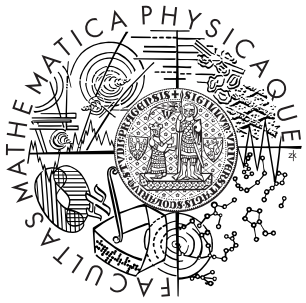
Univerzita Karlova v Prahe
Matematicko-fyzikálna fakulta

BAKALÁRSKA PRÁCA



Radoslav Klíč

Extrakcia kľúčových slov z dokumentov

Ústav formálnej a aplikovanej lingvistiky

Vedúci bakalárskej práce: Mgr. Pavel Pecina, Ph.D.
Študijný program: Informatika

2009

Na tomto mieste by som chcel poďakovať svojmu vedúcemu bakalárskej práce, doktorovi Pavlovi Pecinovi, za odbornú pomoc a jeho čas, ktorý mi ochotne venoval. Chcem tiež vyjadriť vďačnosť kolegovi a kamarátovi Martinovi Babkovi za podnetné rady a pomoc pri problémoch, ktoré sa pri práci vyskytli. Moja vďaka patrí aj ÚFALu, ktorý mi poskytol hosting pre môj projekt.

Prehlasujem, že som svoju bakalársku prácu napísal samostatne a výhradne s použitím citovaných prameňov. Súhlasím so zapožičiavaním práce a jej zverejňovaním.

V Prahe dňa 5. 8. 2009 Radoslav Klíč

Obsah

Názov práce: Extrakcia kľúčových slov z dokumentov
Autor: Radoslav Klíč
Katedra (ústav): Ústav formálnej a aplikovanej lingvistiky
Vedúci bakalárskej práce: Mgr. Pavel Pecina, Ph.D.
e-mail vedúceho: Pavel.Pecina@mff.cuni.cz

Abstrakt: Táto práca sa zaoberá problémom extrakcie kľúčových slov z dokumentov. Obsahuje stručný úvod do problematiky a opis niektorých prístupov k riešeniu tohoto problému. Jej súčasťou je implementácia niektorých opisovaných prístupov a ohodnotenie ich úspešnosti na základe kolekcie dokumentov. V rámci práce boli vytvorené dva softvérové nástroje. Jeden slúži na extrakciu kľúčových slov. Druhým je webové rozhranie k nemu. To poskytuje aj ďalšiu funkciu, ktorou je manuálne priraďovanie kľúčových slov k textom.

Kľúčové slová: extrakcia kľúčových slov, spracovanie prirodzeného jazyka

Title: Document Keyword Extraction
Author: Radoslav Klíč
Department: Institute of Formal and Applied Linguistics
Supervisor: Mgr. Pavel Pecina, Ph.D.
Supervisor's e-mail address: Pavel.Pecina@mff.cuni.cz

Abstract: In the present work, the problem of keyword extraction is studied. The work contains a brief introduction to the problem and description of some approaches to its solution. As a part of the work, some of the approaches are implemented and their efficiency is evaluated on a basis of a collection of documents. Two software tools are created. The first one's purpose is keyword extraction. The other one is a web-based interface for the first tool with one more function. It can be used for manual assigning of keywords to texts.

Keywords: keyword extraction, natural language processing

Kapitola 1

Úvod

1.1 Definícia problému a motivácia

Kľúčové slová pre dokument sú slová, ktoré vystihujú obsah daného dokumentu, vypovedajú, *o čom* je. To, že slová sú z dokumentu *extrahované*, znamená, že vybrané slová sa musia v texte nachádzať.

Informácia o kľúčových slovách dokumentu je užitočná predovšetkým pri vyhľadávaní. Relevancia výsledkov vyhľadávania je zvýšená, ak sú užívateľovi predkladané dokumenty, pre ktoré sú zadané termíny kľúčovými slovami. Existujú veľké kolekcie dokumentov bez priradených kľúčových slov, to platí aj pre veľkú časť obsahu na internete. Manuálne priraďovanie je časovo, a teda aj finančne, náročná úloha, preto je na mieste hľadať vhodný spôsob automatizácie tejto úlohy.

„Kľúčovosť“ slov však nemá jednoznačnú definíciu. Pre výber termínov je možné určiť isté pravidlá, ale človek sa pri výbere kľúčových slov riadi predovšetkým vlastnou intuíciou. Už tento fakt ilustruje, že automatizácia tohto procesu je netriviálny problém. Príbuzné problémy v oblasti spracovania prirodzeného jazyka sú napríklad automatická sumarizácia alebo získavanie kľúčových slov, nie nutne obsiahnutých v texte. Medzi ďalšie patrí automatická indexácia (napr. tvorba registra na konci publikácie) a automatická extrakcia terminológie.

Proces extrakcie kľúčových slov môžeme vo všeobecnosti rozdeliť do troch fáz:

1. Predspracovanie textu
2. Výber kandidátskych termov.
3. Ohodnotenie kľúčovosti kandidátov a selekcia najlepších.

Kľúčovou a najzaujímavejšou fázou je fáza ohodnotenia, keďže ide o jadro problému. Výkon algoritmu však môže byť pozitívne ovplyvnený aj vhodnou filtráciou termov vo fáze 2.

1.2 Práce v odbore

Problém extrakcie kľúčových slov a jemu príbuzné boli skúmané mnohými autormi. Čerpajúc predovšetkým z [2] spomenieme niektorých z nich. Základné myšlienky z oblasti boli vyslovené koncom 50. rokov (Luhn [4] a [5]). Sparck Jones v [6] skúma využívanie *idf*[1]. Informáciu o slovných druhoch využíva Earl v [1] pri automatickej indexácii. Extrahuje menné frázy a dôležitosť im priraďuje podľa frekvencie výskytu.

Neskôr sa začínajú objavovať aplikácie strojového učenia (s učiteľom). Podľa Hulthovej [2] bola táto metóda prvýkrát navrhnutá Turneym v [7]. Strojové učenie s učiteľom používa aj samotná Hulthová v [2]. Tá skúma tri prístupy k výberu kandidátskych termov. Pri prvom z nich sú vyberané všetky ngramy, ktoré nezačínajú ani nekončia stop-slovom[2]. Druhý a tretí prístup využívajú údaje o slovných druhoch. Pri druhom sú vybrané všetky menné frázy, ako ich označil špeciálny softvér (*NP-chunker*). Tretí prístup za kandidátov vyberá ngramy s vybranými kombináciami slovných druhov. Takto vzniknú tri sady učiacich dát. V dátach sú k ngramom priradené hodnoty týchto rysov: *tf*[1], *idf*, miesto prvého výskytu termu, slovné druhy. Na základe troch trénovacích sád sú natrénované tri predikčné modely, ktoré možu fungovať samostatne, alebo môžu byť skombinované. Ich kombináciou sa myslí algoritmus, ktorý vracia termy, na ktorých sa zhodnú aspoň dva z troch modelov (*majority vote*).

V tejto práci budeme používať tie isté rysy ako Hulthová. Ako základný prístup však zvolíme jednoduchý prístup *tfidf*, ktorý modifikujeme, aby využíval uvedené rysy.

Z českých prác na poli automatickej indexácie stojí za zmienku systém MOZAIKA, založený na francúzskom systéme MOSAIC. Ide o nástroj na extrakciu odbornej terminológie z textu. Bol vyvinutý na MFF UK v Prahe v 70. rokoch. Zajímavé je, že nepoužíva rozsiahle slovníky, ale kandidátske termy vyberá podľa prípon, ktorými odborné výrazy často končia. Zaujímavý je tiež spôsob prideľovania váh termom. Do úvahy je braná pozícia v texte s tým, že do úvahy sa berie napr. umiestnenie v nadpise či v prvej vete odstavca. Opis MOZAIKY podáva Z. Kirschner v [3].

[1] Viď 1.3
[2] Viď 2.2

1.3 Terminológia

V tejto sekcii sú zadefinované niektoré pojmy, používané v práci. Je tiež užitočné dať presnejší význam pojmu *slovo*, ktorý je relatívne vágny.

- Pojmom *slovo* budeme rozumieť postupnosť znakov oddelenú aspoň jednou medzerou. Pojmom *term* alebo *ngram* budeme rozumieť usporiadanú n-ticu slov. Termín *kľúčové slovo* budeme používať vo význame kľúčový term.

- *tf (term frequency)* znamená počet výskytov termu v dokumente. Tento počet sa normalizuje voči počtu termov v dokumente pre porovnateľnosť v dokumentoch s rôznou dĺžkou.

  $$\mathrm{tf}(t) = \frac{f_t}{n}$$

  kde f_t je počet výskytov termu t a n je počet všetkých termov v dokumente.

- *idf (inverse document frequency)* – inverzná frekvencia výskytu termu v dokumentoch je definovaná nasledovne:

  $$\mathrm{idf}(t) = \log\left(\frac{n}{f_t}\right)$$

  kde f_t je počet dokumentov, v ktorých sa vyskytuje term t a n je počet všetkých dokumentov.

1.4 Štruktúra práce

V prvej kapitole bol definovaný skúmaný problém a spomenuté niektoré predchádzajúce práce v odbore. Kapitola 2 opisuje predspracovanie textu a výber kandidátskych termov. V kapitole 3 sa venujeme kolekcii dokumentov používanej v práci. Je tam opísaný jej obsah a spôsoby využitia. Prístup *tfidf* je rozoberaný v kapitole 4. Nachádza sa v nej opis prístupu a dvoch jeho modifikácií a krátky opis ich vplyvu na úspešnosť extrakcie. Kapitola 5 je venovaná metodike vyhodnocovania úspešnosti. Sú tam tiež uvedené podrobné výsledky experimentov. V kapitole 6 nájdeme záverečnú diskusiu o použitom riešení, jeho nedostatkoch a možných vylepšeniach. Dodatky A a B obsahujú používateľskú dokumentáciu k softvérovým nástrojom vyvinutým v rámci práce. Programátorská dokumentácia je v dodatkoch C a D.

Kapitola 2

Prístup k podproblémom

Táto kapitola sa venuje problémom, ktoré sa priamo netýkajú hodnotenia kľúčovosti termov. Skôr, ako sa ním začneme zaoberať, je potrebné sa rozhodnúť, do akej podoby bude vstupný text predspracovaný a aké entity budú predstavovať kandidátov na kľúčové slová. Teda zvoliť prístup k fázam 1 a 2 opísaným v 1.1.

2.1 Predspracovanie textu

Predspracovanie textu prebieha v troch krokoch:

1. Rozdelenie textu do slov. Text je rozdelený do slov, oddeľovačom je ľubovoľné množstvo bielych znakov.

2. Odstránenie interpunkcie. Zo slov sa odstránia interpunkčné znamienka, ktoré neboli oddelené medzerou (zátvorky, úvodzovky, čiarka, znamienka ukončujúce vetu...).

3. Lematizácia. Pri jazykoch s bohatou flexiou (medzi ktoré patrí aj čeština a slovenčina) je veľmi dôležité uviesť slová do základného tvaru. Inak by rôzne tvary jedného slova boli algoritmom považované za rôzne slová. Základný tvar slova sa tiež nazýva *lema* a uvedenie slova do základného slovného tvaru sa nazýva *lematizácia*.

2.2 Definícia a výber kandidátov

Za kandidátske termy budeme považovať termy zložené z jedného až troch slov (bezprostredne po sebe nasledujúcich v texte), ktoré spĺňajú určité vlastnosti. Tieto

termy budeme nazývať *platné ngramy*. Platný ngram musí byť zložený z *platných slov* a nesmie začínať ani končiť stop-slovom. Za platné slová sa považujú všetky slová, ktorých prvý znak je písmeno abecedy alebo arabská číslica. Stop-slovami rozumieme slová obsiahnuté v negatívnom slovníku alebo čísla.

Negatívny slovník sa skladá z automaticky extrahovanej časti a z používateľskej časti. Automaticky extrahovaná časť obsahuje slová určitých slovných druhov získané z kolekcie dokumentov. (Viac v kapitole 3.) Používateľskú časť prestavuje textový súbor, do ktorého je jednoduché manuálne pridať slová, ktoré chceme zaradiť do negatívneho slovníka.

2.3 Odstraňovanie podvýrazov

Množina automaticky extrahovaných termov často obsahuje jednu alebo viac dvojíc termov, z ktorých je jeden term podvýrazom druhého. (Term a je podvýrazom termu b, keď množina slov, z ktorých sa skladá term a, je podmnožinou množiny slov termu b). Odstránením podvýrazov bolo dosiahnuté zlepšenie úspešnosti.

Nastávajú však prípady, že odstránený term je vhodnejším kandidátom ako nadterm, ktorý je zachovaný. Preto bolo zavedené nasledujúce pravidlo. Podtermy sú odstránené, len ak sa nevyskytujú dostatočne často aj samostatne, teda mimo daného nadtermu. Podmienka je presnejšie vyjadrená takto:

$$\frac{\mathrm{tf}(b)}{\mathrm{tf}(a)} > c$$

Kde a je podvýrazom b a c je konštanta. Hodnota c použitá v implementácii je 0,7. Aplikácia tejto podmienky má pozitívny vplyv na výsledky extrakcie.

Kapitola 3

Kolekcia dát

V tejto práci je využívaná rozsiahla zbierka textov. Táto kapitola sa venuje opisu jej obsahu a spôsobom jej využitia.

3.1 Obsah kolekcie

Kolekcia je výberom dokumentov z českej testovacej kolekcie Ad-Hoc CLEF 2007 (`http://www.clef-campaign.org`). Obsahuje novinové články z českých denníkov *Lidové noviny* a *Mladá fronta Dnes* z roku 2002. Ich celkový počet je 81 735, pre účely experimentov a vyhodnocovania sa používa redukovaný súbor 746 textov. Ku 261 z nich boli autorom práce priradené kľúčové slová. Texty sú morfologicky a syntakticky analyzované a disambiguované. Ku každému slovu sú teda k dispozícii základný slovný tvar, slovný druh, gramatické kategórie a ďalšie informácie.

3.2 Formát súborov

Každý článok je v samostatnom súbore. Formát súborov vychádza z XML, takže všetky informácie sú umiestnené v XML značkách. Samotný text a gramatické informácie sú uzavreté v značkách `title`, `heading` a `text`.

Ilustračný príklad (Text článku bol skrátený na jednu vetu.):

```
<DOC>
<DOCID>MF-20020404296</DOCID>
<DOCNO>MF-20020404296</DOCNO>
<DATE>04/04/02</DATE>
<TITLE>
```

```
1 Obec obec NNFS1-----A----3 Sb
2 se se_^(zvr._zájmeno/částice) P7-X4----------3 AuxT
3 chystá chystat_:T VB-S---3P-AA---0 Pred
4 spustit spustit_:W Vf--------A----3 Obj
5 kotelnu kotelna NNFS4-----A----4 Obj
6 na na-1 RR--4----------5 AuxP
7 biomasu biomasa NNFS4-----A----6 Atr

</TITLE>
<TEXT>
1 Nová nový AAFS1----1A----2 Atr
2 Cerekev Cerekev_;G NNFS1-----A----3 Sb
3 --Z:-------------4 Apos
4 Spuštění spuštění_^(*5stit) NNNS1-----A----18 Sb
5 první první CrFS2----------7 Atr
6 obecní obecní_^(v_obci;_např._úřad) AAFS2----1A----7 Atr
7 kotelny kotelna NNFS2-----A----4 Atr
8 na na-1 RR--6----------7 AuxP
9 Pelhřimovsku Pelhřimovsko_;G NNNS6-----A----8 Atr
10 ,,Z:-------------13 AuxX
11 která který P4FS1----------13 Sb
12 bude být VB-S---3F-AA---13 AuxV
13 využívat využívat_:T_^(*3t) Vf--------A----9 Atr
14 alternativních alternativní AAIP2----1A----15 Atr
15 zdrojů zdroj NNIP2-----A----13 Obj
16 energie energie NNFS2-----A----15 Atr
17 ,,Z:-------------3 AuxX
18 je být VB-S---3P-AA---0 Pred
19 téměř téměř Db-------------20 Adv
20 připraveno připravit_:W VsNS---XX-AP---18 Pnom
21 . . Z:-------------0 AuxK

</TEXT>
</DOC>
```

Vidíme, že obsah horeuvedených značiek je organizovaný do piatich stĺpcov a že každé slovo vo vete či interpunkčné znamienko je reprezentované samostatným riadkom. Význam jednotlivých stĺpcov:

1. Poradie slova vo vete.
2. Samotné slovo z článku.
3. Základný slovný tvar slova.

4. Gramatické kategórie. Prvé písmeno reťazca určuje slovný druh. Reťazec je ukončený číslom. Toto číslo určuje predka v syntaktickom strome vety, teda slovo, ktoré je daným slovom rozvíjané.

5. Vetný člen.

3.3 Využitie

Kolekcia má pre túto prácu veľký význam a využíva sa niekoľkými spôsobmi:

- Získavanie *idf* – inverznej frekvencie výskytu v dokumentoch. (Viď 3.4)
- Získavanie informácií o slovných druhoch. Tie sú ukladané pre jednotlivé lemy.
- Získavanie stop-slov. Zo zbierky môžeme extrahovať slová určitých slovných druhov a použiť ich ako základ negatívneho slovníka. (Viď 3.4)
- Testovanie a vyhodnocovanie úspešnosti. (Viac v kapitole 5.) Pre tento účel sú však využívané len dokumenty, ku ktorým boli manuálne priradené kľúčové slová.

3.4 Extrakcia lingvistických dát

Získavanie *idf*

Pre spočítanie *idf* pre daný term je potrebné vedieť, v koľkých dokumentoch z kolekcie sa daný term vyskytuje. Táto informácia sa dá získať jednoduchým prejdením všetkých dokumentov s tým, že pre každý platný ngram (v zmysle 2.2) si pamätáme, v koľkých dokumentoch sa vyskytoval. Keďže kolekcia obsahuje veľké množstvo platných ngramov, pre úsporu miesta a odstránenie irelevantných ngramov sú uložené dáta len pre ngramy spĺňajúce isté podmienky. Pri prechádzaní dokumentov sú ignorované ngramy, ktoré sa vyskytujú v danom dokumente len jedenkrát. Z dát sú odstránené ngramy, ktoré sa vyskytujú najviac v jednom dokumente.

Negatívny slovník

Dáta z kolekcie tvoria základ negatívneho slovníka. Pri extrakcii stop-slov z kolekcie sa využíva fakt, že všetky slová v dokumentoch majú informáciu o slovnom druhu. Z každého dokumentu sú do negatívneho slovníka pridané slová, ktoré nemajú jeden

z povolených slovných druhov. Povolené slovné druhy sú podstatné mená, prídavné mená, slovesá a príslovky.

Kapitola 4

Prístup *tfidf*

4.1 Opis

Prístup *tfidf* je jedným z najstarších prístupov k extrakcii kľúčových slov. Je založený na dvoch pozorovaniach:

1. Kľúčové termy sa spravidla v texte vyskytujú častejšie.
2. Kľúčovými termami často bývajú odborné výrazy. Ak predpokladáme rozsiahlu mnohotematickú kolekciu dokumentov, odborný výraz z jednej oblasti sa vyskytuje len v malom množstve dokumentov z kolekcie.

Ak term spĺňa prvú podmienku, má vysoké *tf*. Ak spĺňa druhú, má vysokú hodnotu *idf*. Prístup *tfidf* v základnej verzii tieto dve miery kombinuje ako súčin $tf \cdot idf$.

Súčasťou práce je aj implementácia tohto prístupu. Tiež sú preskúmané a porovnané určité modifikácie za účelom zlepšenia úspešnosti. Ukazuje sa, že úspešnosť môže byť zvýšená, ak sú okrem *tf* a *idf* využité aj ďalšie informácie o termoch.

4.2 Algoritmus

Postupujeme podľa všeobecného algoritmu opísaného v kapitole 1. Predpracovanie a výber kandidátov je opísané v kapitole 2.

1. Z predspracovaného textu extrahujeme všetky platné ngramy. Pri ich extrakcii si pre ne počítame *tf*. Ngramom priraďujeme *idf* podľa dát z kolekcie. (Ich získavanie je opísané v sekcii 3.4.)

2. Pre každý ngram vypočítame ohodnotenie, teda $tf \cdot idf$.

3. Termy zoradíme zostupne podľa ohodnotenia.

4. Vrátime prvých n termov. n je parametrom algoritmu.

4.3 Modifikácie základného prístupu

V tejto sekcii budú opísané dve rozšírenia základného prístupu. V oboch ide o zahrnutie ďalšieho rysu termov do hodnotiacej funkcie. Použitými rysmi sú miesto prvého výskytu termu a slovné druhy slov v terme. Kombinácia oboch rozšírení je použitá v konečnej verzii softvérového nástroja.

Použitie miesta prvého výskytu termu

Pre zrozumiteľne napísaný dokument by malo platiť, že čitateľ je schopný posúdiť predmet textu už podľa nadpisu a úvodných pasáží. V týchto častiach textu by sa teda mohla objaviť veľká časť kľúčových slov dokumentu. Na základe tohto predpokladu si zavedieme heuristiku, ktorá pri hodnotení zvýhodňuje termy, ktorých prvý výskyt v texte sa nachádza blízko začiatku textu. Miesto prvého výskytu termu si skrátene označíme *fo* (z anglického *first occurence*). Pre term t si ho definujeme nasledovne:

$$\mathrm{fo}(t) = \frac{\textit{poradie } t \textit{ v texte}}{\textit{počet slov v texte}}$$

Poradím termu v texte rozumieme poradie prvého slova termu.

Do hodnotiacej funkcie môžeme *fo* zakomponovať viacerými spôsobmi. Najjednoduchším je pridať $1 - fo$ ako ďalší činiteľ do súčinu. Dostávame $tf \cdot idf \cdot (1 - fo)$. Označíme si tento prístup ako prístup A.

Ďalší testovaná možnosť je o niečo zložitejšia. Budeme sa na ňu odkazovať ako na prístup B. Hodnotiaca funkcia pri ňom vyzerá takto:

$$tf \cdot idf \cdot \left(c_{fo} \cdot (1 - fo) + \left(1 - \frac{c_{fo}}{2}\right)\right)$$

Tretí činiteľ teda môže nadobúdať hodnoty v intervale $\langle 1 - \frac{c_{fo}}{2}, 1 + \frac{c_{fo}}{2} \rangle$. Pri takto koncipovanej funkcii sú termy s prvým výskytom v prvej polovici textu „odmeňované“ a s prvým výskytom v druhej polovici „trestané“. Konštanta c_{fo} slúži ako váha. Čím je nastavená vyššie, tým viac celkový výsledok závisí od *fo*.

Využitie informácií o slovných druhoch

Ďalším rysom, ktorý je možné využiť pri rozlišovaní kľúčových a ostatných termov je slovný druh. Kľúčovými slovami sú zväčša podstatné mená, často rozvité nejakým prívlastkom. Toto pozorovanie bolo do hodnotiacej funkcie zahrnuté nasledovne: zvýhodňované sú termy, ktorých prvé slovo je podstatné meno alebo prídavné meno. (Prídavné meno stojace osamote ako jednoslovný term zvýhodňované nie je.) Zvýhodňovaným termom je prideľované hodnotenie podľa hodnoty základnej hodnotiacej funkcie, ostatným 30% z tejto hodnoty.

4.4 Úspešnosť

Úspešnosť implementovaných prístupov budeme hodnotiť takzvanou *F-mierou*. Opis metódy vyhodnocovania, ako aj detailné výsledky, sa nachádzajú v kapitole 5.

Tabuľka 4.1 ukazuje, ako jednotlivé modifikácie ovplyvňujú úspešnosť algoritmu. Obidve modifikácie úspešnosť zvyšujú. Ukazuje sa tiež, že prínos využitia jedného rysu je vyšší, keď už bol použitý druhý rys.

	bez s.d.[1]	s s.d.	Δ
bez *fo*	24,35	27,04	2,69
s *fo*[2]	26,72	29,84	3,12
Δ	2,38	2,80	

Tabuľka 4.1: Vplyv využitia rysov na úspešnosť

Boli testované dva spôsoby zapojenia *fo* do hodnotiacej funkcie (opísané vyššie). Rozdiely v ich úspešnosti sú zanedbateľné, ako vidno v tabuľke 4.2. (Pre detailnejšie porovnanie viď tabuľky 5.2 a 5.3.)

V tabuľkách 4.1 a 4.2 sú ako úspešnosť uvádzané najvyššie dosiahnuté F-miery pri extrakcii 7 až 12 termov.

[1] Bez použitia informácie o slovných druhoch

[2] Prístup A

	bez s.d.	s s.d.
prístup A	26,72	29,84
prístup B	26,60	29,83

Tabuľka 4.2: Porovnanie prístupov k zapojeniu *fo*

4.5 Experiment s genetickým algoritmom

Na záver kapitoly si stručne opíšeme experiment, ktorý mal nájsť možné zlepšenie úspešnosti bez pridávania ďalších rysov. Prístup, ktorého úspešnosť sa pokúšame vylepšiť, je založený na prístupe B. Ohodnocovacia funkcia je však zložitejšia:

$$c_{tf} \cdot tf + c_{idf} \cdot idf + c_{fo} \cdot \overline{fo} + c_{tfidf} \cdot tf \cdot idf + c_{tffo} \cdot tf \cdot \overline{fo} + c_{idffo} \cdot idf \cdot \overline{fo} + c_{all} \cdot tf \cdot idf \cdot \overline{fo}$$

kde

$$\overline{fo} = \left(c \cdot (1 - fo) + \left(1 - \frac{c}{2}\right)\right)$$

Nastavenie konštánt sme nechali vyvinúť genetickým algoritmom [9]. Genómom boli jednotlivé konštanty a počet extrahovaných slov. Hodnoty, ktoré môžu gény nadobúdať, boli obmedzené do určitých intervalov. Fitness funkciou bola F-miera. Jedince pre párenie sú vyberané náhodne s pravdepodobnosťou proporcionálnou ich fitness, avšak len z prvých $\frac{2}{3}$ najlepších kandidátov. Pri krížení sa náhodne vyberie deliace miesto a nový jedinec dostane časť génov po deliace miesto od prvého rodiča, časť génov od deliaceho miesta od druhého (*one-point crossover*). S určitou pravdepodobnosťou môže nastať mutácia – výmena hodnoty génu za náhodnú hodnotu. Do nasledujúcej generácie sa zachovávajú dva najlepšie jedince a je do nej tiež pridaný jedinec s úplne náhodným genotypom.

Bolo vykonaných niekoľko experimentov s rôznymi nastaveniami a úvodnými populáciami. Najlepší vyvinutý jedinec dosahoval úspešnosť 30,19, čo je len minimálne zlepšenie v porovnaní s prístupmi A a B. (Podrobné výsledky ukazuje tabuľka 5.4.) Jeho hodnotiaca funkcia vyzerá nasledovne:

$$-0,1 \cdot tf - 0,73 \cdot tf \cdot idf + 0,5 \cdot tf \cdot \overline{fo} + 0,87 \cdot tf \cdot idf \cdot \overline{fo}$$

kde

$$\overline{fo} = \left(1,39 \cdot (1 - fo) + \left(1 - \frac{1,39}{2}\right)\right)$$

Kapitola 5

Vyhodnocovanie úspešnosti

Vyhodnotenie úspešnosti algoritmu na extrakciu kľúčových slov je netriviálna otázka. Neexistuje spôsob, ako objektívne posúdiť, či daný term je pre daný text kľúčový alebo nie. Hulthová[2] uvádza niekoľko metód, ktoré je možné zvoliť pre vyhodnocovanie. Niektoré vyžadujú ľudských používateľov, aby hodnotili kvalitu ponúkaných termov. Napríklad prístup navrhovaným Turneym [7]. Pri tomto prístupe sú používateľovi navrhované kľúčové slová pre webové stránky. Používateľ následne hodnotí navrhované termy ako „dobré“ či „zlé“. Metódy vyžadujúce externých používateľov pre túto prácu nie sú vhodné, keďže autor nemá k dispozícii zdroje na ich prevedenie. Preto bola použitá metóda založená na manuálne priradených kľúčových slovách, ktorú zvolila aj Hulthová v [2]. Táto nevyžaduje externých používateľov a umožňuje rýchle a opakované vyhodnocovanie.

5.1 Zvolená metóda

Pri tejto metóde máme k dokumentom priradené kľúčové slová ľudským indexérom. Tieto sa považujú za jediné správne. Úspešnosť algoritmu je meraná tzv. *F-mierou*. F-miera (angl. *F-measure*) sa často používa v oblasti získavania informácii (information retrieval) [8]. Jej hodnota je závislá na dvoch veličinách – *presnosti (precision)* a *úplnosti (recall)*. Tie sú definované nasledovne: Nech A je množina automaticky extrahovaných termov a M množina manuálne priradených termov k danému dokumentu. Potom:

$$presnosť = \frac{|A \cap M|}{|A|};\ úplnosť = \frac{|A \cap M|}{|M|}$$

Presnosť teda vyjadruje, koľko z extrahovaných slov bolo správnych a úplnosť vyjadruje, koľko z manuálnych kľúčových slov sa podarilo automaticky extrahovať.

F-miera je definovaná nasledovne:

$$F\text{-}miera = \frac{(1+\beta^2)\cdot(presnosť \cdot úplnosť)}{\beta^2 \cdot presnosť + úplnosť}$$

Koeficient β je možné využiť na priradenie väčšej váhy (dôležitosti) jednej z veličín. Pokiaľ je $\beta > 1$, väčšiu váhu má presnosť. Ak je $\beta < 1$, väčšiu dôležitosť má úplnosť. V tejto práci budeme prikladať obom veličinám rovnaký význam, β bude mať teda hodnotu 1. Dostávame:

$$F\text{-}miera = \frac{2\cdot(presnosť \cdot úplnosť)}{presnosť + úplnosť}$$

Pre účely vyhodnocovania bolo k dispozícii 261 textov s manuálne priradenými k. slovami. Priemerný počet slov pre jeden text bol 6,08.

5.2 Poznámky k metóde

Vhodnou podmienkou pre použitie tejto metódy je, aby manuálne kľúčové slová boli vyberané profesionálnym indexérom. To zaručuje istú konzistenciu a kvalitu vybraných kľúčových slov. Táto podmienka v tejto práci splnená nie je. Použitá kolekcia neobsahuje manuálne kľúčové slová, takže boli priradené autorom práce. To môže negatívne ovplyvňovať kvalitu vyhodnocovania.

Zvolený prístup vyhodnocovania je k algoritmu „prísny“ v tom zmysle, že za jediné správne považuje manuálne priradené termy. Tým nepostihuje situáciu, že niektoré z algoritmom navrhovaných termov by väčšina používateľov považovala za akceptovateľné. Turneyho prístup (opísaný vyššie) túto vlastnosť nemá, keďže využíva hodnotenie používateľmi. Výsledky ukazujú, že nameraná akceptovateľnosť pri jeho metóde je vyššia ako presnosť pri prístupe založenom na manuálnych kľúčových slovách [7].

5.3 Výsledky

Nasledujú výsledky vyhodnocovania implementovaných prístupov. Parameter n znamená počet extrahovaných slov. Výsledky sú uvádzané ako percentá, teda ako stonásobky nameraných hodnôt. Sú zaokrúhlené na dve desatinné miesta.

	bez slov. druhov			so slov.druhmi		
n	presnosť	úplnosť	**F-miera**	presnosť	úplnosť	**F-miera**
7	23,94	24,51	23,44	27,54	28,27	**27,04**
8	23,39	27,11	**24,35**	25,75	30,24	27,02
9	21,39	28,60	24,13	24,20	31,83	26,76
10	20,40	29,45	23,44	22,50	32,87	26,04
11	19,57	31,11	23,40	21,54	34,60	25,91
12	18,71	32,25	23,01	20,59	35,87	25,52

Tabuľka 5.1: Prístup *tfidf*, základná verzia

	bez slov. druhov			so slov.druhmi		
n	presnosť	úplnosť	**F-miera**	presnosť	úplnosť	**F-miera**
7	27,18	27,53	26,47	30,18	30,36	29,28
8	25,14	28,93	26,12	28,71	33,13	**29,84**
9	24,40	31,50	**26,72**	26,75	34,85	29,41
10	23,15	32,94	26,44	25,69	37,06	29,51
11	22,23	34,70	26,40	24,41	38,49	29,10
12	21,20	36,22	26,09	23,07	39,56	28,46

Tabuľka 5.2: Prístup *tfidf* s použitím *fo* (A)

	bez slov. druhov			so slov.druhmi		
n	presnosť	úplnosť	**F-miera**	presnosť	úplnosť	**F-miera**
7	27,06	27,46	26,38	30,02	30,30	29,18
8	25,36	29,13	26,30	28,62	33,17	**29,83**
9	24,27	31,37	**26,60**	26,71	34,90	29,43
10	23,02	32,95	26,38	25,61	37,16	29,48
11	22,17	34,91	26,42	24,59	39,04	29,41
12	21,16	36,21	26,03	23,27	40,20	28,75

Tabuľka 5.3: Prístup *tfidf* s použitím *fo* (B)

	bez slov. druhov			so slov.druhmi		
n	presnosť	úplnosť	**F-miera**	presnosť	úplnosť	**F-miera**
7	27,12	26,81	26,06	30,17	30,00	29,08
8	25,70	29,09	26,48	29,18	33,37	**30,19**
9	24,39	30,99	26,51	27,21	35,13	29,83
10	23,32	33,11	**26,63**	25,97	37,06	29,78
11	22,22	34,63	26,42	24,45	38,24	29,11
12	21,18	35,90	26,03	23,56	39,85	28,95

Tabuľka 5.4: Prístup vyvinutý genetickým algoritmom

Kapitola 6

Záver

V tejto kapitole si stručne zhrnieme výsledky práce a načrtneme možné zlepšenia do budúcnosti.

6.1 Zhrnutie

V tejto práci sme sa venovali problematike extrakcie kľúčových slov z textu. Skúmali sme vplyv využitia ďalších rysov v prístupe *tfidf*. Merania úspešnosti ukázali, že zapojenie miesta prvého výskytu termu a slovných druhov vedie k zvýšeniu úspešnosti extrakcie. Prístup využívajúci oba spomenuté rysy je využitý pri implementácii nástroja na automatickú extrakciu. K nástroju bolo vytvorené webové užívateľské rozhranie. To ako ďalšiu funkciu obsahuje manuálne priraďovanie kľúčových slov k dokumentom. Túto funkciu využil aj autor práce pre dodanie kľúčových slov k niektorým dokumentom z kolekcie za účelom merania úspešnosti.

Dosiahnutá úspešnosť vyjadrená F-mierou bola takmer 0,3. Vzhľadom k náročnosti problému to nie je nízke číslo, avšak priestor na zlepšovanie je veľký. Hulthovej najkvalitnejší algoritmus v [2] dosahuje až 0,45.

6.2 Možné zlepšenia

Výraznejšie zlepšenie kvality extrakcie by mohlo byť dosiahnuté skúmaním prístupov založených na strojovom učení a hľadaním ďalších rysov odlišujúcich kľúčové slová od ostatných.

U softvérového nástroja je niekoľko možností, ako zlepšiť jeho použiteľnosť. Zmeniť by sa mohol spôsob prezentácie výsledkov. Tie sú používateľovi predkladané celé

v malých písmenách a jednotlivé slová termu sú lematizované samostatne. Takže napr. prídavné mená sú vždy v mužskom rode a nie v zhode s rozvíjaným podstatným menom. Softvér tiež nedostatočne podporuje iné kódovania ako UTF-8.

Dodatok A

Konzolové rozhranie

V rámci práce bol vytvorený softvér na extrakciu kľúčových slov z textu. Táto kapitola sa venuje jeho ovládaniu z príkazového riadku, jeho možnostiam a obmedzeniam.

A.1 Požiadavky pre beh programu

Pre spustenie programu je nutné mať nainštalovaný interpret jazyka Python[1] vo verzii 2.5 alebo 2.6. Jeho inštalačný súbor pre operačný systém Windows je súčasťou priloženého DVD. Používatelia operačného systému GNU/Linux si môžu Python nainštalovať pomocou balíčkovacieho systému svojej distribúcie, alebo si ho môžu skompilovať zo zdrojového balíčka, ktorý sa nachádza na DVD.

A.2 Inštalácia

Inštalácia nástroja je jednoduchá. Predpokladajme, že Python už máme nainštalovaný. Súčasťou DVD je inštalačný archív `kwex.zip`. Ten stačí rozbaliť do ľubovoľného adresára a môžeme začať program používať.

A.3 Používanie nástroja

Syntax použitia programu je nasledovná:

```
python kwex.py [voľby] [súbor1, súbor2 ...]
```

[1] Oficiálna stránka Pythonu: `http://python.org/`

Položky v hranatých zátvorkách sú nepovinné. Po prípadných voľbách môže nasledovať ľubovoľný počet názvov súborov. Zadané súbory sú použité ako vstup pre extrakciu kľúčových slov. Ak nie je zadaný ani jeden názov súboru, vstup sa číta zo štandardného vstupu.

Prehľad volieb:

`-n` ⟨`počet slov`⟩ Určuje počet slov na výstupe.

`-v` Zobrazí dodatočné informácie o termoch (Hodnoty niektorých rysov). V dlhej forme `--verbose`.

`-l` ⟨`meno súboru`⟩ Ako vstup budú použité súbory vymenované v zozname dodanom ako parameter. Dlhá form tejto voľby je `--file-list`.

Ak je použitá voľba `-l` a zároveň sú za voľbami uvedené názvy súborov, tieto sú spracované rovnako ako súbory zo zoznamu.

Výstupom sú kľúčové slová pre jednotlivé súbory. Ak je spracovávaný viac ako jeden súbor, zoznam kľúčových termov pre každý súbor je uvedený textom obsahujúcim názov súboru. Výstup je vypisovaný do štandardného výstupu.

Príklad použitia:

```
python kwex.py -n 4 monty.txt python.txt
```

Výstup:

```
Keywords for monty.txt:
monty python
spam
holy grail
michael palin
Keywords for python.txt:
silly walk
moutaineer
defense against fruit
life of brian
```

Dodatok B

Webové rozhranie

Ku konzolovému nástroju bolo vytvorené aj webové rozhranie. Okrem extrakcie kľúčových slov má webové rozhranie ešte jednu funkciu. Umožňuje manuálne priraďovanie kľúčových slov k dokumentom. Táto kapitola obsahuje stručný návod na používanie aplikácie.

B.1 Extrakcia kľúčových slov

Získanie kľúčových slov vo webovom rozhraní prebieha nasledovne:

1. Ak ešte nie je zvolená, prejdeme do záložky „Find keywords“.
2. Do oblasti pre vkladanie textu napíšeme text alebo ho vložíme zo schránky operačného systému.
3. Napravo od textu sa nachádzajú nastavenia. Zmenou hodnoty v políčku „Number“ možeme nastaviť želaný počet slov na výstupe. Ak zaškrtneme políčko „Show details“, okrem slov samotných budú na výstupe aj ďalšie informácie o nich.
4. Stlačíme tlačítko „Find keywords“ a na obrazovke sa objavia výsledky. Ukážka obrazovky s výsledkami je na obrázku B.1.

B.2 Manuálne priraďovanie kľúčových slov

Postup pri priraďovaní kľúčových slov:

1. Prejdeme do záložky „Tag texts“ (ak ešte nie je zvolená).

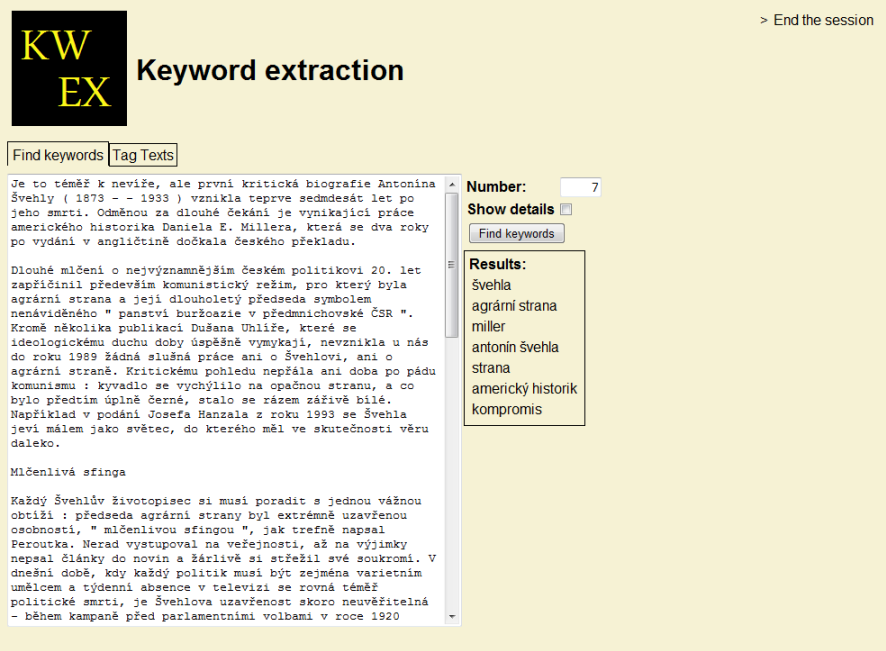
2. Z padacieho menu v ľavej hornej časti obrazovky vyberieme dokument, ku ktorému chceme slová priraďovať. Text dokumentu sa objaví na obrazovke. (Za podmienky, že je v prehliadači povolený Javascript. V opačnom prípade je treba použiť tlačítko „Select“.) Dokumenty je možné prechádzať aj tlačítkami označenými šípkami doľava a doprava. Tými je možné prejsť na predchádzajúci, resp. nasledujúci dokument.

3. Kľúčové slová pre daný text môžeme vkladať dvoma spôsobmi. Môžeme ich vpísať do textového poľa napravo od textu, oddelené čiarkami. Ďalšou metódou je označiť v texte kľúčový termín pomocou myši a použiť tlačítko „Add selected text“.[1]

4. Pre trvalé uloženie vybraných slov použijeme tlačítko „Store keywords“.

Obrázok B.2 ukazuje vzhľad rozhrania na manuálne priraďovanie kľúčových termínov.

[1]Táto metóda funguje v aktuálnych verziách prehliadačov Internet Eplorer, Mozilla Firefox a Opera. Fungovanie v ostatných prehliadačoch nie je zaručené.



Obr. B.1: Výsledok extrakcie kľúčových slov vo webovom rozhraní

Obr. B.2: Manuálne priraďovanie kľúčových slov vo webovom rozhraní

Dodatok C

Implementácia nástroja a experimentov

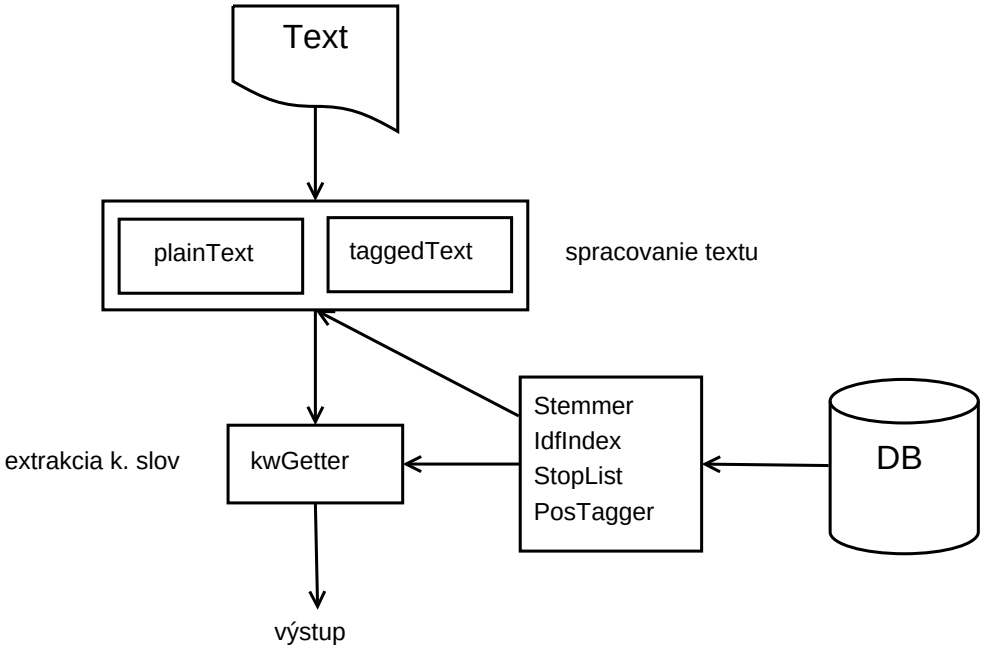
C.1 Zvolená platforma

Ako jazyk implementácie bol zvolený Python. Jeho výhodami sú multiplatformovosť, bohatá štandardná knižnica a rýchly vývoj. Za nevýhodu môže byť považovaný nižší výkon oproti kompilovaným jazykom. Avšak táto nevýhoda sa pri práci výrazne neprejavila.

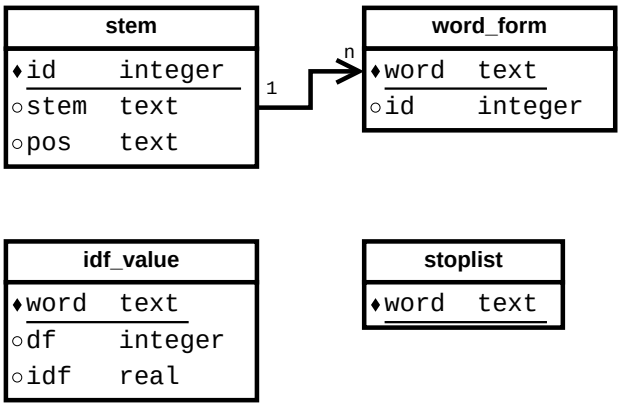
Dáta využívané aplikáciou sú uložené v databáze SQLite. Databáza SQLite ukladá dáta do súboru, ktorý je jednoducho prenositeľný. Pre jej používanie sa nevyžaduje inštalácia žiadneho servera, čo zjednodušuje inštaláciu a údržbu aplikácie.

C.2 Návrh

Na obrázku C.1 je znázornený priebeh extrakcie kľúčových slov. Z textu sú najprv extrahované všetky platné ngramy a dáta o nich. Pre tento účel slúžia moduly `plainText` (pre jednoduchý text) a `taggedText` (pre súbory z kolekcie, ich formát je opísaný v sekcii 3.2). Pre získanie informácií o termoch z databázy slúžia triedy `Stemmer`, `IdfIndex`, `PosTagger` a `StopList`. Ngramy a dáta o nich sú vstupom pre ohodnocovanie kľúčovosti. To je vykonávané nejakým potomkom triedy `BaseKeywordGetter`. V konečnej implementácii je použitá trieda `TfIdfKeywordGetter`. Pri návrhu bolo dbané, aby pridanie inej implementácie ohodnocovania (a možnosti používateľa vyberať medzi viacerými prístupmi) vyžadovalo len málo úsilia.



Obr. C.1: Náčrt architektúry nástroja



Obr. C.2: Štruktúra databázy

C.3 Spracovanie textu

Ako už bolo spomenuté, predpracovanie textu a získanie platných ngramov sa deje v moduloch `plainText` resp. `taggedText`. Oba obsahujú funkcie `getNgramsAndData` a `getValidNgrams`. Prvá z nich vracia pythonovský slovník (dátový typ `dict`, ďalej len slovník), ktorého kľúčmi sú termy ako reťazce a hodnotami sú inštancie triedy `NgramData`. Tie obsahujú informácie o terme, ako *tf*, *idf* a miesto prvého výskytu v texte. Funkcia `getValidNgrams` vracia rovnaký druh slovníka, avšak jedinou informáciou o termoch je frekvencia ich výskytu. Táto funkcia sa používa napríklad pri napĺňaní databázy o *idf*. Modul `taggedText` obsahuje okrem dvoch spomínaných funkcií ešte pomocné funkcie pre prácu s označkovanými textami z kolekcie.

C.4 Prístup k dátam

Niektoré rysy termov sa dajú získať priamo z textu, pre iné je potrebné pristúpiť k dátam uloženým v databáze. Štruktúra databázy je jednoduchá. Je znázornená na obrázku C.2.

Ako lematizátor slúži trieda `Stemmer`. Prístup k informáciám o slovných druhoch slov umožňuje trieda `PosTagger`. Dáta o *idf* sa získavajú pomocou triedy `IdfIndex`. Negatívny slovník reprezentuje trieda `StopList`.

Trieda `Stemmer`

Trieda `Stemmer` slúži ako lematizátor. Na ukladanie dát využíva tabuľky `stem` a `word_form`. Na získanie základného tvaru slova slúži metóda `getStem`. Tá vykoná `select` na dvoch spomenutých tabuľkách spojených cez stĺpec `id`. Pokiaľ sa hľadaný tvar v databáze nenachádza, metóda vráti pôvodný tvar. Pokiaľ bol v konštruktore objektu nastavený parameter `isCached` na `true`, výsledky hľadania sa kešujú. Keš je implementovaný ako slovník, ktorého kľúčmi sú slovné tvary a hodnotami sú tvary základné. Získavanie dát pre lematizátor z kolekcie je opísané v sekcii 3.4. Spolu s lemami sa z kolekcie získavajú aj slovné druhy slov a ukladajú sa do stĺpca `pos` v tabuľke `stem`.

Trieda `PosTagger`

Na získavanie slovných druhov slúži trieda `PosTagger`. Slovné druhy sú uložené pre základné tvary slov v stĺpci `pos` tabuľky `stem`. Slovný druh pre (už lematizované) slovo získame volaním metódy `getPos`. Návratová hodnota je jeden znak kódujúci slovný druh. Kódovanie je rovnaké ako v označkovaných súboroch z kolekcie. Ku každému slovnému druhu existuje príslušná položka triedy `Pos` v module `language`. Hodnotou danej položky je znak príslušiaci danému slovnému druhu. Napríklad pre podstatné meno je v triede `Pos` položka `NOUN` s hodnotou „N“. Ak sa v databáze nenachádza slovný druh pre dané slovo, je vrátená hodnota `PoS.UNDEF` („X“). Výsledky môžu byť kešované analogicky ku triede `Stemmer`.

Trieda `IdfIndex`

Účelom triedy `IdfIndex` je umožniť prístup k dátam o *idf* termov. Tie sú uložené v tabuľke `idf_value`. *Idf* pre daný term vracia metóda `getIdf`. Ak term v tabuľke nie je prítomný, je vrátená východzia hodnota z modulu `config` (`config.DEF_IDF`). Extrakcia dát o *idf* z kolekcie je opísaná v sekcii 3.4.

Trieda `StopList`

Trieda `StopList` reprezentuje negatívny slovník. Ten má uložený ako množinu (`set`) v položke `__stopList`. Slovník sa napĺňa z dvoch zdrojov. Z databázy a zo súboru. V databáze sú v tabuľke `stoplist` uložené automaticky extrahované slová (viď sekciu 3.4). V súbore (cesta k nemu je uložená v premennej `config.STOPLIST_FILE`) sú uložené dodatočné slová editovateľné používateľom. Na doplnenie dát z databázy

slúži metóda `updateFromDb` a na doplenie zo súboru metóda `updateFromFile`. Či sa dané slovo nachádza v negatívnom slovníku, zistíme metódou `isStopWord`.

Inštancie horeuvedených tried sa získavajú pomocou triedy `DBKwexFactory`. Napríklad lematizátor dostaneme volaním

```
DBKwexFactory.getInstance().getStemmer().
```

C.5 Extrakcia kľúčových slov

Ohodnotenie kľúčovosti termov a vrátenie najlepších kandidátov majú za úlohu čiastočne abstraktná trieda `BaseKeywordGetter` a jej potomkovia. Kľúčové slová zo zadaných kandidátov vyberá metóda `getKeyWords`. Tá je implementovaná už v triede `BaseKeywordGetter`, avšak využíva metódu `evaluateTerms`, ktorú musia implementovať potomkovia. Základná trieda umožňuje vybrať n najlepších kandidátov alebo kandidátov s ohodnocovacou funkciou nad určitým prahom podľa parametrov konštruktora. Prístup *tfidf* s modifikáciami (opísaný v kapitole 4) implementuje trieda `TfIdfKeywordGetter`.

C.6 Experimenty

Pre porovnávanie úspešnosti rôznych prístupov metódou opísanou v kapitole 5 bola vytvorená trieda `Evaluator`. Do jej inštancie môžeme pridať objekty[1], ktoré chceme hodnotiť, pomocou metódy `addKwGetter`. Metóda `run` spustí vyhodnocovanie. Prechádza dokumenty, ktoré majú manuálne priradené kľúčové slová a porovnáva ich s automaticky extrahovanými. Pre každý objekt počíta priemernú presnosť, úplnosť a F-mieru. Výsledky vyhodnocovania vypíše metóda `report`.

[1] potomkov triedy `BaseKeywordGetter` alebo ľubovoľné objekty s metódami `getKeyWords` a `infoStr`

Dodatok D

Implementácia webového rozhrania

D.1 Zvolený programovací jazyk

Webové rozhranie je implementované v jazyku PHP vo verzii 5. Tento jazyk patrí medzi najrozšírenejšie programovacie jazyky pre vývoj webových aplikácií. Obsahuje množstvo štandardných funkcií pre tieto účely.

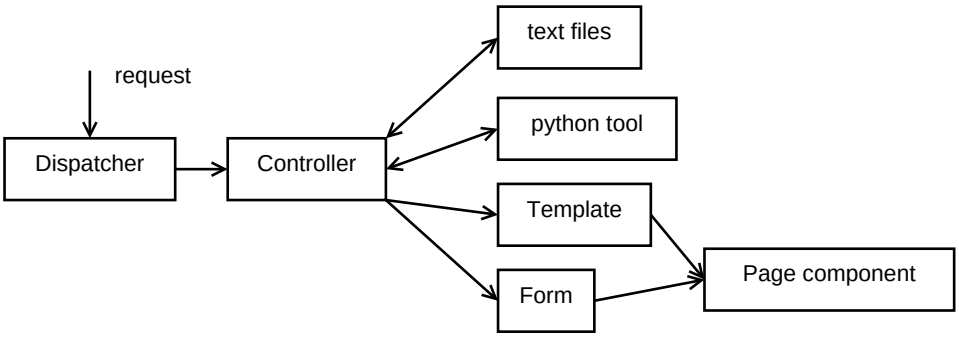
D.2 Návrh

Obrázok D.1 znázorňuje priebeh spracovania požiadavky od používateľa a jednotlivé zložky softvéru. Spracovanie požiadavky prebieha nasledovne:

1. `Dispatcher` určí `Controller` pre danú úlohu a ten prevezme kontrolu.
2. `Controller` spracuje požiadavku. Môže využívať tieto zdroje dát: dáta od užívateľa (GET a POST dáta), textové súbory a konzolový nástroj.
3. `Controller` vyplní výstupné dáta do príslušnej šablóny (`Template`) a tú nechá vypísať na výstup.

D.3 Controller

Triedy implementujúce rozhranie `Controller` majú za úlohu vykonávať logiku aplikácie. Rozhranie má len jednu metódu, a to `run`. V aplikácii sa používajú tri



Obr. D.1: Náčrt architektúry webového rozhrania a toku dát

triedy implementujúce rozhranie `Controller`. `IndexController` implementuje logiku pre extrakciu kľúčových slov. `TaggingController` je pre manuálne priraďovanie kľúčových slov. `AuthController` kontroluje zadávanie hesla na začiatku práce s aplikáciou.

D.4 Zobrazovanie stránok

Na definíciu vzhľadu a rozmiestnenia jednotlivých prvkov stránok slúžia triedy z balíčka `view/template`. Nim je možné metódou `setParams` nastavovať hodnoty niektorých prvkov. Trieda `MainTmpl` predstavuje šablónu pre obrazovku s extrakciou kľúčových slov. `TaggingTmpl` pre manuálne priraďovanie a `AuthTmpl` pre úvodnú obrazovku so zadávaním hesla.

Šablóny môžu pre vykresľovanie nietorých prvkov stránky použiť triedy z balíčka `view/pageComponent`. Nazvime ich komponenty. Tie reprezentujú rôzne prvky stránky, ako napríklad menu pre výber funkcie alebo ovládacie prvky formulára. Niektoré komponenty sú len reprezentáciou HTML značiek. Tie sa dajú použiť na tvorbu zložitejších komponentov.

D.5 Formuláre

HTML formuláre sú reprezentované triedou `Form` a jej potomkami. Táto trieda má položku `data`, v ktorej je asociatívne pole. Kľúčmi sú názvy polí formulára a hodnotami ich hodnoty. Na prácu s poliami formulára slúžia metódy `getField`, `setField` a `addField`. Na načítanie hodnôt odoslaných používateľom sa používa

metóda `setDataFromRequest`. Tá je užitočná pri opätovnom zobrazení formulára s hodnotami zadanými používateľom. Táto metóda nie je implementovaná v základnej triede a je nutné ju implementovať „ručne“ v potomkoch. V aplikácii sa používajú traja potomkovia triedy `Form`: `MainForm`, `TagForm` a `AuthForm`. Prvý sa používa pri automatickej extrakcii, druhý pri manuálnom priraďovaní a tretí pri zadávaní hesla.

D.6 Volanie pythonového jadra

Na volanie pythonových skriptov s parametrami je určená trieda `PyRunner`. Meno skriptu sa nastavuje buď v konštruktore alebo metódou `setScript`. Paramtre môžeme nastaviť v konštruktore, metódou `setParams` alebo priamo v metóde `run`. Táto metóda spúšťa skript a vracia text, ktorý skript vypísal do štandardného výstupu. Text je vrátený ako pole riadkov.

Triedu `PyRunner` využíva trieda `KwGetter`, ktorá má jednu metódu, a to `getKeywords`. Tá volá pythonové jadro a pre zadaný text a parametre vracia kľúčové slová .

D.7 Ukladanie manuálnych kľúčových slov

Manuálne priradené kľúčové slová sú ukladané do textových súborov. Názov textového súboru s k. slovami sa skladá z plného názvu pôvodného súboru, ku ktorému je pridaná koncovka „.kw“. Uloženie manuálnych k. slov vykonáva metóda `storeWords` triedy `TaggingController`.

Dodatok E

Obsah priloženého DVD

- Inštalačný balíček konzolového nástroja sa nachádza v adresári `kwex`.
- Zdrojové kódy k softvéru vytvorenému v rámci bakalárskej práce obsahuje adresár `src`.
- Samotný text bakalárskej práce nájdeme v adresári `thesis`.
- Adresár `images` obsahuje obrázky z používania webového softvéru.
- V adresári `python` sa nachádzajú inštalačné balíčky Pythonu pre Windows a GNU/Linux.

Literatúra

[1] Earl, L.L.: *Experiments in automatic extracting and indexing*, Information storage & retrieval, **6** (1970) 313–334.

[2] Hulthová, A.: *Automatic Keyword Extraction: Combining Machine Learning and Natural Language Processing*, dizertačná práca, Štokholmská univerzita (2004).

[3] Kirschner Z.: *MOSAIC - A Method of Automatic Extraction of Significant Terms from Texts* Praha, MFF UK (1983)

[4] Luhn, H.P.: *A statistical approach to mechanized encoding and searching of literary information*, IBM Journal of Research and Development, **1** (1957) 309–317.

[5] Luhn, H.P.: *Auto-encoding of documents for information retrieval systems*, Modern trends in documentation, Pergamon Press, Londýn (1959) 45–58.

[6] Sparck Jones, K.: *A statistical interpretation of term specificity and its application in retrieval*, Journal od Documentation, **28** (1972) 11–21.

[7] Turney, P.D.: *Learning algorithms for keyphrase extraction*, Iformation retrieval, **2** (2000) 303–336.

[8] Wikipedia: *F-score*, článok na serveri Wikipedia.
`http://en.wikipedia.org/wiki/F-score`

[9] Wikipedia: *Genetic algorithm*, článok na serveri Wikipedia.
`http://en.wikipedia.org/wiki/Genetic_algorithm`