zhľukoch naraz.

V tejto kapitole je popísaný jeden zo štatistických modelov zvaný **aspektový model**. Parametre tohto modelu je však potrebné určiť iteračným algoritmom maximalizácie vierohodnosti zvaným **EM algoritmus**, ktorý popisuje prvá časť tejto kapitoly. Pomocou takto získaných parametrov je následne možné výsledok rôzne spracovať, pričom jednou z možností je práve samotné zhľukovanie. Ide o pravdepodobnostný algoritmus, ktorého výsledkom sú prekrývajúce sa zhľuky.

V ďalšej časti tejto kapitoly je popísaný jeden z najznámejších a najčastejšie používaných vzdialenostných zhľukovacích algoritmov hľadajúci neprekrývajúce sa zhľuky. Ide o štandardný zhľukovací algoritmus k-means, pričom je následne navrhnutá jeho úprava, ktorá umožňuje brať do úvahy aj odkazy medzi dokumentami.

### 3.1 EM algoritmus

Nech  $\theta$  sú také hodnoty parametrov, pre ktoré model maximalizuje  $P(X|\theta) = \prod_x p(x|\theta)$ . Pre odhad hodnôt  $\theta$  sa zvykne zavádzať funkcia logaritmickej vierohodnosti

$$L(\theta) = \ln P(X|\theta) \quad (4)$$

EM algoritmus je iteratívna procedúra pre maximalizáciu tejto vierohodnosti. Nech po  $n$ -tej iterácii je aktuálny odhad parametrov  $\theta_n$ . Hľadáme teda také parametre, ktoré maximalizujú funkciu  $L(\theta)$ . Musí teda platiť, že

$$L(\theta) > L(\theta_n) \quad (5)$$

Čiže vlastne hľadáme také parametre  $\theta$ , ktoré maximalizujú rozdiel

$$L(\theta) - L(\theta_n) = \ln P(X|\theta) - \ln P(X|\theta_n) \quad (6)$$

EM algoritmus v tomto momente zavádza do modelu skryté premenné  $Z$  tak, aby sa uľahčila maximalizácia funkcie vierohodnosti. Pre pravdepodobnosť  $P(X|\theta)$  teda platí

$$P(X|\theta) = \sum_{z \in Z} P(X|z, \theta) P(z|\theta) \quad (7)$$

Vzťah 6 je potom možné prepísať ako

$$L(\theta) - L(\theta_n) = \ln \sum_{z \in Z} P(X|z, \theta) P(z|\theta) - \ln P(X|\theta_n) \quad (8)$$

Za pomoci Jensenovej nerovnosti

$$\ln \sum_{i=1}^n \lambda_i x_i \geq \sum_{i=1}^n \lambda_i \ln x_i \quad (9)$$

kde pre konštanty  $\lambda_i \geq 0$  platí

$$\sum_{i=1}^n \lambda_i = 1 \quad (10)$$

a po dosadení  $\lambda_z = P(z|X, \theta_n)$  dostávame

$$L(\theta) - L(\theta_n) \geq P(z|X, \theta_n) \ln \frac{P(X|z, \theta) P(z|\theta)}{P(z|X, \theta_n) P(X|\theta_n)} = \Delta(\theta|\theta_n) \quad (11)$$

Parametre  $\theta_{n+1}$  v ďalšej iterácii majú maximalizovať tento rozdiel, čo znamená

$$\theta_{n+1} = \arg \max_{\theta} \left\{ L(\theta_n) + \sum_{z \in Z} P(z|X, \theta_n) \ln \frac{P(X|z, \theta) P(z|\theta)}{P(z|X, \theta_n) P(X|\theta_n)} \right\} \quad (12)$$

z čoho po úprave dostávame

$$\theta_{n+1} = \arg \max_{\theta} \left\{ \sum_{z \in Z} P(z|X, \theta_n) \ln (P(X|z, \theta) P(z|\theta)) \right\} \quad (13)$$

Tento algoritmus teda pracuje v nasledovných dvoch krokoch.

V prvom kroku (E-krok) sa vypočítajú očakávané hodnoty skrytých premenných  $P(z|X, \theta_n)$  pomocou dát  $X$  a aktuálnych parametrov  $\theta_n$ . Tieto hodnoty sa väčšinou vypočítajú pomocou Bayesovho pravidla nasledovne

$$P(z|X, \theta_n) = \frac{P(z|\theta_n) P(X|z, \theta_n)}{\sum_{z \in Z} P(z|\theta_n) P(X|z, \theta_n)} \quad (14)$$

V druhom kroku (M-krok) sa vypočítajú také parametre  $\theta_{n+1}$ , ktoré maximalizujú logaritmicke funkciu vierohodnosti pomocou dát  $X$  a očakávaných skrytých premenných  $P(z|X, \theta_n)$ .

Každá iterácia tohto algoritmu zvyšuje hodnotu vierohodnosti, takže zaručené je iba dosiahnutie jej lokálneho maxima [Bor04].

## 3.2 Aspektový model

Tento model je východiskom pre **pravdepodobnostnú latentnú sémantickú analýzu**.

Nech  $N_{ij}$  je frekvencia slova  $t_i$  v dokumente  $d_j$ , potom predikovaná pravdepodobnosť výskytu slova  $t_i$  v dokumente  $d_j$  je

$$P(t_i|d_j) = \sum_k P(t_i|z_k)P(z_k|d_j) \quad (15)$$

Kde  $1 \leq k \leq K$  a  $z_k$  nazývame **faktor** alebo **aspekt**. Pričom musí platiť

$$\sum_i P(t_i|z_k) = 1 \quad (16)$$

pre všetky  $k$  a zároveň

$$\sum_k P(z_k|d_j) = 1 \quad (17)$$

pre všetky  $j$ .

Ako možno vidieť zo vzťahu 15, pravdepodobnosti výskytu slov v dokumentoch sú modelované zaujímavým spôsobom. Každé slovo  $t_i$  má určitú pravdepodobnosť výskytu  $P(t_i|z_k)$  v latentnej triede  $z_k$ . Každá táto trieda má určitú pravdepodobnosť  $P(z_k|d_j)$  výskytu v danom dokumente  $d_j$ . Tieto pravdepodobnosti vyjadrujú, do akej miery dané slovo popisuje určitý aspekt a do akej miery daný aspekt popisuje určitý dokument<sup>6</sup>.

Problém, ktorý vzniká, je spôsob výpočtu týchto pravdepodobností a na jeho riešenie je možné použiť práve EM algoritmus.

---

<sup>6</sup> Tento model sa všeobecne používa na takzvané dyadické (dvojčlenné, párové) dáta. Nemusí teda v zásade vždy ísť len o výskyt slov v dokumente [HP98, GH99].

Ako funkciu vierohodnosti zoberme

$$L = \sum_{i,j} N_{ij} \log \sum_k P(t_i|z_k) P(z_k|d_j) \quad (18)$$

pričom pre E-krok platí

$$P(z_k|t_i, d_j) = \frac{P(t_i|z_k) P(z_k|d_j)}{P(t_i|d_j)} \quad (19)$$

a pre M-krok

$$P(t_i|z_k) = \sum_j \frac{N_{ij}}{\sum_{i'} N_{i'j}} P(z_k|t_i, d_j) \quad (20)$$

$$P(z_k|d_j) \propto \sum_i \frac{N_{ij}}{\sum_{i'} N_{i'j}} P(z_k|t_i, d_j) \quad (21)$$

Po dostatočne veľkom počte iterácii dosiahne algoritmus lokálne maximum funkcie vierohodnosti [Hof99a].

Treba si uvedomiť, že počet aspektov je v skutočnosti oveľa menší ako počet slov v slovníku a pretože dokument je reprezentovaný len kombináciou týchto aspektov, musí nutne dôjsť ku kompresii. Ukazuje sa, že tieto aspekty odrážajú určité skryté závislosti medzi slovami ako aj dokumentami.

Aspekty môžeme pokladať za určité charakteristické črty dokumentu a slová, ktoré majú veľkú pravdepodobnosť  $P(t_i|z_k)$ , najlepšie popisujú práve túto črtu dokumentu. V aspektoch sa takto zhlučujú slová, ktoré majú najväčší vplyv na význam daného aspektu. Môžeme ich považovať za akési kľúčové slová daného aspektu. Na druhej strane aj ku každému dokumentu dokážeme určiť dominantný aspekt (t.j. ten s najväčšou pravdepodobnosťou  $P(z_k|d_j)$ ). Takto vieme zaradiť dokumenty do kategórií a ide o jeden z najjednoduchších spôsobov zhlučovania dokumentov [Hof99b]. Ďalším spôsobom môže byť porovnávanie podobnosti dokumentov na základe všetkých pravdepodobností aspektov. Ku každému dokumentu  $d_j$  najprv vytvoríme vektor  $F_j = (P(z_1|d_j), P(z_2|d_j), \dots, P(z_k|d_j))$ , ktorý potom porovnávame rôznymi metrikami, čím dokážeme určiť mieru podobnosti jednotlivých dokumentov.

### 3.2.1 Zohľadnenie odkazov medzi dokumentami

Väčšina hypertextových dokumentov neexistuje samostatne, ale sú navzájom prepojené odkazmi. Tieto odkazy taktiež nesú určitú váhu vo význame dokumentu.

Nech  $A_{ij}$  je počet odkazov z dokumentu  $d_i$  na dokument  $d_j$  a naopak, potom môžeme analogicky k vzťahu 15 zadefinovať pravdepodobnosť toho, že dokument  $d_j$  odkazuje na dokument  $d_i$ .

$$P(c_i|d_j) = \sum_k P(c_i|z_k) P(z_k|d_j) \quad (22)$$

Po aplikovaní EM algoritmu je možné získať skryté závislosti v odkazovaní medzi dokumentami. Táto technika sa používa pre hľadanie autorít vzhľadom na daný dokument. Tieto dva spôsoby je však možné spojiť do jedného modelu, ktorý bude zohľadňovať ako obsah dokumentu, tak aj odkazovanie medzi nimi. Základom pre toto je úprava funkcie maximálnej vierohodnosti na

$$L = \sum_j \left[ \begin{aligned} & \alpha \sum_i \frac{N_{ij}}{\sum_{i'} N_{i'j}} \log \sum_k P(t_i|z_k) P(z_k|d_j) \\ & + (1-\alpha) \sum_l \frac{A_{lj}}{\sum_{l'} A_{l'j}} \log \sum_k P(c_l|z_k) P(z_k|d_j) \end{aligned} \right] \quad (23)$$

kde  $\alpha$  je relatívna váha slov voči odkazom dokumentu. Táto váha sa určuje experimentálne, pričom záleží na konkrétnej kolekcii dokumentov. Ukazuje sa, že vhodné hodnoty sú z intervalu (0.3, 0.5).

Pre E-krok sú pravdepodobnosti analogické vzťahu 19, ale pre M-krok je potrebné upraviť vzťah pre pravdepodobnosť  $P(z_k|d_j)$  tak, aby faktory odrážali aj skryté závislosti medzi odkazmi [CH01].

$$P(z_k|d_j) \propto \left[ \begin{aligned} & \alpha \sum_i \frac{N_{ij}}{\sum_{i'} N_{i'j}} P(z_k|t_i, d_j) \\ & + (1-\alpha) \sum_l \frac{A_{lj}}{\sum_{l'} A_{l'j}} P(z_k|c_l, d_j) \end{aligned} \right] \quad (24)$$

### 3.3 Zhľukovací algoritmus k-means

Nech  $d_i = (w_{i1}, w_{i2}, \dots, w_{in})$  je váhový vektor slov dokumentu  $i$ . Potom vzdialenosť medzi dvoma dokumentami  $i$  a  $j$  je definovaná ako Euklidovská vzdialenosť týchto dvoch vektorov.

$$d = \|d_i - d_j\| = \sqrt{\sum_{k=1}^n (w_{ik} - w_{jk})^2} \quad (25)$$

Nech  $S$  je neprázdna množina dokumentov, potom pod centroidom množiny  $S$  rozumieme vektor

$$\mu_S = \frac{\sum_{j \in S} d_j}{|S|} \quad (26)$$

Zhľukovací algoritmus k-means hľadá rozdelenie množiny dokumentov  $D$  na  $K$  disjunktných podmnožín  $S_i$ , kde  $1 \leq i \leq K$  pričom

$$\begin{aligned} S_i \cap S_j &= \emptyset, i \neq j \\ D &= S_1 \cup S_2 \cup \dots \cup S_K \end{aligned} \quad (27)$$

a zároveň toto rozdelenie minimalizuje  $J$ , t.j. sumu vzdialeností dokumentov od centroidov množín, v ktorých sa nachádzajú.

$$J = \sum_{j=1}^K \sum_{l \in S_j} \|d_l - \mu_{S_j}\| \quad (28)$$

Algoritmus hľadania podmnožín  $S_j$  sa skladá z náhodnej inicializácie vektorov centroidov<sup>7</sup> a dvoch krokov.

1. Každý dokument je zaradený do tej podmnožiny, od ktorej centroidu je jeho vzdialenosť najmenšia.
2. Vektory centroidov všetkých podmnožín sú pre nové rozdelenie dokumentov prepočítané podľa vzťahu 26.

Tieto dva kroky sa opakujú, kým nie je splnená ukončovacia podmienka, ktorou bežne býva konvergencia vektorov centroidov alebo dosiahnutie určitého počtu iterácií.

<sup>7</sup> Namiesto úplne náhodných vektorov sa pre jednoduchosť inicializácie môžu použiť aj váhové vektory náhodne vybraných dokumentov.

Vlastnosti tohto algoritmu síce nezaručujú dokonca ani dosiahnutie lokálneho minima  $J$ , avšak pre jednoduchosť implementácie je používaný najmä pri porovnávaní s inými algoritmami [WF05].

Výsledné podmnožiny dokumentov potom pokladáme za jednotlivé zhluky podobných dokumentov.

### 3.3.1 Zohľadnenie odkazov medzi dokumentami

Nech  $d_i = (w_{i1}, w_{i2}, \dots, w_{in})$  je vektor váh slov dokumentu  $i$ ,  $c_i = (c_{i1}, c_{i2}, \dots, c_{im})$  je vektor váh odkazov smerujúcich z dokumentu  $i$  a  $\alpha$  je relatívna váha slov voči odkazom. Potom vektor zohľadňujúci slová ako aj odkazy medzi dokumentami definujeme ako

$$v_i = (\alpha w_{i1}, \alpha w_{i2}, \dots, \alpha w_{in}, (1 - \alpha) c_{i1}, (1 - \alpha) c_{i2}, \dots, (1 - \alpha) c_{im}) \quad (29)$$

Takto upravený vektor dokumentu je možné použiť ako vstup pre zhlukovanie algoritmom k-means.

## 4 Špecifikácia požiadaviek

Cieľom tohto projektu je vytvoriť softvérový balík na automatické zhľukovanie hypertextových dokumentov, pričom tento balík je potrebné overiť formou experimentov na netriviálnej množine dát. Vychádzajúc z analýz, zhľukovanie dokumentov prebieha vo viacerých fázach, pričom tieto dáta treba najprv vhodne transformovať a až potom zhľukovať.

Dáta, ktoré budeme spracovávať, je možné získať rôznymi spôsobmi. Dajú sa využiť existujúce predpripravené množiny dát alebo vytvoriť vlastná množina. Ako ideálny kandidát pre tvorbu vlastnej množiny sa javí celosvetová encyklopédia Wikipedia, ktorá je voľne dostupná na Internete. Obsahuje kvalitný, jednotne formátovaný obsah, ktorý je veľmi dobre previazaný odkazmi.

Softvérový balík sa bude skladať z troch hlavných častí:

- zberač, ktorý bude schopný zozbierať a uložiť dáta z daného zdroja (v našom prípade Wikipedia).
- transformačný modul, ktorý pripraví reprezentáciu dokumentov vhodnú pre zhľukovanie.
- zhľukovací modul, ktorý vykoná samotné zhľukovanie dokumentov.

## 5 Návrh riešenia

### 5.1 Zberač

Tento modul bude automaticky ukladať dáta stránok Wikipedie, pričom bude vyhľadávať odkazy v týchto stránkach a následne spracovávať aj tie. Takto bude pracovať, až kým nezíska vhodne veľkú a kvalitnú množinu dát. Tieto dáta bude ukladať v jednoduchej forme na pevný disk.

### 5.2 Transformačný modul

Tento modul bude mať niekoľko častí. Najprv bude potrebné urobiť extrakciu dát zo získaných stránok, čiže odfiltrovať dáta netýkajúce sa priamo obsahu dokumentu. Pôjde najmä o rôzne navigačné prvky stránok. Keďže formát Wikipedie je pomerne jednotný, vidíme túto fázu ako jednoducho automatizovateľnú. Takto získané dokumenty je potrebné ešte raz filtrovať, aby sa odstránilo zbytočné formátovanie textov, avšak treba brať do úvahy, že niektoré logické časti dokumentu budú mať rôznu váhu a tieto časti vyznačiť. V tejto fáze bude taktiež vykonaná extrakcia odkazov medzi dokumentami.

Ďalšou časťou bude transformácia na váhový vektoru dokumentu. Najprv budú odstránené stop slová, potom budú Porterovým algoritmom ostatné slová transformované na koreň slova. Pri tejto transformácii sa súčasne vytvára slovník.

Bude tiež potrebné zvoliť váhy jednotlivých logických častí dokumentu, ako sú napríklad rôzne úrovne nadpisov a zohľadniť ich vo váhovom vektore dokumentu.

### 5.3 Zhlukovací modul

V tomto module bude implementovaný spomínaný aspektový model so zohľadnením odkazov medzi dokumentami. Následne sa vykoná samotné zhlukovanie niektorým z vhodných spôsobov: dominantný aspekt alebo k-means algoritmus.

## 5.4 Implementačné prostredia

Zberací a transformačný modul budú implementované skriptami v jazyku PHP<sup>8</sup>, pričom ako úložisko údajov bude použitá relačná MySQL<sup>9</sup> databáza. Samotné zhlukovanie bude realizované skriptami vo voľne dostupnom prostredí pre štatistické výpočty – R<sup>10</sup>.

---

8 <http://www.php.net/>

9 <http://www.mysql.com/>

10 <http://www.r-project.org/>

## 6 Opis riešenia

### 6.1 Zber a transformácia dát

Na zber dát z celosvetovej encyklopédie Wikipedia sme vytvorili niekoľko skriptov v jazyku PHP. Zberací skript začal na hlavnej stránke anglickej verzie encyklopédie<sup>11</sup>, odkiaľ postupne prechádzal objavené odkazy a ukladal ich do databázy. Výsledkom tohto procesu vznikla tabuľka 150 000 odkazov identifikovaných unikátnou adresou (URL), z čoho k 4 000 odkazom bol stiahnutý aj ich obsah. Ďalší skript zo získaných dokumentov vytvoril tabuľku previazania dokumentov odkazmi.

Z HTML kódu stiahnutých dokumentov boli najprv odstránené zbytočné informácie ako navigačné prvky, hlavička a pätička stránky. Z takto získaného obsahu dokumentu sa potom ešte odstránili aj zbytočné znaky ako čiarky, bodky, pomlčky, zátvorky a iné. Následne boli vynechané všetky značky HTML okrem `title`, `h1` až `h6`, `strong`, `em`, `a`, ktoré sa používali na váhovanie logických častí dokumentu. Takto získaný text sa rozdelil na slová, ktoré boli upravené na koreň voľne dostupným PHP skriptom Portero-rovho stemmera<sup>12</sup>. V tomto procese vznikol slovník koreňov slov a zároveň tabuľka výskytov týchto koreňov slova v dokumente. Ku každému výskytu slova bol navyše priradený aj zoznam HTML značiek, v ktorých sa slovo nachádza.

### 6.2 Úprava dát pre experimenty

Pred použitím zhlukovacích algoritmov na jednotlivé experimenty bolo potrebné zozbierané údaje ešte upraviť. Vytvorili sme skripty, ktoré majú slúžiť na zjednodušenie načítavania údajov z databáz a ich transformáciu do formátu vhodného pre jednotlivé zhlukovacie algoritmy.

Využitím naprogramovaných skriptov sa zozbierané dáta pripravovali nasledovne:

Z databázy sa najprv načítal určitý počet dokumentov a k nim prislúchajúce odkazy a slová, z ktorých sa vytvorili matice výskytov zodpovedajúce frekvencii výskytov. Frekvencia výskytu bola ešte ováňovaná heuristikou podľa logickej časti dokumentu.

<sup>11</sup> <http://en.wikipedia.org/>

<sup>12</sup> <http://www.chuggnutt.com/stemmer.php>

Váhové koeficienty (multiplikátory) heuristiky zvolené pre jednotlivé značky jazyka HTML sú v tabuľke 1.

<i>Značka</i>	<i>Multiplikátor</i>
em	1,5
strong, a	2
h6	2,5
h5	3
h4	3,5
h3	4
h2	5
h1	7
title	10

Tabuľka 1: Zvolené multiplikačné koeficienty vybraných značiek HTML dokumentu.

Celkový počet dokumentov závisel od charakteru experimentu, avšak vo všeobecnosti postačovalo 100 dokumentov. Vytvorená matica výskytov slov v dokumentoch potom obsahovala približne 15 000 unikátnych slov. Táto vysoká dimenzionalita však spôsobovala do istej miery interpretovanému prostrediu R značné výpočtové problémy. Ako vhodné riešenie sa ukázalo orezávanie matíc o slová a odkazy s nízkou alebo príliš vysokou frekvenciou výskytu v korpuse. Takéto slová majú z hľadiska reprezentácie dokumentu v rámci korpusu nízku váhu a preto ich môžeme bez veľkej straty presnosti odstrániť. (viď kapitola 2.1.3)

Z matice sa odstránili slová a odkazy, ktoré sa vyskytovali len v jednom dokumente a viac ako 20 dokumentoch. Takéto orezanie zmenšilo počet unikátnych slov približne na hodnotu 3000, pre ktorú už bolo možné vykonávať výpočty v prijateľnom čase.

Váhy v matici výskytov boli nakoniec upravené pomocou TFIDF váhovania opísaného v kapitole 2.1.3.

## 6.3 Zhlukovanie

Pri zhlukovaní dokumentov pomocou EM algoritmu sme príslušnosť dokumentu do zhľuku zisťovali na základe dominantného aspektu a ako relatívna váha slov voči odkazom  $\alpha$  bola použitá hodnota 0,5. Rovnaká hodnota bola použitá aj pri algoritme k-means.

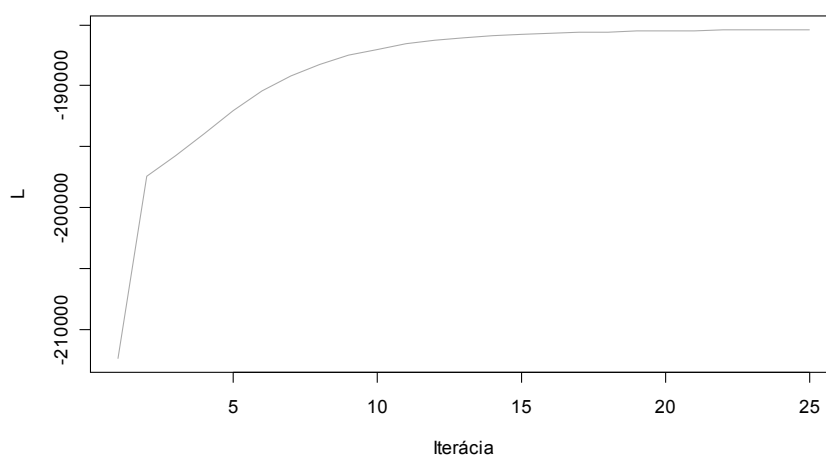
## 7 Výsledky experimentov

Táto kapitola sa zameriava na otestovanie vytvorených softvérových nástrojov voči teoretickým poznatkom získaným v analytickej časti tejto práce. Popisuje priebeh niekoľkých experimentov, pričom sa na základe teoretických znalostí snaží objasniť výsledky a ich závislosti od rôznych parametrov.

### 7.1 Priebeh maximalizácie vierohodnosti modelu

Iteračný EM algoritmus slúžiaci na maximalizáciu vierohodnosti modelu by mal po určitom počte iterácií uviaznuť v lokálnom maxime. Ukázalo sa, že na takéto uviaznutie pri nami testovanej vzorke dát stačilo v priemere 20 až 30 iterácií. Priebeh typickej maximalizácie vierohodnosti modelu je znázornený na obrázku 1.

Vo všetkých behoch tohto experimentu bolo možné pozorovať v prvej iterácii veľký skok vierohodnosti. Domnievame sa, že je to spôsobené tým, že parametre modelu sú na začiatku inicializované úplne náhodne a s veľkou pravdepodobnosťou je takýto náhodný model len veľmi slabou aproximáciou ideálneho modelu. Po prvej iterácii už však parametre modelu odrážajú určitú závislosť na vstupných dátach a preto je vierohodnosť omnoho vyššia.

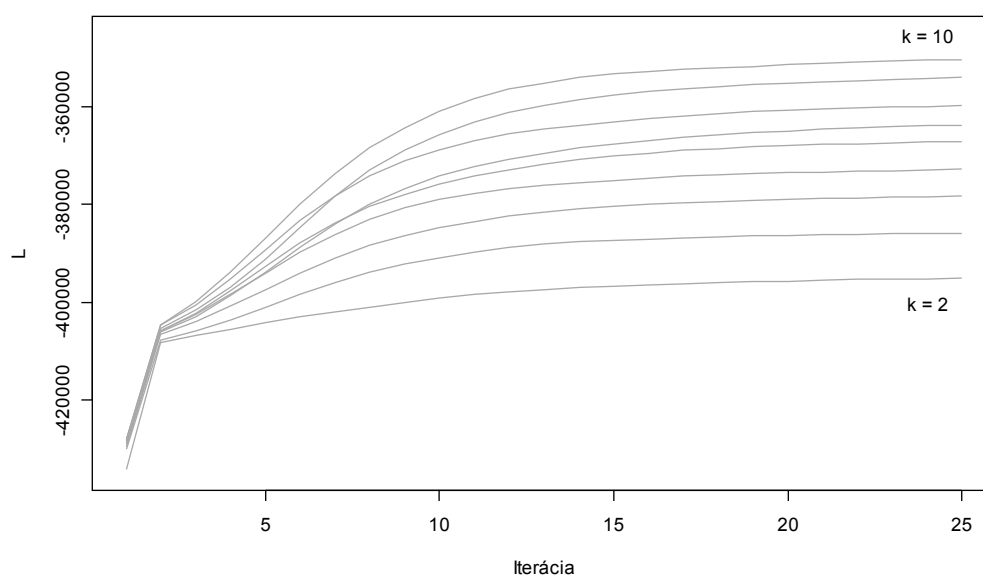


Obrázok 1: Priebeh vylepšovania vierohodnosti modelu počas behu EM algoritmu. Na osi y je znázornená veľkosť logaritmickej vierohodnosti modelu.

## 7.2 Zmeny počtu hľadaných aspektov

Kedže EM algoritmu je potrebné explicitne určiť počet hľadaných aspektov, na trénovacej vzorke 50 dokumentov sme sledovali, aký vplyv má tento počet na výslednú vierohodnosť modelu ako aj samotné zhlukovanie dokumentov vzhľadom na ich dominantný aspekt.

Na obrázku 2 je možné pozorovať priebeh vylepšovania vierohodnosti modelu pre počty aspektov 2 až 10. Všeobecne možno povedať, že vierohodnosť modelu narastá s počtom hľadaných aspektov. Je to spôsobené tým, že EM algoritmus takto získa väčší počet parametrov, jednotlivé aspekty sú viac špecifické a dokážu tak lepšie aproximovať model.

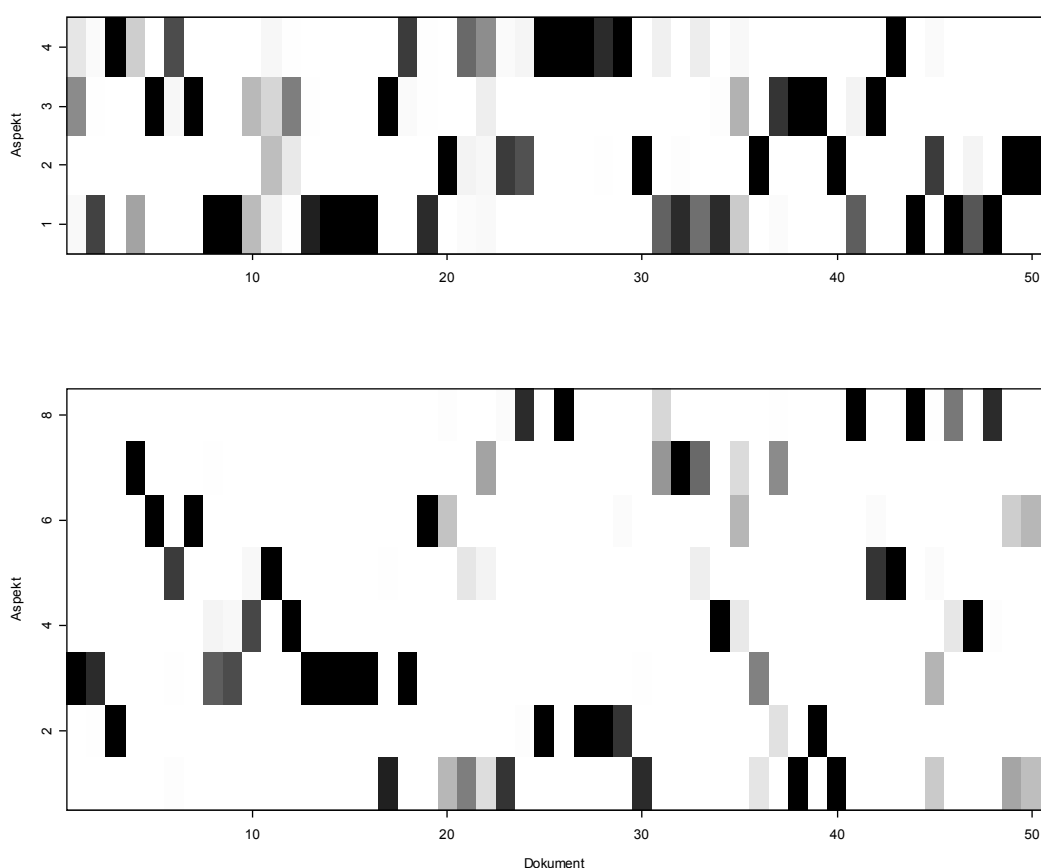


Obrázok 2: Vývoj zlepšovania vierohodnosti modelu pre rôzne počty hľadaných aspektov. Počty aspektov  $k$  sú 2 až 10, pričom logaritmickej vierohodnosť modelu s počtom aspektov stúpa.

Zvyšovanie špecifičnosti aspektov s ich narastajúcim počtom je možné demonštrovať na zmenách pravdepodobností  $P(z|d)$ . Matice týchto pravdepodobností boli vizualizované do tzv. **pravdepodobnostnej mapy**. V pravdepodobnostnej mape na ob-

rázku 3 sú na osi x vyznačené dokumenty, ktoré boli pre jednoduchosť očíslované, na osi y sú vyznačené jednotlivé aspekty. V takejto pravdepodobnostnej mape potom tmavosť farby vyznačuje veľkosť pravdepodobnosti  $P(z|d)$  pre dokument a aspekt daný jednotlivými súradnicami. Čierna farba je pravdepodobnosť blízka hodnote 1 a analogicky biela farba zase reprezentuje hodnotu blízku 0.

Ukazuje sa, že zatiaľ čo pri menšom počte aspektov patria dokumenty niekedy aj do viacerých aspektov, tak pri vyššom počte už patria väčšinou iba do jedného. Na obrázku 3 je to možné pozorovať napríklad na prvom dokumente, kde pri modeli so štyrmi aspektami patrí do aspektov 3 a 4. V modeli s 8 aspektami patrí však už len do aspektu 3. Model s 8 aspektami sa ukazuje byť viac špecifický ako model so 4 aspektami.



Obrázok 3: Mapa pravdepodobností  $P(z|d)$  pre počet aspektov 4 a 8. Tmavšia farba reprezentuje väčšiu hodnotu pravdepodobnosti.

Výsledky z experimentu ukazujú, že počet aspektov má veľký vplyv na výsledné zhľukovanie dokumentov. Logicky teda vzniká požiadavka toto výsledné zhľukovanie nejakým spôsobom systematicky ohodnotiť.

### 7.3 Kumulatívna entropia ako metrika kvality zhľukovania

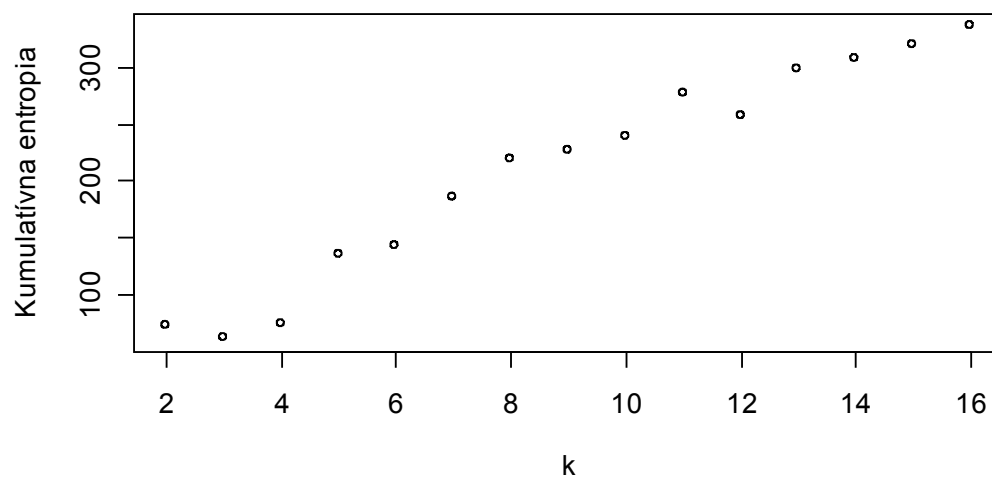
Hlavná idea navrhutej metriky vychádza z predpokladu, že pokiaľ je výsledok zhľukovania kvalitný, mal by byť každý dokument opísaný len jedným aspektom. Pri zhľukovaní formou dominantného aspektu sú totiž všetky ostatné aspekty zanedbané a preto je vhodné, aby v rámci opisu dokumentu niesli čo najmenšiu váhu. Keďže výsledkom zhľukovania sú aj pravdepodobnosti  $P(z|d)$ , je možné zistiť, s akou pravdepodobnosťou je daný dokument  $d_j$  opísaný daným aspektom. Potom vypočítaním entropie (vzťah 30) dostávame mieru neusporiadanosti dokumentu  $d_j$  k aspektom  $z_1, z_2, \dots, z_k$ . To znamená, že čím je táto entropia väčšia, tým je daný dokument opísaný aspektami menej jednoznačne.

$$H[P(z|d_j)] = - \sum_{i=1}^k P(z_i|d_j) \ln P(z_i|d_j) \quad (30)$$

Potom pre všetky dokumenty vypočítame kumulatívnu entropiu zo vzťahu

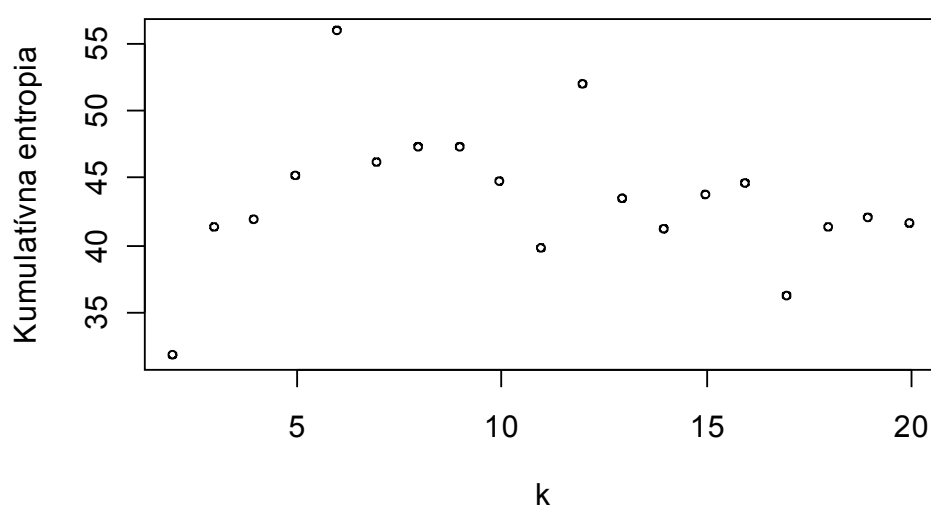
$$H = \sum_{j=1}^n H[P(z|d_j)] = - \sum_{j=1}^n \sum_{i=1}^k P(z_i|d_j) \ln P(z_i|d_j) \quad (31)$$

Túto metriku sme overili na syntetickej množine dát, pri ktorej bol počet zhľukov vopred známy. Množina obsahovala 100 slov v 348 dokumentoch a bola pomerne jednoznačne rozdelená na 4 kategórie dokumentov. Závislosť kumulatívnej entropie od počtu hľadaných aspektov je vyobrazená na obrázku 4. Ukazuje sa, že kumulatívna entropia je nízka pre počty hľadaných aspektov blízke skutočnému počtu kategórii. Pre vyššie počty hľadaných aspektov začne entropia pomerne rýchlo rásť. Je to spôsobené tým, že pridávaním aspektov sa nasilu rozdeľujú dokumenty pôvodne jednoznačne opísané jedným aspektom do dvoch menších zhľukov. Pravdepodobnosť výskytu dokumentu v oboch nových zhľukoch je približne rovnaká, čoho dôsledkom je zvýšenie kumulatívnej entropie.



Obrázok 4: Závislosť veľkosti kumulatívnej entropie od počtu hľadaných aspektov.

Tento experiment sme zopakovali aj pre údaje z encyklopédie Wikipedia. Pre testovaciu množinu 100 dokumentov bola vypočítaná kumulatívna entropia, ktorej závislosť od počtu aspektov je znázornená na obrázku 5.



Obrázok 5: Závislosť veľkosti kumulatívnej entropie od počtu hľadaných aspektov.

Ukázalo sa, že kumulatívna entropia je nízka práve pre počty aspektov, kde bolo zhlukovanie lepšie. Táto entropia má s narastajúcim počtom aspektov klesajúcu tendenciu, čo potvrdzuje hypotézu zvyšovania špecifičnosti aspektov spomínanú v 7.2. Toto správanie sa však líši od výsledkov z predchádzajúceho experimentu. Domnievame sa, že tento rozdiel je daný charakterom zvolených množín dokumentov. Zatiaľ čo v prvom experimente boli použité jednoznačné syntetické dáta, tak v druhom experimente reálne dáta odrážajú už oveľa komplikovanejšie závislosti a správanie kumulatívnej entropie nie je úplne zrejmé.

So zvyšujúcim sa počtom aspektov sa však zvyšuje aj riziko pretrénovania modelu. Tento problém sa ukázal pri ďalšom skúmaní výsledkov tohto experimentu. Pri zvyšovaní počtu aspektov vznikali síce disjunktné množiny dokumentov v aspektoch, avšak začali sa rozlišovať na základe oveľa jemnejších detailov. Napríklad skupina dokumentov o hudbe sa rozdelila na dve úplne disjunktné množiny, jedna o populárnej a druhá o folklórnej hudbe. Pri zhlukovaní na základe dominantného aspektu sa neskôr môže zdať, že tieto dva zhluky dokumentov nemajú nič spoločné.

Výsledkom týchto experimentov je záver, že výber vhodného počtu aspektov má veľký vplyv na samotnú kvalitu zhlukovania dokumentov. Keďže tento počet je daný väčšinou empiricky, vidíme to ako jeden z nedostatkov tohto prístupu ku zhlukovaniu dokumentov. Kvalitu výsledného zhlukovania je možné pre rôzne počty aspektov odmerať a porovnať navrhnutou metrikou kumulatívnej entropie.

## 7.4 Overenie správnosti výsledkov zhlukovania

Kvalitu výsledkov zhlukovania pomocou EM algoritmu bolo potrebné nielen odmerať navrhnutou formálnou metrikou, ale aj overiť správnosť tohto zhlukovania. Ako problémové miesto<sup>13</sup> sa však ukázala nedostupnosť testovacích množín, ktoré by brali do úvahy aj odkazy medzi dokumentami. Tento problém sme sa snažili vyriešiť tak, že sme sa pokúsili porovnať výsledky zhlukovania s iným zhlukovacím algoritmom. V našom prípade išlo o zhlukovací algoritmus k-means, ktorý bol podľa kapitoly 3.3 upravený tak, aby bral do úvahy aj odkazy medzi dokumentami.

---

<sup>13</sup> Bez referenčných výsledkov nie možné použiť štandardné metriky ako *precision*, *recall*, *fallout*, *accuracy*, *F-measure* [Lew91].

## 7.5 K-means verus EM algoritmus

Výsledky získané experimentáciou so zhlukovacím algoritmom k-means poukazovali na určitú zvláštnosť a nekorektnosť správania. Vo väčšine prípadov vznikal len jeden zhluk, ktorý obsahoval veľké množstvo dokumentov a ostatné zhľuky obsahovali naopak veľmi málo dokumentov. Ukázalo sa, že toto na prvý pohľad chybné správanie je charakteristické pre algoritmus k-means a je ho možné demonštrovať na jednoduchom príklade.

Nech  $C_1=(2,2,0,0,0)$  a  $C_2=(0,0,0,2,2)$  sú centroidy zhľukov  $C_1$ ,  $C_2$  a  $V=(1,1,0,1,1)$  je vektor dokumentu  $V$ .

Vzdialenosť vektora dokumentu  $V$  je potom pri použití Euklidovskej vzdialenosti od oboch centroidov rovnaká. Zatiaľ čo pri jednoduchých vstupoch nie je toto správanie takmer badateľné, pri vysoko dimenzionálnych vstupoch, akými textové dokumenty sú, je toto správanie veľmi nežiadúce. Je totiž zrejmé, že určité slová majú oveľa väčšiu váhu pri rozlišovaní kategórie, do ktorej dokumenty patria, avšak algoritmus k-means toto vôbec neberie do úvahy. Všetky slová v zhľukoch majú rovnakú váhu, čo má za následok to, že dokument nemusí byť zaradený do správneho zhľuku. Slová, ktoré rozhodujú o príslušnosti k danému zhľuku sú vďaka vysokej dimenzionalite vektorov prevážené viacerými slovami, ktoré nenesú takmer žiadnu relevantnú informáciu príslušnosti k zhľuku. Vzniká tak potom postupnou iteráciou algoritmu jeden priemerný centroid, ku ktorému má väčšina vektorov dokumentov najbližšie. Výsledkom je jeden veľký zhluk a zopár malých, reprezentovaných jedným či dvoma dokumentami.

EM algoritmus má oproti algoritmu k-means veľkú výhodu v tom, že jednotlivé slová môžu mať rôznu váhu pre každý zhluk. V zhľukoch počas iterácií algoritmu emergujú niektoré slová ako reprezentanti jednotlivých skupín dokumentov, ktoré slúžia ako pri rozhodovaní o príslušnosti do zhľukov.

Zhlukovacie algoritmy založené na klasických vzdialenostných metrikách nemôžeme porovnávať s EM algoritmom, pretože sa nedokážu vysporiadať s vysoko dimenzionálnym vstupným priestorom. Vzdialenostné metriky sú natoľko deformované vysokou dimenzionalitou, že výsledky sú takmer nepoužiteľné.

## 7.6 Diskusia

Výsledky zhukovania dokumentov na dátach encyklopédie Wikipedia pomocou EM algoritmu sme podrobnejšie skúmali. Ako základ sme zobrali výsledky zo zhukovania získaných z experimentov v rámci kapitoly 7.3. Konkrétne sme pozorovali výsledok pre 11 aspektov, pretože sa v ňom dosiahla nízka kumulatívna entropia pre nie príliš vysoký počet aspektov.

V tabuľke 2 sú uvedené niektoré<sup>14</sup> z dokumentov, ktoré boli zaradené do rovnakých zhukov, teda také, ktoré mali rovnaký dominantný aspekt. Dokumenty sú v tabuľke reprezentované nadpisom dokumentu. V zhuku 1 je možné pozorovať jasné sémantické prepojenie medzi dokumentami, ktoré sa všetky dotýkajú problematiky úžitkových zvierat. Zhuk 3 zase napríklad obsahuje dokumenty, ktoré sa týkajú folklórnej hudby pôvodom z Nigérie.

<i>Zhluk 1</i>	<i>Zhluk 2</i>	<i>Zhluk 3</i>	<i>Zhluk 4</i>
Domestic sheep	Alexander John Cuza <sup>15</sup>	Music of Nigeria	Wikipedia
Sheep	1959	Jùjú music	Free content
Moufflon	Wallachia <sup>16</sup>	Palm-wine music	Encyclopedia
Breed	Moldavia <sup>16</sup>	Apala	Wiki
Wool		Fuji music	Volunteer

Tabuľka 2: Ukážka časti výsledku zhukovania dokumentov pomocou dominantného aspektu.

Bližšie pochopenie vzniknutých zhukov je možné aj pomocou ďalšej analýzy slov alebo odkazov, ktoré najlepšie reprezentujú dané zhuky. Tabuľka 3 obsahuje odkazy zoradené zostupne podľa pravdepodobnosti  $P(c|z)$  pre jednotlivé zhuky. Zhuk 2 obsahuje jasné prepojenia na dokumenty týkajúce sa Rumunska, jeho oblastí

<sup>14</sup> Kompletný výpis dokumentov zhukovania je v prílohe B.

<sup>15</sup> Alexander John Cuza bol panovníkom v Rumunsku od roku 1959.

<sup>16</sup> Wallachia, Moldavia a Bessarabia sú oblasti v Rumunsku.

a panovníkov. Zhluk 4 je zase dobre reprezentovaný odkazmi z dokumentov týkajúcich sa samotnej encyklopédie Wikipedia.

<i>Zhluk 2</i>	<i>Zhluk 4</i>
List of state leaders in 1859	Jimmy Wales <sup>17</sup>
Romanians	List of encyclopedias
Kingdom of Romania	Internet
Romania	Everything2 <sup>18</sup>
Basarabia <sup>16</sup>	Criticism of Wikipedia
History of Romania	Wikimedia
Iași <sup>19</sup>	Wikimedia Foundation

Tabuľka 3: Odkazy zoradené zostupne podľa pravdepodobnosti výskytu v daných zhlukoch.

---

17 Jimmy Wales je zakladateľom encyklopédie Wikipedia.

18 Everything2 je menej známy, ale podobný projekt ako samotná Wikipedia.

19 Iași je mesto v Rumunsku.

## 8 Zhodnotenie

Táto práca sa zaoberá zhlukovaním hypertextových dokumentov na základe pravdepodobnostnej latentnej sémantickej analýzy. Na zhlukovanie dokumentov získaných z celosvetovej encyklopédie Wikipedia bol použitý iteračný EM algoritmus implementovaný v prostredí R. Táto práca formou experimentov overuje funkčnosť a vysvetľuje niektoré vlastnosti samotného EM algoritmu nad uvedenou množinou dokumentov.

K výsledkom našej práce patrí získanie pomerne veľkej množiny kvalitných dokumentov z encyklopédie Wikipedia, ktorá by mohla poslúžiť pre iné práce s podobným zameraním. K dispozícii sú dáta v nespracovanej, tak aj v predspracovanej forme v podobe invertovaného indexu. Boli vytvorené zhlukovacie a rôzne analyzačné skripty, pomocou ktorých boli vykonávané aj jednotlivé experimenty. Tie je možné znovu použiť pri ďalšom skúmaní vlastností EM algoritmu alebo na samotné zhlukovanie a analýzu dátových množín. Skripty boli otestované na množine syntetických ako aj reálnych dokumentov, kde sme formou experimentov sledovali vlastnosti algoritmu a zmeny jeho správania od rôznych parametrov.

K výhodám EM algoritmu patrí jeho jednoduchosť a dobré výsledky. Naopak k jeho nevýhodám patrí veľká výpočtová zložitosť vychádzajúca z jeho iteratívneho charakteru, ktorá je ešte umocnená faktom, že celú procedúru zhlukovania je potrebné pri každej zmene množiny dokumentov zopakovať. Využitie tohto spôsobu zhlukovania preto vidíme skôr v štatistickom spracovaní dát. Ďalšou nevýhodou je potreba dopredu empiricky určovať počet hľadaných zhlukov, ktorý však silne ovplyvňuje výslednú kvalitu zhlukovania. Táto práca prezentuje formálnu metriku kumulatívnej entropie, na základe ktorej je možné metodicky odmerať a porovnať kvalitu zhlukovania pre rôzne počty aspektov, čím sa problém s určením správneho počtu zhlukov do istej miery podarilo odstrániť.

Výsledky EM algoritmu sme sa následne pokúsili porovnať s iným štandardným zhlukovacím algoritmom. Ako referenčný algoritmus bol zvolený k-means algoritmus, ktorý bol upravený tak, aby bral do úvahy aj odkazy medzi dokumentami. Ukázalo sa, že tento algoritmus, ako aj iné založené na vzdialenostných metrikách, nedokážu do-

statočne dobre vyriešiť problém s vysokou dimenzionalitou dát. Pri vysokodimenzionálnych textových vstupoch sú výsledky natoľko zdeformované, že sa pre ďalšie použitie stávajú takmer nepoužiteľnými. Príčinu tohto problému sme identifikovali a demonštrovali na jednoduchom ukázkovom príklade.

## 9 Možnosti rozšírenia

Zhlukovanie pomocou pravdepodobnostnej latentnej sémantickej analýzy je možné robiť na párových (dyadických) dátach všeobecne. Nemusí ísť vždy len o slová či odkazy v dokumentoch, ako je tomu v tejto práci. Podobne by sa dali zhlukovať podobné produkty na základe množín nakúpených zákazníkmi a využiť tieto znalosti napríklad pri odporúčaníach pre nových zákazníkov (kolaboratívne filtrovanie).

V tejto práci sme pri zhlukovaní dokumentov vzali do úvahy nielen slová, ale aj odkazy medzi dokumentami. Tu vidíme priestor pre možné rozšírenia. V zhlukovaní by mohli byť zohľadnené používateľské preferencie, odporúčania na základe ostatných používateľov a iné. Vhodnou kombináciou váhovania by bolo možné takto spojiť ľubovoľné množstvo charakteristík ovplyvňujúcich výsledné zhlukovanie.

Samotný výsledok zhlukovania dokumentov je možné použiť na štatistické vyhodnocovanie ako aj pri vyhľadávaní dokumentov na základe slov. Vďaka skrytým závislostiam, ktoré sa odrážajú v jednotlivých aspektoch, dokážeme algoritmickejšie odhadnúť pravý význam hľadaných slov a tak značne zvýšiť efektivitu vyhľadávania. Keď zoberieme do úvahy aj odkazy medzi dokumentami a používateľské preferencie, výsledkom môže byť základ pre personalizované vyhľadávanie.

## 10 Použitá literatúra

- [Bor04] Sean Borman. The expectation maximization algorithm - a short tutorial, [http://www.seanborman.com/publications/EM\\_algorithm.pdf](http://www.seanborman.com/publications/EM_algorithm.pdf), Júl 2004.
- [CH01] David Cohn and Thomas Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. *Neural Information Processing Systems 13*, 2001.
- [GH99] D. Gildea and T. Hofmann. Topic-based language models using EM, 1999.
- [Hof99a] Thomas Hofmann. Probabilistic latent semantic indexing. *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, s. 50-57, Berkeley, California, August 1999.
- [Hof99b] Thomas Hofmann. Probabilistic latent semantic analysis. *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.
- [HP98] Thomas Hofmann and Jan Puzicha. Unsupervised learning from dyadic data. Technical Report TR-98-042, International Computer Science Institute, Berkeley, CA, 1998.
- [Joa97] Thorsten Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, s. 143-151, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.
- [KO98] Tamara G. Kolda and Dianne P. O'Leary. A semidiscrete matrix decomposition for latent semantic indexing information retrieval. *ACM Transactions on Information Systems*, 16(4):322-346, 1998.
- [Lew91] D. D. Lewis. Evaluating Text Categorization. *Proceedings of Speech and Natural Language Workshop*, s. 312-318. Morgan Kaufmann, 1991.
- [Nav02] Pavol Návrat et al. *Umelá inteligencia*, s. 339-352. Vydavateľstvo STU v Bratislave, prvé vydanie, 2002.

- [Por97] M. F. Porter. An algorithm for suffix stripping. s. 313-316, 1997.
- [SBM96] Amit Singhal, Chris Buckley, and Mandar Mitra. Pivoted document length normalization. In *Research and Development in Information Retrieval*, s. 21-29, 1996.
- [WF05] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. s. 137-138, 254. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition, Jún 2005.

## 11 Príloha A - Technická dokumentácia

Táto kapitola popisuje implementačné detaily jednotlivých modulov vytvoreného softvérového balíka.

### 11.1 Dátové štruktúry

Zozbierané dáta ako aj medzivýsledky sú uložené v MySQL databáze. Nasledujúca časť opisuje štruktúru jednotlivých tabuliek a im zodpovedajúcim údajom.

Pre zbierané stránky (dokumenty) bola navrhnutá nasledovná tabuľka `raw_pages`.

```
CREATE TABLE `raw_pages` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `url` varchar(255) NOT NULL default '',  
  `contents` text,  
  `is_valid` int(1) unsigned NOT NULL default '0',  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `url` (`url`),  
  KEY `is_valid` (`is_valid`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Kde `id` je číselný identifikátor stránky, `url` jednoznačný identifikátor stránky vo forme URL, `contents` je obsah získanej stránky a stĺpec `is_valid` slúži ako príznak identifikácie toho, či obsah stránky už bol získaný. Na rozdiel od ostatných tabuliek bol zvolený typ `InnoDB`, ktorý dokáže pracovať s transakciami a to z dôvodu spustenia viacerých inštancií zberača súčasne.

Kvôli rýchlejšiemu ďalšiemu spracovaniu dokumentov bola vytvorená takmer identická tabuľka typu `MYISAM`, ktorá obsahuje kópiu zozbieraných dokumentov.

```
CREATE TABLE `documents` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `url` varchar(80) NOT NULL default '',  
  `contents` text,  
  `was_processed` int(1) NOT NULL default '0',  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `url` (`url`),  
  KEY `was_processed` (`was_processed`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Tabuľka `documents` sa od tabuľky `raw_pages` líši pomocným slúpcom `was_processed`, ktorý slúži ako príznak toho, či už bol obsah stránky ďalej transformovaný.

Tabuľka `links` obsahuje previazanie dokumentov odkazmi a jej štruktúra je nasledovná:

```
CREATE TABLE `links` (  
  `source_id` int(10) unsigned NOT NULL default '0',  
  `target_id` int(10) unsigned NOT NULL default '0',  
  `count` int(10) unsigned NOT NULL default '1',  
  KEY `source_id` (`source_id`),  
  KEY `target_id` (`target_id`)  
) ENGINE=MyISAM;
```

Kde `source_id` je cudzím kľúčom smerujúcim na `documents.id` a je identifikátorom dokumentu, v ktorom sa odkaz nachádza. Stĺpec `target_id` je tiež cudzím kľúčom na `documents.id` a je identifikátorom dokumentu, na ktorý odkaz smeruje. Stĺpec `count` obsahuje počet rovnakých odkazov v rámci jedného dokumentu.

Slová slovníka sú ukladané do tabuľky `terms` so štruktúrou

```
CREATE TABLE `terms` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `term` varchar(45) NOT NULL default '',  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `word` (`term`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Kde `id` je identifikátor slova a `term` je samotné slovo. Dĺžka slova bola obmedzená na 45 znakov.

Výskyt slov v dokumentoch je uložený vo forme invertovaného indexu reprezentovaného tabuľkou `occurrences` s nasledovnou štruktúrou.

```
CREATE TABLE `occurrences` (  
  `term_id` int(11) NOT NULL default '0',  
  `document_id` int(11) NOT NULL default '0',  
  `modifiers` set('title', 'strong', 'em', 'h1', 'h2',  
'h3', 'h4', 'h5', 'h6', 'a') default NULL,  
  KEY `term_id` (`term_id`),  
  KEY `document_id` (`document_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Kde `term_id` je identifikátor slova a cudzí kľúč na `terms.id`, `document_id` je identifikátor dokumentu, v ktorom sa slovo nachádza a cudzí kľúč na `documents.id` a stĺpec `modifiers` obsahuje zoznam značiek, v ktorých sa slovo nachádza.

Pomocné tabuľky `term_aspect_probabilities`, `link_aspect_probabilities` a `aspect_document_probabilities` slúžia na ukladanie výsledkov EM algoritmu a ich štruktúra je nasledovná.

```
CREATE TABLE `term_aspect_probabilities` (  
  `term_id` int(10) unsigned NOT NULL default '0',  
  `aspect_id` int(10) unsigned NOT NULL default '0',  
  `probability` double unsigned NOT NULL default '0',  
  PRIMARY KEY (`term_id`, `aspect_id`)  
) ENGINE=MyISAM;
```

```
CREATE TABLE `link_aspect_probabilities` (  
  `link_id` int(10) unsigned NOT NULL default '0',  
  `aspect_id` int(10) unsigned NOT NULL default '0',  
  `probability` double unsigned NOT NULL default '0',  
  PRIMARY KEY (`link_id`,`aspect_id`)  
) ENGINE=MyISAM;  
CREATE TABLE `aspect_document_probabilities` (  
  `aspect_id` int(10) unsigned NOT NULL default '0',  
  `document_id` int(10) unsigned NOT NULL default '0',  
  `probability` double unsigned NOT NULL default '0',  
  PRIMARY KEY (`aspect_id`,`document_id`)  
) ENGINE=MyISAM;
```

Kde `term_id` je identifikátor slova a cudzí kľúč do `terms.id`, `aspect_id` je číslenny identifikátor aspektu, `link_id` je identifikátor odkazu a cudzí kľúč do `documents.id`, `document_id` je identifikátor dokumentu a cudzí kľúč do `documents.id` a `probability` je pravdepodobnosť vyjadrená číslom z intervalu  $\langle 0,1 \rangle$ .

## 11.2 Zberač

Zberacím modul bol implementovaný v jazyku PHP skriptom `heracles.php`. Tento skript načítava po jednom riadku z tabuľky `raw_pages`, pre ktoré je `is_valid` rovné nule. Pre danú adresu nastaví `is_valid` na 1, z adresy `url` získa obsah a uloží ho. Taktiež z tohto obsahu získa odkazy s prefixom `/wiki/`<sup>20</sup> a nové z nich uloží do tabuľky ako nespracované s príznakom `is_valid` rovným nule. Na začiatku zberacieho procesu je potrebné do tabuľky vložiť adresu začiatkovej stránky (v našom prípade to bola `/wiki/Wikipedia`) s nastaveným príznakom `is_valid` na nulu.

---

<sup>20</sup> Prefixom `/wiki/` sa začínajú všetky adresy stránok encyklopédie Wikipedia, ktoré obsahujú čitateľný obsah. Obrázky, špeciálne kategórie, externé odkazy a iné nevhodné stránky sa tak odfiltrujú.

### 11.3 Transformačný modul

Transformačný modul je tvorený niekoľkými skriptami v jazyku PHP.

Skript `linker.php` naplňa tabuľku `links`, ktorá obsahuje prepojenia dokumentov odkazmi. Ten postupne prechádza dokumenty tabuľky `documents` s príznakom `was_processed` rovným nule, pričom zo samotného obsahu získava odkazy identickým spôsobom ako zberací skript `heracles.php`.

Skript `cleaner.php` naplňa tabuľku `occurrences`, ktorá obsahuje výskyty slov v dokumentoch. Postupne prechádza dokumenty tabuľky `documents` s príznakom `was_processed` rovným nule pričom zo samotného obsahu dokumentu sa zachovávajú iba dva fragmenty HTML kódu. Prvý sa nachádza medzi HTML párovými značkami `title` a predstavuje nadpis, druhý predstavujúci obsah je medzi HTML komentármi `<!-- start content -->` a `<!-- end content -->`. Následne sú odstránené interpunkčné znamienka (bodky, čiarky, pomlčky,...), špeciálne znaky (tabulátory, HTML entity, zátvorky,...) a všetky HTML značky okrem `title`, `h1-h6`, `strong`, `em`, a. Získaný reťazec obsahuje už len čistý text dokumentu a povolené HTML značky. Tento reťazec je rozdelený na slová, ktoré sú prevedené na malé písmená. Slová sú upravené na koreň voľne dostupným Porterovým stemmerom (`stemmer.php`) a vložené do slovníka (tabuľka `terms`). Pre každé slovo je vytvorený riadok v tabuľke `occurrences`, pričom tento riadok obsahuje identifikátor slova, identifikátor dokumentu a zoznam HTML značiek, v ktorých sa slovo nachádza.

Pomocný skript `storage.php` zabezpečuje perzistenciu a získavanie údajov do/z MySQL databázy vo forme jednoduchej triedy `Storage`. Pomocný skript `common.php` obsahuje spoločné funkcie niektorých skriptov ako získanie obsahu dokumentu `get_wiki_contents`, poľa odkazov z daného HTML fragmentu `get_urls` a jednoduchý filter odkazov `strip_special_urls`, ktoré začínajú prefixom `/wiki/`.

## 11.4 Implementácia EM algoritmu

EM algoritmus bol implementovaný v prostredí R ako funkcia `aspects`. Algoritmus pracuje v nasledovných krokoch:

1. Koeficienty matíc parametrov  $P(t|z)$ ,  $P(c|z)$  a  $P(z|d)$  sa inicializujú na náhodné čísla z intervalu  $\langle 0,1 \rangle$ , ktoré sa následne normalizujú podľa vzťahov 16 a 17. Zodpovedajúce premenné týmto maticiam parametrov sú `termAspect`, `linkAspect` a `aspectDocument`.
2. Vytvoria a na nulu sa inicializujú matice parametrov  $P(z|t,d)$  a  $P(z|c,d)$  ako premenné `termAspectDoc` a `linkAspectDoc`.
3. Iterácia algoritmu začína výpočtom pravdepodobností  $P(t|d)$ ,  $P(c|d)$  zo vzťahu 15 a ich uložením do pomocných premenných `td` a `cd`.
4. Vypočíta sa logaritmická vierohodnosť zo vzťahu 23 a uloží sa do poľa `likelihoods` s indexom aktuálnej iterácie. Ak je rozdiel posledných dvoch vierohodností menší ako dané `epsilon`, tak sa pokračuje krokom 8.
5. Pomocou vzťahu 19 sa prepočítajú parametre  $P(z|t,d)$  a  $P(z|c,d)$ . E krok.
6. Pomocou vzťahov 20 a 24 sa prepočítajú pravdepodobnosti  $P(t|z)$ ,  $P(c|z)$  a  $P(z|d)$  a následne normalizované podľa vzťahov 16 a 17.
7. Ak nebol dosiahnutý cieľový počet iterácií, pokračuje sa krokom 3.
8. Vypočítané parametre a priebeh logaritmickkej vierohodnosti sa vráti ako výsledok.

## 11.5 Implementácia algoritmu k-means

Zhlukovací algoritmus k-means bol implementovaný ako funkcia `kmeans_data` pomocou funkcie `kmeans`, ktorá je štandardne dostupná v prostredí R. Výstup funkcie bol upravený tak, aby bol kompatibilný s výstupom funkcie EM algoritmu `aspects` pre jednoduchosť porovnania výsledkov.

## 11.6 Popis funkcií v prostredí R

Táto kapitola obsahuje popis hlavných funkcií, slúžiacich ako základné rozhranie k vytvorenému zhlukovaciemu nástroju v podobe jednoduchého skriptovacieho jazyka. Funkcie sú rozdelené do troch kategórií podľa fázy, v ktorej sa používajú.

### 11.6.1 Funkcie na získavanie a prípravu dát

`create_connection(host, user, password, db)` – vytvorí spojenie s MySQL databázou `db` na adrese `host` pod užívateľom `user` a heslom `password`.

`load_data(con, maxId)` – načíta cez spojenie `con` dokumenty s identifikátorom menším alebo rovným ako `maxId` a vráti štruktúru dát na ďalšie spracovanie.

`prune_data(data, minSupport, maxSupport)` – oreže matice výskytov slova a odkazov v dokumentoch tak, aby sa v maticiach vyskytovali iba prvky, ktoré sú minimálne v `minSupport` dokumentoch a maximálne v `maxSupport` dokumentoch.

`tfidf_data(data)` – upraví matice výskytov TFIDF transformáciou.

### 11.6.2 Funkcie na zhlukovanie

`aspects(data, aspectsCount, iterationsCount, epsilon)` – vykoná EM algoritmus nad dátami, kde `aspectsCount` je počet hľadaných aspektov/zhlukov, `iterationsCount` je maximálny počet iterácií a `epsilon` je minimálna zmena logaritmickej vierohodnosti medzi dvoma iteráciami. Táto verzia počíta iba s výskytom slov v dokumentoch.

`aspects2(..., alpha)` – parametre ako `aspects` a navyše parameter `alpha`, ktorý určuje relatívnu váhu slov voči odkazom. Táto verzia EM algoritmu zohľadňuje aj odkazy medzi dokumentami.

`kmeans_data(data, centers, iter.max, nstart)` – vykoná zhlukovanie algoritmom k-means nad dátami, kde `centers` je počet centroidov, t.j. počet hľadaných zhlukov, `iter.max` je maximálny počet iterácií a `nstart` je počet štartov algoritmu. Táto verzia počíta iba s výskytom slov v dokumentoch.

`kmeans2_data(..., alpha)` – parametre ako `kmeans_data` a parameter `alpha`, ktorý určuje relatívnu váhu slov voči odkazom. Táto verzia algoritmu k-means zohľadňuje aj odkazy medzi dokumentami.

### 11.6.3 Funkcie na analýzu výsledkov

`graph_data(data)` – vykreslí mapy pravdepodobnostných matíc a priebeh vývoja logaritmickej vierohodnosti z výsledkov `data` EM algoritmu.

`top_terms(data, n)` – vypíše prvých `n` slov s najväčšou pravdepodobnosťou výskytu v jednotlivých aspektoch.

`top_links(data, n)` – vypíše prvých `n` odkazov s najväčšou pravdepodobnosťou výskytu v jednotlivých aspektoch.

`document_aspects(data)` – vypíše všetky dokumenty a číslo ich dominantného aspektu. Túto funkciu je možné použiť aj pre výsledky získané funkciou `kmeans_data`.

`save_data(con, data)` – uloží výsledky do databázy cez spojenie `con`.

## 12 Príloha B – Vybrané výsledky zhlukovania

<i>Zhluk</i>	<i>Dokumenty</i>
1	Popular music, Restoration spectacular, 1917 Constitution of Mexico, Mexico, Stadium, Domestic sheep, Sheep, Ruminant, Moufflon, Breed, Wool, Farmer, Crimp (wool)
2	1917, 1885, 1924, Royal Greenwich Observatory, February 3, List of historical anniversaries, Saudi Arabia, Denmark, Leather
3	Democratic Republic of the Congo, Congo Free State, National Revolutionary Front for the Liberation of Haiti, Gonaïves, 2004 Haiti rebellion, Iran, Egypt, Red Sea
4	February 5, 1859, Alexander John Cuza, Wallachia, Moldavia, February 4, February 2, Embassy, Deaths in 2006, January 15
5	Music of Nigeria, Highlife, Palm-Wine, Apala, Fuji music, Jùjú music, Yo-pop, BBC, Pound sterling, Pasig City stampede, Gameshow, Wowowee
6	Demographics of Nigeria, Nigeria, List of Belgian monarchs, Leopold II of Belgium, Africa, Football (soccer)
7	Wikipedia, Comet Hyakutake, Comic Relief, Charity, Syria, Lebanon, Quadru-ped, Main Page, World Wide Web, Free content, Encyclopedia, Wiki, Volunteer, Nupedia
8	Brazil, Adriaen van der Donck, Belgium, Norway
9	Principality, 1988, International Atomic Energy Agency, Nuclear program of Iran, Meat, Wikimedia Foundation, Wikipedia logo, Web browser, 2001
10	Cuba, Greenwich Time Signal, United Nations Security Council, Philippines, Jyllands, Multilingualism
11	Folk music, Bronze, Kingdom of Romania, United Kingdom, 2004, M/V al-Salam Boccaccio 98, Current events, Staple (textiles), IPA chart for English

Tabuľka 4: Výsledok zhlukovania 100 dokumentov encyklopédie Wikipedia pre 11 hľadaných aspektov.

## 13 Príloha C - Obsah priloženého CD

<i>Adresár</i>	<i>Popis</i>
/data/syntetic/	Syntetická testovacia množina dokumentov.
/data/wikipedia/	Nespracovaná ako aj predspracovaná množina dokumentov z encyklopédie Wikipedia.
/document/	Dokumentácia diplomovej práce.
/experiments/cluster-counts-50/	Výsledky experimentu pre množinu Wikipedia a počet dokumentov 50.
/experiments/cluster-counts-100/	Výsledky experimentu pre množinu Wikipedia a počet dokumentov 100.
/experiments/syntetic/	Výsledky experimentu pre syntetickú množinu.
/materials/	Použitá literatúra.
/sources/php/	Zdrojové súbory zberacieho a transformačného modulu.
/sources/R/	Zdrojové súbory EM algoritmu a analyzačných skriptov.
/tools/apache/	Apache server verzie 2.2.3 pre platformu Windows.
/tools/mysql/	MySQL server verzie 4.1 pre platformu Windows.
/tools/php/	PHP skriptovací jazyk verzie 5.2 pre platformu Windows.
/tools/R/	Prostredie R na štatistické výpočty.