MASARYKOVA UNIVERZITA
FAKULTA INFORMATIKY

Automatické určení domény a klíčových slov stránky

DIPLOMOVÁ PRÁCE

Jiří Materna

Brno, jaro 2008

Prohlášení

Prohlašuji, že tato diplomová práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Vedoucí práce: doc. RNDr. Lubomír Popelínský, Ph.D.

Poděkování

Děkuji vedoucímu své diplomové práce doc. RNDr. Lubomíru Popelínskému, Ph.D. za odborné vedení, konzultantům Mgr. Pavlu Rychlému, Ph.D a Ing. Štěpánu Škrobovi za cenné rady, Elišce Mikmekové za pomoc při tvorbě grafů a v neposlední řadě celé své rodině za podporu během všech pěti let studia.

Shrnutí

Cílem této práce je navrhnout a otestovat přístup, který umožní automatickou klasifikaci webových stránek do domén a určení klíčových slov stránky. Klasifikace stránek je založena na použití strojového učení. Hlavním problémem je však malý rozsah webových stránek, který užití metod strojového učení znesnadňuje. V práci jsou navrženy dva přístupy, které se snaží tento nedostatek minimalizovat. Prvním z nich je zohledňování širšího kontextu webové stránky, to znamená, že se analyzují i stránky, umístěné ve stejné internetové doméně, které jsou ze zkoumané stránky odkazovány. Druhou metodou je shlukování termů dokumentu na základě jejich podobného gramatického kontextu. Pro tyto účely je vytvořen poměrně rozsáhlý thesaurus a z něho shlukový slovník.

Určení klíčových slov je navrženo jako navazující úloha po klasifikaci, která užívá informací o dané doméně. Při omezení na užší doménu je již možné určit klíčová slova na základě výskytů slov v dokumentu.

Výsledky testování ukázaly, že použité přístupy dokázaly výrazně zvýšit přesnost klasifikace.

Klíčová slova

Klasifikace, klíčová slova, strojové učení, klasifikace webových stránek, thesaurus.

Obsah

Kapitola 1

Úvod

1.1 Motivace

V současné době jsou na webu veřejně přístupné miliardy stránek a jejich počet každým dnem rapidně narůstá. Při takovém množství dostupných informací není dobře možné procházet jednotlivé stránky postupně, ale je třeba uživateli poskytnout nástroje, pomocí nichž lze vybrat pouze ty relevantní, v závislosti na uživatelově dotazu.

Dnes se můžeme nejčastěji setkat se dvěma přístupy k získávání informací na webu – fulltextové vyhledávače (např. Google [1], Seznam.cz [2] nebo Jyxo [3]) a katalogy (např. Dmoz [4] nebo Seznam.cz [5]). Fulltextové vyhledávače sice nabízejí celou řadu možností jak dotaz přesně specifikovat, ale k sémantické analýze stránky mají ještě daleko. Naproti tomu katalogy třídí webové stránky do hierarchií skupin právě podle jejich obsahu. Při vytváření velkých katalogů se však naráží na problém obrovského počtu webových stránek, jež jsou do jednotlivých kategorií zařazovány zatím většinou ručně nebo strojově s asistencí člověka. Dalším příkladem potřeby klasifikace webových stránek do kategorií jsou velké korpusy vytvářené z webu. Všestranně použitelné korpusy by měly obsahovat kromě samotných textů také tzv. *metadata*, tedy informace o žánru, typu dat, zdroji apod. V případě, že chceme vytvářet korpusy o rozsahu miliard pozic, je ruční doplňování takovýchto informací neúnosné.

1.2 Existující přístupy ke klasifikaci

Problémem automatické klasifikace anglicky psaných stránek do kategorií se již zabývala řada prací, jmenujme např. [1, 9, 24, 30, 31, 37]. Existující

1. http://www.google.com
2. http://search.seznam.cz/
3. http://www.jyxo.cz
4. http://www.dmoz.org
5. http://www.seznam.cz

metody můžeme podle použitých přístupů rozdělit do tří základních skupin [36]:

- **Metody založené na analýze META tagů** – tyto techniky berou v úvahu pouze META tagy jako `<META name="keywords">` nebo `<META name="descriptions">`. Hlavními nevýhodami takového přístupu je, že META tagy nejsou povinnou součástí všech HTML stránek, tudíž by některé stránky nemohly být vůbec klasifikovány, a především fakt, že autoři stránek někdy úmyslně uvádějí zavádějící informace, aby se jejich stránky dostaly na přední místa vyhledávačů. Z těchto důvodů se v současné době ve většině klasifikátorů hodnoty META tagů vůbec neberou v úvahu.

- **Metody vycházející ze strukturní podobnosti** – v těchto technikách se předpokládá, že stránky patřící do stejné kategorie mají podobnou strukturu [1]. Strukturou dokumentu se rozumí například rozmístění odkazů, existence, velikost či barevnost obrázků, rozsah textu a jeho formátování apod. Při klasifikaci se nebere v úvahu obsah textu, ale pouze vzhledová podobnost s nějakým modelem. Tyto metody mohou být pro některé případy relativně úspěšné, ale většinou je třeba je zkombinovat i s textově závislými přístupy.

- **Metody zkoumající obsah textu** – tyto metody jsou nejúspěšnější a také v praxi nejčastěji používané. Hlavní myšlenkou je využití podobností vektorových modelů textů jednotlivých stránek. Složky vektoru představují slova v dokumetu a zachycují jejich četnost, často různými způsoby normalizovanou. Tímto způsobem vektorizace je sice ztracena informace o pořadí slov v dokumentu, na druhou stranu je však úloha značně zjednodušena.

  Pro každou třídu je pak z takovýchto vektorů vytvořena množina trénovacích příkladů, na kterých je naučen klasifikátor (Naivní Bayesovský klasifikátor, Support Vector Machines, C4.5 apod.)

  Je zřejmé, že úspěch těchto metod silně závisí na kvalitě, počtu a rozsahu trénovacích příkladů, přičemž je však kvalita a rozsah jednotlivých stránek na webu velice variabilní. Bylo zjištěno, že 94,65 % webových stránek obsahuje méně než 500 různých slov [46]. Výzkum také ukázal, že průměrná četnost výskytu jednoho slova v jednom dokumentu je menší než 2,0. Proto pro klasifikaci webových stránek s dostatečnou přesností nemohou být použity klasické metody bez vhodného předzpracování, jinak dobře použitelné při klasifikaci delších textů.

1.3 Cíle a struktura práce

Cílem této diplomové práce je navrhnout a otestovat metodu, která umožní pro libovolnou česky psanou webovou stránku automaticky rozhodnout příslušnost k jedné, případně několika kategoriím a nalézt její klíčová slova.

Vzhledem k výsledkům předchozích experimentům pro angličtinu byl pro náš systém zvolen přístup, který kombinuje metody založené na analýze textů s metodami zkoumajícími strukturu dokumentu.

Bylo prokázáno, že při dostatečně velkém množství trénovacích příkladů, s užitím metod strojového učení vhodných pro práci s velkým počtem atributů, nemá zvolená metoda na přesnost příliš velký vliv [48], proto se práce soustředí především na metody předzpracování a shlukování, které se zdají být pro úspěch klasifikátoru klíčové. Problém malého rozsahu textů v HTML stránkách je v práci řešen dvěma novými přístupy:

- Experimenty bylo prokázáno, že linky odkazující z webové stránky v drtivé většině případů směřují na stránku podobného tematického zaměření. Proto není pro účely strojového učení považována za jeden příklad pouze úvodní stránka, ale jsou k ní přidruženy i všechny stránky z ní odkazované, které se však nacházejí ve stejné internetové doméně.

- Přestože předchozí rozšíření často výrazně zvýší počet slov v dokumentu, stále ještě nemusí být dostačující a je třeba přejít ke shlukování výrazů. Kromě klasického přístupu, který ztotožňuje slova se stejným lemmatem, je v práci představena metoda, jež ztotožňuje výrazy, často se vyskytující ve stejném gramatickém kontextu.

Důležitou otázkou při tvorbě klasifikátorů je výběr vhodných tříd, pojednává o ní například publikace [43]. V práci byly kategorie převzaty z katalogu Seznam.cz. Tento český portál nabízí dva katalogy – Firmy [6] a Odkazy [7]. Katalog Firmy obsahuje pouze stránky, představující webové prezentace firem, zatímco Odkazy zahrnují všechny typy webových stránek. Pro naše účely byl vybrán katalog Odkazy, neboť lépe vystihuje rozmanitost českého webu a výsledný klasifikátor potom může být použitelný univerzálněji.

Následující kapitola se zaměřuje na teoretické zázemí práce. Bude v ní podrobně popsán obecný přístup ke klasifikaci textových dokumentů a násled-

6. http://firmy.cz
7. http://odkazy.seznam.cz

ně představeny vybrané algoritmy strojového učení, které budou použity v praktické části práce. Kapitola III pojednává o přípravě trénovacích dat a problémech při tvorbě korpusu, reprezentujícího příklady jednotlivých kategorií. Nejprve bude popsán problém crawlování, automatické detekce jazyka a kódování, poté se zaměříme na vytváření a značkování korpusů. V kapitole IV se budeme zabývat vytvořením vektorových modelů dokumentů. Součástí této kapitoly bude také popis metod shlukování termů včetně tvorby thesauru a výběru vhodných atributů pro strojové učení. V kapitole V popíšeme výsledky experimentů, získané jednotlivými metodami a porovnáme jejich klady a zápory. Dále z různých přístupů vybereme ten nejlepší, který by mohl být použit při praktickém nasazení aplikace. Kapitola VI je zaměřena na popis metod pro určení klíčových slov stránky. Zde již bude využito znalosti domény webové stránky. Kapitola VII shrne získané poznatky a bude pojednávat o možnostech dalšího směřování práce.

Kapitola 2

Užití strojového učení při klasifikaci textových dokumentů

Strojové učení (*machine learning*) je součástí velké rodiny počítačových problémů, které se týkají umělé inteligence. Jeho cílem je především návrh a vývoj algoritmů, které počítačům umožňují učit se novým poznatkům, jež nejsou explicitně definovány.
V zásadě lze metody strojového učení rozdělit na následující skupiny [33]:

- **Učení se zapamatováním (rote learning)** – systém pouze zaznamenává data nebo dílčí znalosti dodané externím zdrojem. Neprovádí se žádná transformace.

- **Učení se z instrukcí (learning from instructions)** – systém získává znalosti z externího zdroje a integruje je se znalostmi již získanými. Provádí se transformace znalostí ze vstupního jazyka do vnitřní reprezentace.

- **Učení se z analogie (learning by analogy)** – získávání znalostí je založeno na zapamatování si případů, resp. situací podobných těm, které bude třeba řešit v budoucnu.

- **učení na základě vysvětlení (explanation based learning)** – při učení se využívá několik málo příkladů a rozsáhlé databáze znalostí z dané oblasti (background knowledge).

- **Učení se z příkladů (learning from examples)** – zde se využívá velké množství příkladů (a protipříkladů) konceptu, který se má systém naučit. Role databáze znalostí naopak ustupuje do pozadí.

- **Učení se z pozorování a objevování (learning from observation and discovery)** – opět se pracuje s velkým množstvím dat, za využití indukce. Systém si ale často sám musí vytvářet koncepty, které se pak pokouší popisovat.

2.1 Klasifikační úloha

Metody klasifikace vycházejí z předpokladu, že jednotlivé objekty (příklady) patřící ke stejnému konceptu mají podobné charakteristiky. Pokud je každý příklad popsán n atributy, je možné jej zobrazit jako bod v n-rozměrném prostoru atributů. Objekty patřící ke stejnému konceptu potom v tomto prostoru vytvářejí shluky a cílem klasifikační úlohy je nalézt funkci, která rozhoduje příslušnost příkladů k těmto daným shlukům.
Pro formalizaci těchto úvah se inspirujeme popisem klasifikační úlohy z knihy [3]:

Analyzovaná data jsou uložena v tabulce D, tvořené n řádky a m sloupci.

$$D = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

Řádky tabulky reprezentují jednotlivé příklady, i-tý objekt je tedy řádek X_i.

$$X_i = \begin{bmatrix} x_{i1} & x_{i2} & \cdots & x_{im} \end{bmatrix}$$

Sloupce datové tabulky odpovídají atributům, j-tý atribut označíme symbolem A_j.

$$A_j = \begin{bmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{nj} \end{bmatrix}$$

Abychom mohli data klasifikovat, je třeba, aby existoval atribut, který obsahuje informaci o zařazení objektu do třídy. Tento atribut nazvěme *cílový atribut* a označme ho symbolem C. Ostatním, necílovým, atributům A_j říkejme *vstupní atributy*.

$$C = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Přidáme-li cílový atribut do datové tabulky, získáme data vhodná pro strojové učení. Těmto datům říkejme trénovací data a příslušnou tabulku označme D_{TR}.

$$D_{TR} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} & y_1 \\ x_{21} & x_{22} & \cdots & x_{2m} & y_2 \\ \vdots & \vdots & & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} & y_n \end{bmatrix}$$

Klasifikační úlohou rozumějme úkol najít takové znalosti reprezentované funkcí f, které by umožňovaly k hodnotám vstupních atributů nějakého objektu přiřadit vhodnou hodnotu cílového.

V průběhu klasifikace se pro hodnoty vstupních atributů X nějakého objektu odvodí hodnota cílového atributu. Označme tuto odvozenou hodnotu y'.

$$y' = f(X)$$

Odvozená hodnota y' se samozřejmě může pro nějaký konkrétní objekt od hodnoty y lišit, a proto je třeba počítat chybu klasifikace, která ukazuje, do jaké míry funkce f pokrývá správné hodnoty. Jádrem celého procesu klasifikace je tedy nalezení funkce f.

2.2 Klasifikace textových dokumentů

Klasifikace textových dokumentů je jedním z problémů strojového učení. Cílem je automaticky přiřadit textový dokument jedné nebo i více kategoriím v závislosti na jeho obsahu. Dokumenty můžeme klasifikovat např. podle tématu, žánru, autora, jazyka apod. Příkladem aplikace, kde se automatická klasifikace dokumentů již běžně používá, je např. detekce nevyžádané pošty (spamu).

Aby bylo možné použít běžné klasifikační algoritmy, je třeba textový dokument vhodně reprezentovat. Nejčastěji používanou reprezentací dokumentu je jeho vektorový model. V tomto modelu odpovídá dokument vektoru, v němž každá složka reprezentuje frekvenci slova či termu v dokumentu [23]. Frekvenci termu je někdy výhodnější nahradit nějakou normalizovanou hodnotou. Často je dokonce nejlepších výsledků dosaženo použitím binární charakterizace.

2.2.1 Selekce atributů

Pro vytvoření vektorového modelu nemusí být použita všechna slova, která se v trénovacích dokumentech vyskytují. Dokonce je to v případě dostatečného množství dat nevhodné. Dvěma extrémy nežádoucích slov jsou ta, která se v dokumentech vyskytují příliš často (jedná se předložky, spojky, pomocná slovesa a další termy, které nejsou pro klasifikaci rozhodující) a naopak velmi málo (pokud se slovo vyskytne v jediném trénovacím příkladu, nemá smysl jej uvažovat). Ze slov, která po odstranění hodně a málo četných zbudou můžeme dále vybrat ta nejzajímavější pomocí vhodné míry (relativní četnost, χ^2, MI-score, information gain, apod.) [13, 28].

2.2.2 Metody shlukování

Různí lidé pro popsání stejných konceptů nepoužívají stejná slova ani stejná slovní spojení. Proto může být výhodné sloučit slova stejných nebo podobných významů do skupin [25].

Nejjednodušší způsob shlukování slov je na základě stemizace nebo lemmatizace. To znamená, že jsou v jedné skupině slova se stejným kořenem nebo lemmatem. Pokud půjdeme do důsledků, nemusí to vždy znamenat, že by v jedné skupině byla slova se stejným významem. Existují totiž slova různých významů, která jsou i v původním textu různě zapsaná, ale mají stejná lemmata. Jiným příkladem potíží jsou případy, kdy právě morfologická informace u slova může rozlišit různé kategorie a stemizací či lemmatizací je nenávratně ztracena. Ve většině aplikací je však přesto výhodné stemizaci nebo lemmatizaci provést.

Slukování termů je dokonce možné dovést ještě dále, a to pomocí thesauru nebo jiného slovníku. Thesaurus je slovník podobných slov, přičemž podobnost nemusí znamenat jen synonymii, ale např. i antonymii, hypo/hyperonymii, mero/holonymii apod. U takových shluků je však problém nejednoznačnosti ještě markantnější než u stemizace či lemmatizace. Pravých synonym takových, jak je definuje Frege (výraz A je synonymní s výrazem B, jestliže lze oba výrazy v libovolné větě zaměnit a nezmění se tak její význam) je bohužel velmi málo, proto se běžně za synonymní výrazy považují i takové, které mají stejný význam jen v některých kontextech. Zde vzniká hlavní zdroj nejednoznačností, které mohou vést dokonce ke snížení celkové úspěšnosti klasifikace. Proto je třeba thesaurus tvořit s ohledem na tyto skutečnosti.

2.3 Vybrané metody strojového učení

V obecnosti nelze říci, které algoritmy strojového učení jsou nejlepší. Každá úloha může být tak specifická, že se pro ni jedny algoritmy hodí více a jiné méně. Nejinak je tomu u klasifikace textových dokumentů. Tato úloha je charakteristická především velkým počtem dimenzí vektorů.

Výběrem vhodným metod pro klasifikaci textů se v minulosti zabývala řada prací, jmenujme např. [48]. Jejich společným výsledkem bylo, že nejlepších výsledků dosáhly zpravidla metody k-nejbližších sousedů, naivní bayesovský klasifikátor a metoda podpůrných vektorů (Support Vector Machines). Pro porovnání výsledků přidejme do výčtu metod nejznámější algoritmus strojového učení – rozhodovací stromy a proveďme všechny experimenty s těmito metodami.

2.3.1 Rozhodovací stromy

Rozhodovací stromy pracují na principu algoritmů rozděl a panuj (*divide and concquer*). Trénovací data se postupně dělí na menší a menší množiny, ve kterých převládají příklady jedné třídy. Na začátku tvoří trénovací data jednu velkou množinu, na konci dostaneme rozklad této množiny tvořený příklady jedné třídy.

Tento postup bývá nazýván *top down induction of decision trees (TDIDT)*. Cílem je najít nějaký strom konzistentní s trénovacími daty, přičemž je kladen důraz na jednoduchost těchto výsledných stromů. Následuje popis obecného algoritmu TDIDT [3]:

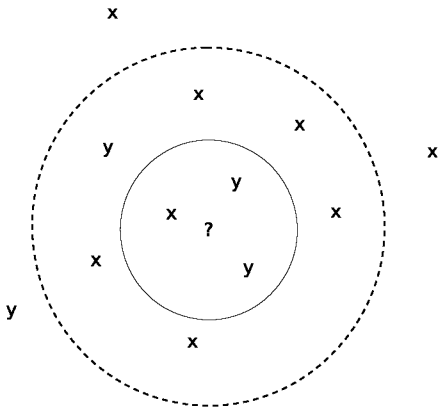
```
1. zvol jeden atribut jako kořen dílčího stromu
2. rozděl data v tomto uzlu na podmnožiny podle
   hodnot zvoleného atributu a přidej uzel pro
   každou podmnožinu
3. existuje-li uzel, pro který nepatří všechna
   data do téže třídy, pro tento uzel opakuj
   postup od bodu 1, jinak skonči
```

Je zřejmé, že uvedený algoritmus funguje pouze pro kategoriální data (nikoliv pro číselné atributy), která nejsou zatížena šumem. Proto je třeba číselné atributy diskretizovat a pro šumem zatížená data algoritmus upravit. Úprava může spočívat buď ve změně podmínky z bodu 3 (povolíme, aby do jednoho uzlu patřilo i několik málo příkladů z jiné třídy) nebo ve vhodném prořezávání výsledného stromu.

Klíčovým problémem algoritmu je výběr vhodného atributu pro větvení stromu. Tím se také liší jednotlivé algoritmy, založené na rozhodovacích stromech. Nejznámějším algoritmem je algoritmus C4.5 spojený se jménem Johna Rosse Quinlana. Tento algoritmus je rozšířením algoritmu ID3 a k výběru dělícího atributu využívá teorii informace, konkrétně informační entropii.

2.3.2 *k*-nejbližších sousedů

Jedná se o jednu z nejjednodušších metod strojového učení, u které výpočet funkce f probíhá až ve fázi klasifikace. Proto se řadí mezi algoritmy tzv. *líného učení.* Objekt je algoritmem klasifikován na hodnotu, převažující u jeho k nejbližších sousedů v prostoru atributů. K je typicky malé kladné celé číslo. Pokud je jeho hodnota 1, klasifikace se provádí pouze na základě třídy nejbližšího souseda. V případě binárních rozhodovacích problémů (problémů o dvou třídách) je výhodné zvolit k liché, aby se předešlo nejednoznačnostem. Jako metrika pro měření vzdáleností bývá většinou volena eukleidovská vzdálenost, ale z principu může být použita libovolná metrika. Na obrázku 2.1 je zachycen příklad binární klasifikace



Obrázek 2.1: Metoda *k*-nejbližších sousedů

ve dvourozměrném vektorovém prostoru metodou *k*-nejbližších sousedů. Znak otazníku představuje klasifikovaný objekt, znaky x příklady jedné třídy, znaky y příklady druhé třídy. Pokud zvolíme $k = 1$, bude neznámý objekt klasifikován jako x. V případě $k = 3$ bude klasifikován jako y, neboť v definovaném okolí převládají objekty y (vyznačeno plnou kružnicí). Při

volbě $k = 9$ bude neznámý objekt opět klasifikován jako x (vyznačeno čárkovanou kružnicí).

Vhodná volba hodnoty *k* je klíčová pro úspěch algoritmu. Obecně se dá říci, že větší hodnoty redukují šum, ale smazávají rozdíly mezi jednotlivými třídami. Pro výběr správné hodnoty k se dají použít různé heuristiky jako např. křížová validace [21].

2.3.3 Naivní bayesovský klasifikátor

Naivní bayesovský klasifikátor je metoda strojového učení založená na Bayesově větě o podmíněné pravděpodobnosti. Naivní je nazývána proto, že předpokládá absolutní nezávislost vstupních atributů. Tento předpoklad sice u většiny praktických aplikací splněn není, ale při dostatečném počtu atributů, který zajistí redundanci informace, dosahuje algoritmus velmi dobré přesnosti.
Pro formální definici se inspirujme knihou [3]:

Pravděpodobnost hypotézy H za předpokladu platnosti evidence E je podle Bayesova vztahu definována:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Pravděpodobnost $P(H)$, která odpovídá pravděpodobnosti jevu H bez znalosti jakékoliv další informace se nazývá *apriorní pravděpodobnost*, pravděpodobnost $P(H|E)$ se nazývá *aposteriorní pravděpodobnost*. Veličina $P(E)$ vyjadřuje pravděpodobnost evidence E.

Hypotéz, mezi kterými se rozhodujeme je v případě klasifikační úlohy více, nás bude zajímat ta nejpravděpodobnější. Pro každou hypotézu $H_t, t \in \{1, \ldots, T\}$ můžeme spočítat aposteriorní pravděpodobnost a vybrat H_{MAP} (maximum aposteriory probability) s největší $P(H_t|E)$.

$$H_{MAP} = H_J$$
$$\Longleftrightarrow$$
$$P(H_J|E) = max\left\{\frac{P(E|H_t)P(H_t)}{P(E)}|t \in \{1, \ldots, T\}\right\}$$

Vzhledem k tomu, že nás většinou zajímá pouze to, pro které H_t je aposteriorní pravděpodobnost maximální a nezajímá nás její hodnota, můžeme výraz zjednodušit:

$$H_{MAP} = H_J$$
$$\Longleftrightarrow$$
$$P(E|H_J)P(H_J) = max\{P(E|H_t)P(H_t)|t \in \{1,\ldots,T\}\}$$

Nyní již přejděme od matematiky ke strojovému učení. Jak už je zřejmě patrné, hypotézy pro nás představují jednotlivé třídy a evidence hodnoty atributů. V případě K atributů a T tříd je funkce $f(X)$ definována:

$$f(X) = H_J$$
$$\Longleftrightarrow$$
$$P(H_J)\prod_{k=1}^{K} P(E_k|H_J) = max\{P(H_t)\prod_{k=1}^{K} P(E_k|H_t)|t \in \{1,\ldots,T\}\}$$

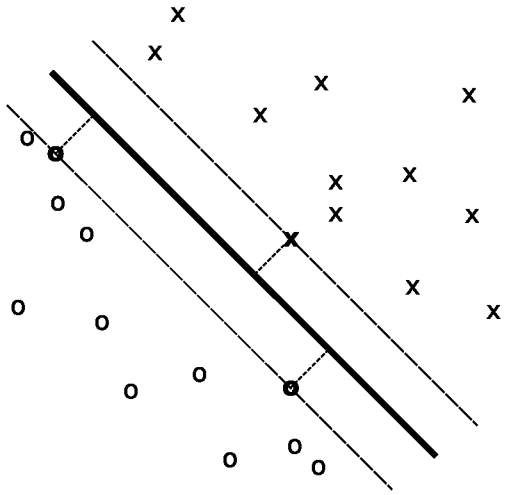
Abychom mohli tento způsob klasifikace použít, potřebujeme znát hodnoty $P(H_t)$ a $P(E_k|H_t)$. Ty se nahradí relativními četnostmi v trénovacích datech. Na rozdíl od většiny jiných metod strojového učení se neprovádí prohledávání prostoru hypotéz, výpočet je tedy velmi rychlý.

2.3.4 Support Vector Machines

Support Vector Machines (metoda podpůrných vektorů) je relativně novou metodou [8], jejíž obliba díky dobrým výsledkům klasifikace stále stoupá.

Hlavní myšlenkou je nalezení rozdělující nadroviny, separující jednotlivé příklady do tříd. V případě, že data nejsou lineárně separovatelná se provede vhodná datová transformace, která převede úlohu na vícerozměrný problém, ve kterém již lineární separace možná je. Místo v prostoru původních atributů se tedy pohybujeme v prostoru transformovaných atributů. Ze všech nadrovin, které data separují vybereme tu, která má největší odstup od transformovaných příkladů trénovací množiny a přitom odděluje jednotlivé třídy. Příklady, které jsou nejblíže této nadrovině se nazývají podpůrné vektory (odtud název metody).

Obrázek 2.2 zobrazuje příklad lineární separace dvou tříd ve dvourozměrném prostoru. Tučně je vyznačena separující nadrovina (v našem, dvourozměrném, prostoru přímka) a podpůrné vektory. Zjednodušenou úlohu lineárního klasifikátoru do dvou tříd formálně definujme podle [41]:



Obrázek 2.2: Metoda Support Vector Machines

Nechť trénovací množinu tvoří množina bodů v p-rozměrném prostoru

$$\{(\overrightarrow{x_1}, c_1), (\overrightarrow{x_2}, c_2), \ldots, (\overrightarrow{x_n}, c_n)\}$$

kde c_i je buď 1 nebo -1 v závislosti na tom, do jaké třídy přísluší bod $\overrightarrow{x_i}$. Každý bod $\overrightarrow{x_i}$ je p-dimenzionální vektor. Cílem je nalézt nadrovinu s maximálním odstupem od příkladů různých tříd, současně oddělující body, pro které $c_i = 1$ od bodů s $c_i = -1$. Každá nadrovina může být zapsaná jako množina bodů $\overrightarrow{x}$, splňující

$$\overrightarrow{w} \cdot \overrightarrow{x} - b = 0$$

Vektor $\overrightarrow{w}$ je normálový vektor, kolmý na nadrovinu. Parametr b určuje posunutí od počátku souřadnicového systému.

Chceme zvolit $\overrightarrow{w}$ a b tak, aby vzdálenost mezi dvěma paralelními nadrovinami byla co největší a zároveň oddělovaly obě třídy. Všimněme si, že zde je užit předpoklad lineární separability. Tyto nadroviny mohou být popsány dvěma rovnicemi

$$\overrightarrow{w} \cdot \overrightarrow{x_i} - b = 1$$

$$\overrightarrow{w} \cdot \overrightarrow{x_i} - b = -1$$

Z geometrie víme, že vzdálenost těchto dvou nadrovin je $\frac{2}{|\overrightarrow{w}|}$, chceme tedy minimalizovat $\overrightarrow{|w|}$ při splnění omezení

$$\overrightarrow{w} \cdot \overrightarrow{x_i} - b \geq 1, c_i = 1$$
$$\overrightarrow{w} \cdot \overrightarrow{x_i} - b \leq -1, c_i = -1$$

Všechny tyto požadavky nám dohromady dávají optimalizační problém, který řeší danou úlohu:
Najdi $\overrightarrow{w}$ a b tak, aby $|\overrightarrow{w}|$ bylo minimální při splnění omezení $c_i(\overrightarrow{w} \cdot \overrightarrow{x_i} - b) \geq 1$ pro všechna $1 \leq i \leq n$, kde n je počet příkladů.

V případě, že data nejsou lineárně separabilní, je třeba převést úlohu na lineárně separabilní. K tomu se využívá tzv. „kernel trick“, který převede původní úlohu do vícerozměrného vektorového prostoru, kde již data lineárně separabilní jsou.

Formální definice obecného principu Support Vector Machines a především transformace na lineárně separabilní úlohu je relativně komplikovaná a přesahuje rámec této diplomové práce. Případné zájemce odkažme například na [5].

Kapitola 3

Příprava trénovacích dat

3.1 Použité kategorie pro klasifikaci

Aby mohla být klasifikace webových stránek úspěšná, je třeba vhodně zvolit jednotlivé kategorie. Pro naše účely není důležité o čem konkrétně webová stránka pojednává, ale zajímáme se o její žánr.

Žánr je dnes často používaný pojem především ve spojení s hudbou, literaturou, filmem apod. Zde se zajímáme o žánr HTML stránek. V minulosti bylo podáno několik definic žánru textových dokumentů [4, 16, 17], my se ztotožníme s tou nejpoužívanější: „The genre describes something about what kind of document it is rather than what the document is about" [11]. Nebudeme se tedy zabývat podrobnou analýzou obsahu textu či jiných částí dokumentu, ale půjde nám o zařazení dokumentu do nějaké z doménových tříd.

Zvolené kategorie by v souladu s předchozí definicí žánru HTML stránky měly splňovat především následující požadavky:

- **Dobrá reprezentativnost** – kategorie by měly pokrývat všechny žánry, které se mohou na českém webu objevit.
- **Co nejmenší překrývání jednotlivých kategorií** – jen těžko dokážeme zvolit kategorie tak, aby zařazení jednotlivých webových stránek bylo vždy úplně jednoznačné, avšak vhodnou volbou kategorií je možné počet nejednoznačných stránek eliminovat.
- **Rozumný počet kategorií s ohledem na následné klasifikační techniky** – je zřejmé, že s větším počtem kategorií klesá úroveň jejich překrývání. Je však třeba brát zřetel na proces strojového učení. S velkým počtem kategorií totiž výrazně klesá přesnost.

Výběr vhodných kategorií je komplikovaná záležitost, která vyžaduje netriviální lidské zdroje (v případě bližšího zájmu můžeme odkázat čtenáře na [43]), navíc je třeba pro každou kategorii vybrat dostatečné množství vhodných reprezentantů pro fázi strojového učení. Proto se práce tímto

problémem nezabývá, ale je využito top-level kategorií z katalogu `http://odkazy.seznam.cz`, které splňují výše uvedené požadavky a jsou prověřeny praxí. Jejich seznam spolu se stručnou charakterizací je uveden v tabulce 3.1.

Kategorie	**Popis**
Cestování	*cestopisy, jízdní řády, mapy a atlasy, turistické cíle, ubytování a stravování*
Erotika	*chaty a fóra, seznamky, obrázky, videa, zboží*
Hry	*hlavolamy a křížovky, deskové hry, počítačové hry, on-line hry*
Informační a inzertní servery	*firmy a organizace, finanční a bankovní informace, gastronomie, úřady a politika, inzerce, zdraví*
Kultura a umění	*divadlo, film, fotografie, hudba, literatura, výtvarné umění, kulturní dění*
Lidé a společnost	*ankety a průzkumy veřejného mínění, chatování, diskuzní fóra, fankluby, osobní stránky, seznamky, weblogy, sdružení a spolky*
Počítače a internet	*software, hardware, multimédia, internet, grafika, počítačové manuály a návody, bezpečnost a hacking, programování*
Sport	*fotbal, hokej, motoristické sporty, cyklistika, rybaření, vodní sporty...*
Věda a technika	*přírodní a technické vědy, společenské a humanitní vědy, vědecké konference a semináře, elektronika, elektrotechnika, stavebnictví, železniční technika*
Volný čas a zábava	*chovatelství, modelářství, humor a zábava, sběratelství, pěstitelství, pohádky a pověsti, koníčky a záliby*
Zpravodajství	*denní tisk, dopravní zpravodajství, informace z regionů, počasí, rádio, televize, internetové časopisy*

Tabulka 3.1: Použité kategorie

3.2 Sběr dat

Pro úspěšné použití strojového učení jsou nezbytná kvalitní trénovací data. Stejně jako pro výběr kategorií posloužil katalog webových stránek `http://odkazy.seznam.cz` pro sběr trénovacích příkladů. Tento kata-

log obsahuje odkazy na úvodní stránky jednotlivých webů, zařazené do příslušných kategorií. Otázkou však je, zda tyto úvodní stránky dostatečně reprezentují obsah daného webu. Často se setkáváme s tím, že na úvodní stránce je jen nadpis, navigace a minimum informací.

Řešením tohoto problému, představeným v této práci, může být rozšíření dokumentu o obsah stránek odkazovaných z úvodní stránky. Zde však narážíme na problém přechodu do jiné třídy. Snadno se může stát, že odkaz povede na stránku, která je jiného žánru než zdrojová. Počet takových odkazů se dá přesto velice účinně redukovat na minimum dodržením následujících dvou zásad:

- **Použití výhradně stránek ze stejné domény** – neuvažují se odkazy, které směřují na jinou internetovou doménu a tedy většinou i na jiný server. Nebudou tak použity reklamní odkazy a jiné stránky, které se přímo netýkají daného webu.
- **Nepoužití rekurzivního prohledávání** – odkazy jsou prohledávány pouze v úvodní stránce, nikoliv rekurzivně v dalších odkazovaných stránkách. Tím se výrazně sníží pravděpodobnost odbočení od tématu.

Za účelem potvrzení předpokladu, že při použití těchto pravidel bude počet nežádoucích odkazů zanedbatelný byl proveden jednoduchý experiment. Z katalogu bylo náhodně vybráno 100 různých stránek, u kterých byly ručně analyzovány žánry jejich odkazů. Výsledek experimentu prokázal, že 98% linků skutečně odkazovalo na stránky stejného žánru. Pro účely strojového učení lze 2% chybu v trénovacích datech zanedbat, proto je tento způsob rozšíření trénovacích dat velmi dobře použitelný.

Postupně byly z jednotlivých kategorií stahovány stránky, dokud velikost stažených dat pro danou kategorii nepřesáhla 1GB. Vzhledem k tomu, že hyperlinky mohou odkazovat nejen na HTML stránky, ale i na soubory jiných formátů (např. obrázky, videa, zvukové soubory apod.), byl vytvořen filtr, který zakázal stahování souborů jejichž tzv. *content-type* (typ obsahu) byl různý od HTML. Kromě těchto souborů nebyly staženy ani žádné stránky nacházející se v doméně *.sk*. Tím se výrazně eliminoval počet slovensky psaných stránek, které jsou v katalogu často zařazeny.

3.3 Čištění trénovacích dat

Pro vytvoření korpusu trénovacích dat z webu je třeba ze stránek odstranit jejich části, které pro námi zvolený přístup ke klasifikaci nemají žádný nebo

minimální význam. Taková data mohou velmi negativně ovlivnit přesnost klasifikace.

Čištění dat získaných z webu je dnes pro vytváření korpusů velmi aktuální téma. Obecně platí, že čím větší korpus, tím lepších výsledků lze při práci s ním dosáhnout, a právě velké množství textových dat lze snadno získat z internetu. V roce 2007 se konal první ročník soutěže Cleaneval [1] spolu s workshopem zaměřenými právě jen na metody čištění webových stránek. Vítězem se stal systém založený na CRF (*Conditional random fields*) [26], který se však bez úprav pro naše účely nedá dobře použít. Tyto nástroje jsou určeny k čištění korpusů. Naším cílem není vytvořit kvalitní korpus, ale sestavit trénovací data pro strojové učení. V případě, že by webová stránka, kterou chceme klasifikovat neobsahovala data vhodná pro vytvoření korpusu, byla by celá, nebo její velká část zahozena a nebylo by podle čeho určit její zařazení do správné kategorie. Je tedy třeba zachovat v korpusu co nejvíce informací i za cenu, že trénovací data zatížíme malým šumem.

Nejprve se podívejme na definici správně vytvořené HTML stránky. O definice dokumentů HTML a XHTML se stará konsorcium W3C. Na webových stránkách konsorcia [2] lze nalézt nejen formální DTD (*Document Type Definition*), což je jazyk pro popis struktury XML případně SGML dokumentu, ale i neformální tutoriály o vytváření HTML stránek.

HTML stránka je tvořena ze dvou základních komponent – hlavičky (`<head>`) a těla (`<body>`). V hlavičce jsou uloženy především meta informace, skripty jiných jazyků (typicky JavaScriptu) a informace o vzhledu stránky. Meta informace webové stránky mohou obsahovat důležité informace. S rozvojem fulltextových vyhledávačů, jež k nim přikládaly velký význam však došlo k jejich zneužívání. Autoři stránek do nich vkládali nepravdivé informace, čímž chtěli dosáhnout lepší pozice ve výsledcích vyhledávání. V dnešní době je již sice vyhledávače většinou ignorují, přesto ale nemusí poskytovat spolehlivé informace. Jediným údajem z hlavičky, který je zařazován do korpusu je tedy titulek (`<title>`). Jedná se o informaci, která je ve webových prohlížečích viditelná, tudíž příliš nehrozí její zneužívání.

Hlavním zdrojem podkladů pro vytvoření trénovacích příkladů je tělo webové stránky (`<body>`). Z něho jsou v první fázi odstraněny komentáře a všechny typy skriptů. Pro další zpracování strukturních informací je třeba zachovat HTML tagy, které vyznačují logickou strukturu textu. K těmto

1. http://cleaneval.sigwac.org.uk/
2. http://www.w3.org

účelům byly vybrány značky, které jsou často zohledňovány i fulltextovými vyhledávači:

- název stránky(`<title>`)
- odstavce(`<p>`)
- nadpisy (`<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`)
- hyperlinky (`<a>`)
- odrážky (`<li>`)

Ostatní HTML tagy byly odstraněny. V poslední fázi byla data zbavena dlouhých netextových sekvencí, jako jsou číselná data v tabulkách nebo jiný, lingvisticky těžko zpracovatelný materiál. Algoritmus v textu hledá všechny n-gramy termů pro n větší než 10, ve kterých je podíl nealfanumerických znaků větší než 50%. Sekvence termů jsou zarovnány na věty a následně z dokumentu odstraněny. Takto vyčištěný dokument slouží jako hrubý datový podklad pro vytvoření korpusu. Následuje automatická detekce kódování znaků, rozpoznání jazyka dokumentu, odstranění stránek, které nejsou psány česky a samotné vytvoření korpusu.

3.4 Automatická detekce kódování znaků

Na začátku počítačové éry a postupem času bylo vytvořeno mnoho kódování pro reprezentaci různých znakových sad národních abeced. S nástupem internetu však došlo k prudkému nárůstu potřeby sdílet informace napříč různými jazyky, národy i platformami. Existence velkého množství kódování tak přinesla nemalé problémy. Jako řešení vzniklo nové univerzální kódování UNICODE [3]. Bohužel, z mnoha různých důvodů jím stále ještě ostatní kódování nahrazena nebyla.

V našem, českém, národním prostředí se v současné době používají v naprosté většině případů 3 různá kódování: UTF-8, což je implementace znakové sady UNICODE, středoevropské kódování podle normy ISO-8859-2 a středoevropské kódování používané v produktech firmy Microsoft CP-1250.

Pro korpus bylo vybráno kódování UTF-8, které je dnes již standardem, a bylo tedy nutné do něho stránky napsané v jiných kódováních převést. Informace o kódování bývá obvykle uložena v hlavičce stránky, podobně

3. http://unicode.org/

jako u jiných údajů v hlavičce to však nemusí být pravidlem. Je tedy nutné detekovat kódování automaticky.

Volně šiřitelných nástrojů pro automatickou detekci kódování je k dispozici hned několik. Uveďme například Mozilla chardet [4] nebo Universal encoding detector [5]. Společným jmenovatelem těchto produktů je snaha o univerzalitu a s ní spojená i snížená přesnost. Experimenty prokázaly, že u těchto nástrojů dochází často k záměně ISO-8859-2 za CP-1250 a naopak, což je samozřejmě neakceptovatelné.

K našim účelům potřebujeme detekovat pouze kódování používaná pro češtinu. Tím je výrazně zúžena zkoumaná doména a je tedy možné poměrně snadno vytvořit nástroj, který bude pro tuto oblast kvalitnější.

Metoda, která byla pro automatickou detekci kódování použita, je založena na frekvenční analýze bajtů zkoumaného dokumentu [39]. Jako trénovací vzorek posloužil 100 megabajtový vzorek českého textu z korpusu SYN2000. Ten byl z původního kódování ISO-8859-2 převeden pomocí nástroje *iconv* [6] do UTF-8 i CP-1250 a následně byla provedena analýza relativních četností bajtů v každém ze tří vzorků. Výsledné modely jsou zachyceny na obrázku 3.1. Z obrázku je dobře vidět podobnost mezi modely ISO-8859-2 a CP-1250 a naopak odlišnost modelu UTF-8, především v oblasti vyšších bitů. To je dáno navržením UTF-8 tak, že znaky z rozsahu #00 až#7F kóduje stejným způsobem jako US-ASCII a všechny znaky vetší než #7F kóduje do sekvence několika bajtů, přičemž je nejvýznamnější bit nastaven na 1, aby nemohlo dojít k záměně s některým z US-ASCII znaků.

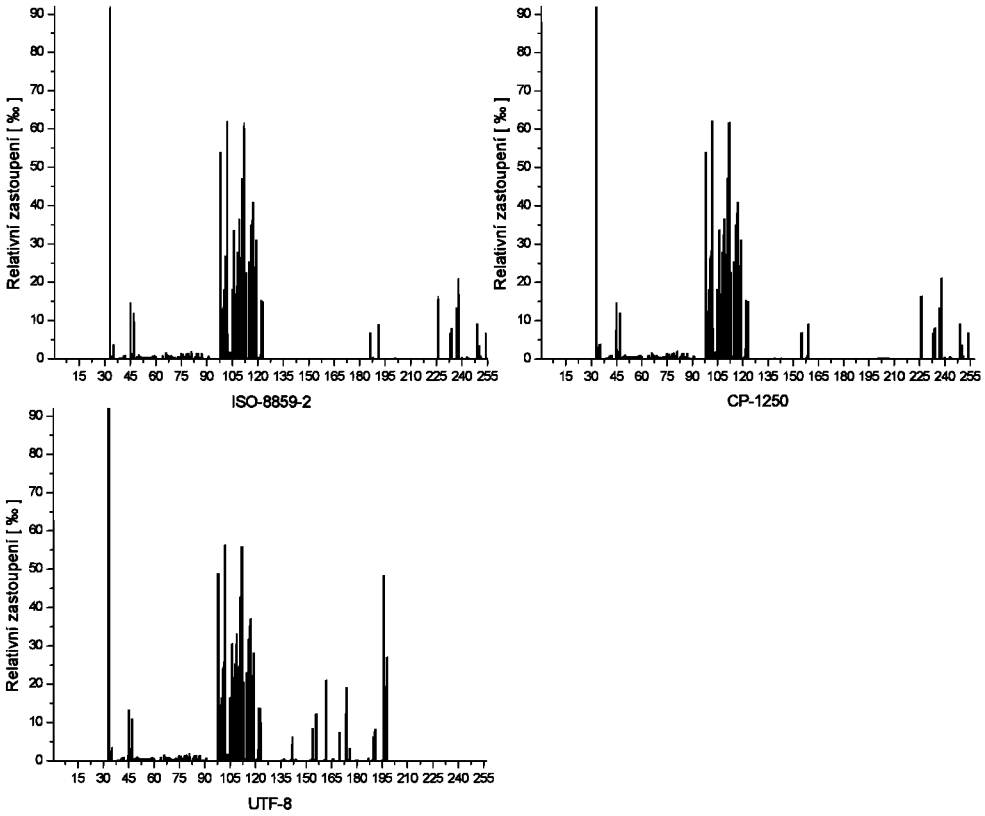
Problémem je tedy především rozlišení ISO-8859-2 a CP-1250. Podívejme se podrobněji do tabulky 3.2 na bajty, které tyto dva modely rozlišují. Pro češtinu jsou to především znaky š, Š, ť, Ť, ž, Ž.

Na základě těchto dat spolu s četnostmi pro kódování UTF-8 byly vytvořeny rozhodovací modely. Experimenty byly provedeny se dvěma typy modelů – spojitým a diskrétním. Spojitým se rozumí ponechání relativních četností ve formě reálných čísel, diskrétním rozdělení relativních četností do intervalů. Podívejme se podrobněji na definice obou typů modelů.

3.4.1 Spojitý model

Nechť $M_{utf}, M_{iso}, M_{win}$ jsou modely jednotlivých kódování, $M_C(x)$ pro $C \in \{utf, iso, win\}, x \in \{0 \dots 255\}$ jejich relativní četnosti bajtů a M_a model zkoumaného dokumentu a.

4. http://www.mozilla.org/projects/intl/chardet.html
5. http://chardet.feedparser.org/
6. http://www.gnu.org/software/libiconv/



Obrázek 3.1: Modely různých kódování

Vzdálenost dokumentu a od modelu M_C definujeme jako eukleidovskou:

$$d(a, C) = \sqrt{\sum_{i=0}^{255} (M_a(i) - M_C(i))^2}$$

Dokumentu je přirozeně přiřazeno kódování, jehož model je mu nejblíže.

3.4.2 Diskrétní model

Před samotnou definicí vzdálenosti je nejprve třeba rozdělit hodnoty relativních četností do intervalů. Při stanovování počtu intervalů proti sobě stojí dva požadavky – čím více intervalů, tím lepší diskriminace jednotlivých kódování a naopak čím méně intervalů, tím lepší schopnost generalizace. Protože rozhodnutí optimálního počtu není triviální záležitost, byly

Kód bajtu	četnost v ISO-8895-2 [promile]	četnost v CP-1250 [promile]
138	0.0	0.163415090375
141	0.0	0.00238657560439
142	0.0	0.120697450379
154	0.0	6.74169780672
156	0.0	5.43341420019e-04
157	0.0	0.470533669736
158	0.0	8.95264345539
159	0.0	1.9257670583e-04
161	1.78821226842e-04	0.0183154202723
166	7.5655134433e-05	0.0
169	0.163415090375	0.0
171	0.00238657560439	0.0
174	0.120697450379	0.0
177	1.16921571396e-04	0.0
181	0.00143056981473	0.0
182	5.43341420019e-04	0.0
183	0.0183154202723	0.0
185	6.74169780672	1.16921571396e-04
187	0.470533669736	0.0
190	8.95264345539	0.00143056981473

Tabulka 3.2: Četnosti bajtů

provedeny 3 experimenty pro počty intervalů 2, 3 a 4.

Hranice intervalů byly zvoleny tak, aby jednotlivá kódování byla co nejlépe rozlišitelná, především s ohledem na podobnost ISO-8859-2 A CP-1250. Vzdálenost dokumentu od diskrétního modelu definujeme následovně:

$$d(a, C) = \sum_{i=0}^{255} |\{M_a(i)\} - \{M_C(i)\}|$$

$M_C(x) \in H$, $M_a(x) \in H$ pro libovolné $x \in \{0 \dots 255\}$, kde

- $H = \{0, 1\}$ pro 2 intervaly
- $H = \{0, 1, 2\}$ pro 3 intervaly
- $H = \{0, 1, 2, 3\}$ pro 4 intervaly

Všechny čtyři typy modelů byly testovány na souboru 3000 testovacích dokumentů. Nejlepších výsledků dosáhl diskrétní model se třemi intervaly, kde byla přesnost na testovacím vzorku 99,93% (2 dokumenty označeny chybně). Při analýze chybně označených dokumentů se dokonce ukázalo, že mohou odpovídat i kódování označenému systémem.

3.5 Automatická detekce jazyka

Většina větších webových portálů nabízí uživateli možnost zvolit si jazyk, ve kterém budou stránky zobrazeny. Některé stránky v českém katalogu jsou dokonce primárně psány v cizím jazyce (typicky ve slovenštině) nebo obsahují cizojazyčné části (diskuze, weblogy). S ohledem na další zpracování korpusu, které je jazykově závislé, je třeba tyto dokumenty odstranit.

Automatická detekce jazyka je, podobně jako u detekce kódování, založena na frekvenční analýze. Vědecké výzkumy psychologů totiž ukázaly, že člověk je schopen z textu správně rozpoznat cizí jazyk i když nerozumí žádnému ze slov. Princip je v tom, že pro každý jazyk jsou charakteristické jiné posloupnosti hlásek, v textové podobě potom i písmen. Člověk jazyk podvědomě identifikuje právě na základě pravděpodobnosti výskytu těchto posloupností. Dokonce existují na tomto principu založené generátory pseudo textů, např. [42].

Pro účely sestavení modelu češtiny byl vytvořen 100 megabajtový korpus českých webových stránek. Data byla získána crawlováním české verze internetové encyklopedie Wikipedia [7]. Na stažená data byly aplikovány všechny fáze čištění popsané výše. Kódování detekovat třeba nebylo, neboť Wikipedia standardně používá UTF-8. Co se týče jazyku, mělo by být zaručeno, že všechny české stránky Wikipedie jsou skutečně psány česky.

Z tohoto korpusu byl vytvořen model češtiny. Pro každý 3-gram (posloupnost třech znaků, nikoliv bajtů jako u detekce kódování) byla napočítána jeho relativní četnost a tyto četnosti uloženy jako jeden vektor.

Při určování podobnosti zkoumaného dokumentu a modelu je vytvořen vektor relativních četností v dokumentu stejným způsobem jako u modelu jazyka. 3-gramy, které se vyskytují pouze v jednom vektoru jsou v druhém doplněny nulovou hodnotou, přičemž musí platit, že stejné složky reprezentují v obou vektorech stejné 3-gramy. Podobnost dokumentu a modelu jazyka je definována kosinovou mírou jako kosinus úhlu, který oba vektory svírají, tedy 1 pokud jsou vektory totožné a 0

7. http://cs.wikipedia.org/

v případě absolutně odlišných.
Formálně, nechť $m = (m_1, m_2, \ldots, m_n)$ je vektor představující model jazyka a $a = (a_1, a_2, \ldots, a_n)$ vektor reprezentující zkoumaný dokument. Podobnost tohoto modelu m a dokumentu a definujeme jako:

$$S(m, a) = \frac{m_1 a_1 + m_2 a_2 + \cdots + m_n a_n}{\sqrt{m_1^2 + m_2^2 + \ldots m_n^2} \cdot \sqrt{a_1^2 + a_2^2 + \ldots a_n^2}}$$

Při rozhodování, zda některý dokument z korpusu odebrat nebo ne se porovnává dokument pouze s modelem češtiny. V případě, že podobnost dokumentu s modelem nepřekročí určitou, předem stanovenou hodnotu, je dokument odstraněn. Tímto principem se odstraní nejen cizojazyčné dokumenty, ale i dokumenty, které jsou sice česky, ale obsahují velké množství číselných údajů nebo speciálních značek, které by po přidání do korpusu zhoršily jeho kvalitu. Mezní hodnota podobnosti byla experimentálně stanovena na 0,85. Tato hodnota je stanovena velmi střízlivě, to znamená, že jsou odstraněny pouze ty dokumenty, které jsou v jiném jazyce s velkou pravděpodobností.

3.6 Tvorba korpusu

Z vyčištěných dat již je možné vytvořit korpus. Korpus je (většinou rozsáhlý) soubor textů, které jsou v různé míře opatřeny metajazykovými značkami, vypovídajícími o samotném dokumentu (autor, rok vydání, žánr apod.) a zařazení jednotlivých slov do kategorie slovních druhů, o frekvenci slova v korpusu, případně dalších lingvistických a frekvenčních aspektech. Některé korpusy jsou budovány jako takzvaně vyvážené, což znamená, že by měly obsahovat vyvážený podíl textů tříděných podle žánru, doby vzniku, případně dalších hledisek (mluvený, psaný, regionálně zaměřený apod.) [40].

Pro práci s korpusy slouží korpusové manažery. Korpusový manažer je počítačový program, který umožňuje efektivní práci s korpusy. Efektivností se rozumí pokud možno konstantní časová složitost základních dotazů vzhledem k velikosti korpusu, což je umožněno především díky uložení dat ve speciálním formátu a sofistikovaným algoritmům.

Mezi základní operace, které by měl korpusový manažer poskytovat patří nalezení konkordancí (seznamu všech výskytů jistého jevu v korpusu, přičemž jevem může být např. slovo nebo slovní skupina jistých vlastností), statistických informací jako globálních a lokálních četností, kolokací

(víceslovných výrazů obvykle spojovaných do samostatných lexikálních jednotek) apod.

Pro práci s naším korpusem byl použit korpusový manažer Manatee [34]. Kromě základních operací nad korpusy poskytuje celou škálu statistických funkcí a díky propojení s nástrojem Sketch Engine také tzv. *word sketches* (profily slov), což jsou informace o gramatických kontextech, počítané na základě statistiky a sady gramatických pravidel. Tyto informace jsou později v práci použity pro shlukování termů.

Jako zdrojový formát korpusového manažeru Bonito slouží vertikální text. Vertikální text je běžný textový soubor, ve kterém jsou všechny pozice vyznačeny na jednom řádku. Pokud korpus obsahuje kromě slov nějaké další atributy, jsou zapsány společně v jednom řádku, oddělené tabulátory. Strukturní značky a metadata jsou uloženy v podobě XML, také vždy na samostatném řádku.
V našem korpusu jsou zachyceny následující atributy:

- **word** – slovo v takovém tvaru, v jakém bylo použito ve zdrojových textech
- **lemma** – základní tvar slova
- **tag** – morfologická značka v „brněnské" notaci
- **imp** – údaj zachycující významnost pozice v HTML dokumentu. Významnost je učena analogicky jako u fulltextových vyhledávačů, kde např. slovo v nadpisu hraje větší význam než slovo vyskytující se v odstavci po ním. V korpusu byla použita následující škála významností:
  1. `t` – tag `<title>`, slovo obsaženo v názvu dokumentu
  2. `h` – tagy `<h1>`, `<h2>`, `<h3>` `<h4>`, `<h5>`, `<h6>`, nadpisy všech úrovní
  3. `a` – tag `<a>`, hypertextový odkaz
  4. `l` – tag `<li>`, odrážka seznamu
  5. `p` – tag `<p>`, prostý text

Údaje zachycují významnost slov odpovídají strukturní podobě dokumentu a jeho logickému členění. Z tohoto důvodu by měly být vyznačeny spíše jako strukturní značky. Od tohoto značení však bylo upuštěno s ohledem na další zpracování dat, pro které je vyznačení ve formě atributu výhodnější.

Mezi strukturní značky, které použity byly patří:

- `<class>` – vyznačující hranice jednotlivých top-level kategorií
- `<subclass>` – vyznačující hranice podkategorií
- `<doc>` – vyznačující hranice dokumentů
- `<s>` – vyznačující hranice slov

Příklad vertikálního textu může vypadat následovně:

```
<class="cestovani">
<doc id=2>
<s>
OFICIÁLNÍ       oficiální     k2eAgFnSc1d1    h
JEDNOTNÁ        jednotný      k2eAgFnSc1d1    h
KLASIFIKACE     klasifikace   k1gFnSc1        h
UBYTOVACÍCH     ubytovací     k2eAgNnPc2d1    h
ZAŘÍZENÍ        zařízení      k1gNnPc2        h
V               v             k7c6            h
ČESKÉ           český         k2eAgFnSc6d1    h
REPUBLICE       republika     k1gFnSc6        h
</s>
<s>
...
```

Uvedený příklad zachycuje začátek druhého dokumentu v kategorii cestování. Jedná se o první větu, která je nadpisem.

V prvním sloupci vertikálního textu je uvedeno slovo v originální podobě, v druhém sloupci je základní tvar (lze si všimnout, že kromě převedení na lemma jsou i všechny kapitálky změněny na malá písmena), ve třetím sloupci je morfologická značka slova tak, jak ji reprezentuje morfologický analyzátor Ajka [38] a poslední, čtvrtý, sloupec zachycuje významnost slova určenou na základě strukturního členění dokumentu (v našem případě jde o nadpis).

3.6.1 Tokenizace a detekce hranic vět

Základním stavebním kamenem korpusu jsou části textů, které se nazývají pozice nebo *tokeny* (odtud je tento proces rozdělení do pozic označován jako *tokenizace*) a velmi často odpovídají rozdělení na slova.

Problémem je však najít hranice těchto pozic. Texty totiž neobsahují jenom písmena a mezery, ale ještě číslice a celou řadu speciálních značek jako jsou čárky, tečky, vykřičníky, pomlčky apod. Slova navíc nemusí být oddělena jen mezerou, ale např. novým řádkem, odstavcem nebo nemusí být oddělena vůbec. Příkladem je interpunkce, která se píše bezprostředně za předchozí slovo.

Za jeden token tedy považujeme nejmenší samostatný celek který odpovídá slovu, číslici nebo symbolu. Toto zjednodušení nám sice umožní dělit texty na jednotlivé pozice, přináší však s sebou jisté problémy:

1. **Rozdělení pevných kolokací.** Slovní spojení, která po rozdělení buď nedávají smysl nebo nabudou jiných významů, by měla zůstat jako jedna pozice. Např. latinské výrazy typu *ad hoc*. Takováto slovní spojení je třeba detekovat, například pomocí kolokačního slovníku.

2. **Nejednoznačnost rozdělení.** Otázkou je, jak by se měla správně rozdělit např. spojení obsahující spojovník typu *propan-butan* nebo *víš-li*. Jednou z možností je nechat celé spojení jako jednu pozici, druhou je vytvořit ze spojovníku jeden token a získat tak 3 pozice. Spojení *víš-li* můžeme dokonce rozdělit na 2 pozice jako [*víš*, *-li*].

   S ohledem na další zpracování bylo v našem korpusu použito co nejjemnější dělení.

Dalším problémem souvisejícím s tokenizací je detekce hranic vět. Jednoduché pravidlo, které říká, že věta končí po interpunkčním znaménku, následovaném velkým písmenem sice pokrývá velkou část případů, ale nelze se na něj spolehnout.

Jednoduchým protipříkladem jsou zkratky. Např. spojení „Ing. Ladislav Houba“ nelze rozdělit na dvě věty, ale je nutno jej z pravidla vyloučit. Za tímto účelem byla vytvořena sada výjimek, která pokrývá ty nejčastější.

Jistě by bylo možné pro detekci hranic vět vytvořit sofistikovanější systém, pro naše využití při pozdějším hledání profilů slov však přesnost tohoto přístupu zcela postačuje.

3.6.2 Morfologické značkování a lemmatizace

Vzhledem k tomu, že v dalších částech práce budeme chtít z textu získávat relativně složité jazykové informace, je vhodné mít předem připravena alespoň ta nejzákladnější data o slovech. K nim beze sporu patří morfologická značka a lemma.

Morfologické značky jsou pro flektivní jazyky jako je čeština velmi důležité. U každého slova vyznačují např. pád, číslo, rod u substantiv nebo osobu u sloves. Lemmatem se rozumí základní tvar slova, přičemž o něm má smysl mluvit pouze u ohebných slovních druhů, což jsou substantiva, adjektiva, zájmena, číslovky a slovesa. U ostatních slovních druhů je v našem korpusu vždy lemma nahrazeno slovem samotným.

Základním tvarem je pro substantiva nominativ jednotného čísla, u adjektiv je to nominativ v singuláru mužského rodu prvního stupně, podobně pro zájmena a číslovky je volen nominativ singuláru mužského rodu a konečně u sloves je základním tvarem infinitiv. Tato přiřazení však nemusí být vždy jednoznačná, proto se jednotlivé implementace lemmatizátorů mohou částečně lišit.

K morfologickému značkování a lemmatizaci češtiny v současné době existují dva publikované přístupy, jež se z uživatelského hlediska liší nejvíce v reprezentaci morfologických informací. „Pražský" lemmatizátor s pozičním systémem uložení informací [15] a „brněnský" systém Ajka [38], uchovávající data v podobě hodnot atributů, který byl pro značkování našeho korpusu vybrán.

Ajka dokáže k většině českých slov najít morfologické značky a lemma. U velkého množství slov však tyto informace nejsou jednoznačné. Pro ilustraci se podívejme na morfologickou analýzu slova „ženu":

```
<s> ==žen=u= (580-hnát)
    <l>hnát
    <c>k5eAaImIp1nS
<s> =žen==u= (1768-žena)
    <l>žena
    <c>k1gFnSc4
```

Z výstupu analyzátoru je vidět, že může jít buď o sloveso *hnát* v první osobě jednotného čísla nebo o substantivum *žena* ve 4. pádu singuláru. Podrobný popis morfologických značek, které Ajka používá lze nalézt v dodatku A.

Pro zjednoznačnění morfologických informací slouží nástroje nazývané *taggery*. Většina z nich pracuje na základě statistických modelů (n-gramový model, feature-based model), paměťových modelů nebo pravidlových modelů. Zatímco pro angličtinu lze se značným úspěchem použít jen jeden z modelů, pro flektivní jazyky (češtinu) je třeba přístupy kombinovat.

Kombinací přístupů využívá i námi použitý tagger Desamb [45]. Jeho charakteristickým rysem je především schopnost učit se z předem neoznačkovaných dat. Užitím tohoto taggeru již dostaneme morfologické

informace v podobě, v jaké je lze použít při konstrukci vertikálního textu.

Aplikací všech předchozích procedur byl vytvořen označkovaný korpus o rozsahu cca 350 mil. pozic.

Kapitola 4

Model dokumentu

Před započetím procesu strojového učení je nutné převést trénovací data do vhodného formátu. Nejúspěšnější a nejpoužívanější reprezentací dokumentu, které se přidržíme i my, je vektorový model. Každá složka vektoru odpovídá jednomu slovu (případně tokenu), přičemž významnost daného slova v dokumentu může být reprezentována mnoha způsoby.

4.1 Vektorový model

Nejjednodušší možností reprezentace tokenu ve vektoru je binární reprezentace. Pro každý token z vektoru je zaznamenána jedna z hodnot 0 a 1, v závislosti na tom, zda se daný token v dokumentu vyskytuje. Jinou možností je zaznamenání četnosti výskytu. V tomto případě je zpravidla nutno hodnotu normalizovat délkou dokumentu. Podle publikace [44] bylo největší přesnosti klasifikace dosaženo použitím koeficientu TF-IDF (Term Frequency-Inverse Document Frequency). Další možností je vážení termů hodnotami χ^2 nebo Information gain [2]. Nyní tři nejčastěji používané přístupy formalizujme:

Nechť m je počet dokumentů v trénovacích datech, $f_d(t)$ frekvence termu t v dokumentu d pro $d \in \{1, 2, \ldots, m\}$ a $Terms$ množina termů $\{t_1, t_2, \ldots, t_n\}$.

Binární reprezentace
Dokument d je reprezentován jako vektor $(v_1, v_2, \ldots, v_n) \in \{0, 1\}^n$, kde

$$v_i = \begin{cases} 1 & \text{jestliže } f_d(t_i) > 0 \\ 0 & \text{jinak} \end{cases}$$

Frekvenční reprezentace (TF)
Dokument d je reprezentován jako vektor $(v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$, kde

$$v_i = \frac{f_d(t_i)}{m}$$

Reprezentace hodnotami TF-IDF (Term Frequency – Inverse Document Frequency) [35]
Nevýhodou předchozích dvou modelů je fakt, že mají všechny termy stejnou váhu. Přitom intuitivně slova konkrétnější mají při klasifikaci větší význam než slova obecná, která se vyskytují ve všech dokumentech. Tento nedostatek je eliminován koeficientem IDF, který je definován pro všechny $t_i \in Terms$ jako:

$$IDF(t_i) = log_2 \left(\frac{m}{|\{j : f_j(t_i) > 0\}|} \right)$$

Kombinací TF a IDF dostáváme:

$$v_i = \frac{f_d(t_i)}{m} \cdot log_2 \left(\frac{m}{|\{j : f_j(t_i) > 0\}|} \right)$$

Pro další zpracování vektorů metodami strojového učení je v případě TF a TF-IDF reprezentace výhodné reálné hodnoty vektoru diskretizovat. Pro diskretizaci byl použit algoritmus MDL [10], založený na minimalizaci informační entropie.

4.1.1 Vážení termů

Z předchozích definic je vidět, že popsané přístupy zacházejí se všemi výskyty termu uniformě. Každý výskyt termu má tedy v dokumentu vždy stejnou váhu. Tímto způsobem se však může ztratit důležitá informace o struktuře dokumentu.

Jak bylo popsáno v kapitole 2, ke každé pozici v korpusu je zaznamenána informace o strukturní jednotce, ve které se v originálním dokumentu nachází. Těchto příznaků je možné využít při vážení významnosti jednotlivých termů.
Nechť váha termu, který se vyskytuje v prostém textu je 1. Váha termů ze speciálních struktur byla experimentálně stanovena jako její násobek následovně:

- název dokumentu: 10x
- nadpisy všech úrovní: 5x
- hypertextový odkaz: 3x
- odrážka seznamu: 3x

Je zřejmé, že pro úspěch klasifikace je klíčová volba množiny $Terms$, proto se v dalších částech kapitoly budeme věnovat výběru jejích prvků.

4.2 Shlukování termů

Shlukování termů je prováděno na základě předem připraveného slovníku. V této fázi již předpokládáme, že jsou trénovací data připravena ve formě vertikálního textu včetně jednoznačného určení lemmatu.

Slovníkem v našem případě rozumíme totální funkci

$$s : Terms \rightarrow Rep,$$

která libovolnému termu z množiny $Terms$ přiřazuje právě jednoho reprezentanta z $Rep \subseteq Terms$. Tato funkce jednoznačně určuje rozklad množiny $Terms$ na shluky podobných výrazů ekvivalencí σ:

$$(a, b) \in \sigma \iff s(a) = s(b)$$

Opačně, pokud máme rozklad množiny $Terms$, lze vždy najít *nějakou* funkci s, která daný rozklad určuje.
Důkaz: z každého shluku (prvku rozkladu) $C \in Terms/\sigma$ stačí vybrat libovolného reprezentanta $r \in C$ a definovat

$$s(x) = r \quad \text{pro všechna} \quad x \in C.$$

Při tvorbě našeho slovníku budeme postupovat následovně:

1. Nalezení charakteristické množiny pro každý term $t \in Terms$.
2. Určení rozkladu množiny $Terms$ na základě charakteristických množin.
3. Určení funkce s.

4.2.1 Charakteristická množina termu

Základním nástrojem pro vytvoření charakteristických množin termů je Sketch Engine [19]. Jedná se o korpusový nástroj, který pro slova v korpusu umožňuje nalezení jejich častých gramatických kontextů a kolokací, tzv. slovních profilů. Příklad některých profilů slova *auto*, napočítaných na trénovacím korpusu, je na obrázku 4.1. Na obrázku jsou zachycena lemmata, která se často vyskytují v daných gramatických relacích. Tyto

a_modifier	8567	0.9
ojetý 317	1145	65.88
zaparkovaný 149 nákladní 372		
jezdící 92 projíždějící 107		
havarovaný 37 protijedoucí 34		
jedoucí 37		
osobní	1255	57.71
popelářský 60	131	50.73
kolemjedoucí 16 kradený 20		
ujíždějící 14 pancéřový 8		
parkující 7 dodávkový 6		
policejní	359	45.66
nastartovaný 25	50	42.06
kaskadérský 9 narážející 8		
srazící 8		

prec_do	2054	8.4
navigace	381	65.86
nasednout 78	141	53.71
nasedat 53 přesednout 10		
sednout	132	51.49
sedat 54	69	45.28
usednout 15		
sunout 35	54	44.29
naskákat 19		
vlézt	53	43.56
drcnout 17	47	41.66
vykráčet 12 vloupat 18		
naložit	38	38.41
nabíječka	25	36.93

prec_verb	2425	2.3
vlastnit	282	53.42
přibrzdit	57	52.72
zmoct	53	46.53
půjčit	51	38.15
půjčovat	41	38.14
parkovat 29	44	34.93
zaparkovat 15		
sbalit	33	34.76
umýt	27	34.7
koupit	113	34.41

is_obj4_of	1461	2.3
zaparkovat	74	54.7
demolovat 34	42	51.93
zdemolovat 8		
pronajmout	71	51.83
krást	63	47.04
zařídit	44	37.37
půjčovat	29	35.5
přiřítit	16	34.59
předjíždět	14	30.75
ukradnout	21	30.5

Obrázek 4.1: Profil slova *auto*

relace jsou definovány sadou syntaktických pravidel. Typy gramatických relací jsou na obrázku zapsány ve zvýrazněných pruzích a mají následující význam:

- **a_modifier** – modifikátor
- **prec_do** – slovo připojené předložkou *do*
- **prec_verb** – připojené sloveso
- **is_obj4_of** – je předmětem ve čtvrtém pádě

Příklad pravidla v CQS (Corpus Query System) pro zachycení relace **is_obj4_of** vypadá následovně:

```
*DUAL
=is_obj4_of/has_obj4
  2:[k="k5" & m="m[^AN]" & p="p3" & !(lemma="být" |
    lemma="mít.*" | lemma="chtít" | lemma="muset" |
    lemma = "moci.*" | lemma = "smět") ]
    [k!="k[15I837]"]{0,5} 1:[k="k1" & c="c4"]
  2:[k="k5" & m="m[AN]" & !(lemma="být" | lemma=
    "mít.*" | lemma="chtít" | lemma = "muset" |
    lemma = "moci.*" | lemma = "smět") ]
    [k!="k[15I837]"]{0,5} 1:[k="k1" & c="c4"]
```

Všech gramatických relací, které jsou použity, je mnohem více a lze je nalézt na přiloženém DVD.

Profily slov jsou velmi dobrou charakteristikou jednotlivých slov z korpusu. Nabízí se možnost hledat slova s podobnou charakterizací a vytvořit tak thesaurus. Sketch Engine vytvoření thesauru na základě podobných profilů slov umožňuje. Ke každému lemmatu l, jež má v korpusu frekvenci, umožňující úspěšnou aplikaci statistických metod, lze nalézt seznam podobných slov $SP_l = [w_1, w_2, \ldots, w_n]$, seřazený sestupně podle indexů podobnosti $i_1 \ldots i_n$ s lemmatem l [18].

V této práci byl pro každé lemma l z korpusu vytvořen *charakteristický seznam* $CHS(l)$ následujícím principem:

- Jestliže je četnost lemmatu l v korpusu menší než 100:
  $CHS(l) = [l]$
- Jinak:
  $CHS(l) = [w_1, w_2, \ldots, w_k] : \forall i_j \in \{i_1, i_2, \ldots i_k\} : i_j \geq 0.1$

V tabulce 4.1 je zachycen charakteristický seznam napočítaný pro lemma *auto*, včetně indexů podobnosti.
Z tabulky je vidět, že je skutečně většina slov ze seznamu podobná slovu *auto* – jedná se o dopravní prostředky. Jedinou výjimkou je lemma *aut*. Pravděpodobným důvodem výskytu tohoto slova v seznamu je chybné určení 2. pádu substantiva *auto* v plurálu jako 1. pád slova *aut* v singuláru u nezanedbatelného množství výskytů v korpusu. Takováto slova jsou samozřejmě v seznamu nežádoucí a je třeba je odstranit.

auto	1
automobil	0.184
autobus	0.171
vůz	0.166
vozidlo	0.153
vlak	0.141
aut	0.133
tramvaj	0.126
loď	0.124
letadlo	0.112
trolejbus	0.11

Tabulka 4.1: Charakteristický seznam lemmatu *auto*

Pro jejich eliminaci byla v práci navržena metoda, která z charakteristického seznamu odstraňuje slova vyskytující se častěji v jiných kontextech. Tímto mechanismem jsou navíc ze seznamu odstraněna homonyma, jež se v korpusu vyskytují častěji v jiném významu a mohla by tak později zkreslovat výsledky. Z takto vyčištěného seznamu již je vytvořena *charakteristická množina*. Formálně:

Nechť $CHS(l) = [w_1, w_2, \ldots, w_k]$ je charakteristický seznam lemmatu l. Označme $S(l) = \{w_1, w_2 \ldots, w_k\}$ a $S_p(l) = \{w_i | i \leq k/p\}$ množinu nejpodobnějších slov, kde $p \in \mathbb{R}^+$ je vhodně zvolená konstanta.
Charakteristickou množinu $CH(l)$ lemmatu l definujeme jako

$$CH(l) = \{w_i : q \cdot |S(w_i) \cap S_p(l)| \geq |S_p(l)|\}$$

kde $q \in \mathbb{R}^+$ je zvolená opět vhodně zvolená konstanta. Při experimentech bylo dosaženo nejlepších výsledků s hodnotami konstant $p = 2, q = 2$.

4.2.2 Vytvoření slovníku

Dalším úkolem při tvorbě slovníku je nalezení shluků. Intuitivně do jednoho shluku patří termy, které mají podobné charakteristické množiny. Podobnost množin lze určit na základě různých metrik [12]. V této práci byla použita Jaccardova míra, podle níž je vzdálenost dvou termů a, b definována jako

$$j(a, b) = \frac{|CH(a) \cap CH(b)|}{|CH(a) \cup CH(b)|}$$

Vytváření shluků pracuje na principu algoritmu hierarchického shlukování [3] metodou „zdola nahoru". Začíná se v situaci, kdy každý term tvoří jeden samostatný shluk. Postupně spojujeme dva nejbližší shluky, dokud minimální vzdálenost nestoupne na předem stanovenou hodnotu v. Schematický algoritmus vypadá následujícím způsobem:

```
INICIALIZACE:
1. urči vzájemné vzdálenosti mezi všemi termy
2. zařaď každý term do samostatného shluku

HLAVNÍ CYKLUS:
dokud je vzdálenost nejbližších dvou shluků < v:
    1. najdi dva nejbližší shluky a spoj je
    2. charakteristická množina nového shluku vznikne
       sjednocením původních
    3. pro tento nový shluk spočítej vzdálenosti
       od ostatních shluků
```

Problémem tohoto algoritmu je jeho velká časová a prostorová složitost. Úzkým hrdlem je především počítání vzdáleností mezi shluky, neboť v případě našich dat je počet dvojic všech shluků na začátku algoritmu větší než 10^{10}.

Lze snadno nahlédnout, že matice vzdáleností je velmi řídká. Jako řešení problému se tedy nabízí počítání vzdáleností jen mezi těmi shluky A, B, u kterých $CH(A) \cap CH(B) \neq \emptyset$. Nalezení takových dvojic by však mohlo znamenat procházení celého prostoru všech možností. Proto je využito další skutečnosti. Jestliže jsou si lemmata a, b dostatečně podobná, pak $b \in CH(a), a \in CH(b)$. Na základě těchto pozorování definujeme množinu dvojic lemmat (případně shluků), pro které je počítána vzdálenost jako

$$V = \{(a, b) \in Terms \times Terms : a \in CH(b) \vee b \in CH(a)\}$$

Modifikovaný algoritmus vypadá následovně:

```
INICIALIZACE:
1. urči vzájemné vzdálenosti mezi všemi dvojicemi
   z množiny V
2. seřaď všechny dvojice do seznamu vzestupně
   podle jejich vzdálenosti
3. zařaď každý příklad do samostatného shluku
```

```
HLAVNÍ CYKLUS:
vyber první dvojici shluků ze seznamu
jestliže je jejich vzdálenost < v:
    1. spoj tyto dva shluky v jeden
    2. charakteristická množina nového shluku vznikne
       sjednocením původních
    3. pro tento nový shluk S spočítej vzdálenosti
       od ostatních shluků, které obsahují alespoň
       jeden prvek z CH(S)
    4. přeuspořádej seznam dvojic shluků, aby zůstal
       ve vzestupném pořadí
```

Maximální vzdálenost pro shlukování byla v práci experimentálně stanovena na 0,45. Z algoritmu je snadno vidět, že konstruuje rozklad na množiny $Terms$, který je určen ekvivalencí

$$a\sigma b \Longleftrightarrow a, b \quad \text{patří po provedení algoritmu do stejného shluku}$$

Nechť $freq(x)$ je frekvence termu x v korpusu. Definujme slovníkovou funkci s: $\forall S \in Terms/\sigma, \forall a \in S : s(a) = b$ takové, že $b \in S, freq(b) = max\{freq(x)|x \in S\}$. V případě více termů s maximální frekvencí vyber první v lexikografickém uspořádání.

Shlukování podle slovníku bylo provedeno tak, že všechna lemmata $l \in DOM(s)$ byla nahrazena lemmaty $s(l)$.

4.3 Výběr atributů

I po aplikaci slovníkové funkce na seznam všech lemmat z korpusu je stále ještě počet různých termů pro vytváření vektorových modelů dokumentů příliš velký ($> 10^6$).

Problém velkého počtu atributů při strojovém učení je nejen ve velké výpočetní náročnosti, ale i ve snížení přesnosti. I algoritmy, které jsou uzpůsobené pro práci s velkým počtem atributů ztrácejí na přesnosti při použití desetitisíců a více atributů.

Otázkou je, jaké atributy by se pro klasifikaci měly vybrat. Intuitivně to jsou ty, které nejlépe rozlišují jednotlivé třídy.

4.3.1 Stop list

Pokud se podíváme na seznam termů v korpusu, seřazený podle četnosti, uvidíme, že nejfrekventovanějšími termy jsou spojky, předložky, interpunkční znaménka, způsobová slovesa a další výrazy, které v našem případě obsahují minimální informaci.

Na druhém konci stojí naopak tzv. *hapax legomena*, tedy slova, která se v korpusu vyskytují právě jednou. Tato slova mají také pro klasifikaci minimální význam. Vzhledem k velikosti korpusu (350 milionů pozic) se dokonce můžeme omezit jen na slova, která mají četnost větší než 20. Po odstranění nejfrekventovanějších a nejméně četných výrazů se seznam slov zmenšil o 60% na cca. 480000.

Vzhledem k tomu, že máme k dispozici morfologické informace a lemmata, můžeme seznam dále zmenšovat. Ukazuje se, že pro klasifikaci mají význam pouze některé slovní druhy. Jsou to substantiva, adjektiva, slovesa, adverbia a zkratky. Ostatní slovní druhy je možno vynechat. Termy, u kterých morfologický analyzátor nedokáže rozpoznat slovní druh je lépe ponechat, neboť mohou nést důležitou informaci.

Po nahrazení slov jejich základním tvarem a odstranění nevyjmenovaných slovních druhů se seznam dále zmenšil na 290000 slov.

Z tohoto množství již je třeba vybrat vhodné atributy sofistikovanějšími metodami. Algoritmus výběru pracuje následujícím způsobem:

```
1. Pro každý term ze seznamu spočítej jeho ohodnocení
   na základě použité míry.
2. Seřaď seznam atributů sestupně podle ohodnocení.
3. Se seznamu vyber prvních n atributů.
```

Zaměřme se na 2 nejpoužívanější míry pro výběr atributů podle [6].

4.3.2 χ^2 test

Jedná se o statistickou metodu, která je známá z řady aplikací. V našem případě je užita pro určení významnosti termu vzhledem k dané třídě. Pro jednoduchost uvažujme binární model dokumentu, označme t ohodnocovaný term a $Class$ množinu všech tříd. Nechť:

- $k_{i,0}$ = počet dokumentů ve třídě i, které neobsahují term t
- $k_{i,1}$ = počet dokumentů ve třídě i, které obsahují term t

Dostáváme tak kontingenční tabulku:

I_t\C	1	2	...	11
0	$k_{1,0}$	$k_{2,0}$	...	$k_{11,0}$
1	$k_{1,1}$	$k_{2,1}$	...	$k_{11,1}$

kde C a I_t jsou náhodné proměnné a $k_{l,m}$ označuje počet pozorování, u kterých $C = l$ a $I_t = m$. Naším cílem je zjistit, do jaké míry jsou tyto náhodné proměnné nezávislé. Nechť

$$n = \sum_{i \in Class} (k_{i,0} + k_{i,1})$$

Marginální pravděpodobnosti jsou

- $P(C = i) = (k_{i,0} + k_{i,1})/n$
- $P(I_t = 0) = \sum_{i \in Class} k_{i,0}/n$
- $P(I_t = 1) = \sum_{i \in Class} k_{i,1}/n$

Pokud by C a I_t byly nezávislé, platilo by $P(C = 1, I_t = 0) = P(C = 1) \cdot P(I_t = 0)$. V našem pozorování platí $P(C = 1, I_t = 0) = k_{1,0}/n$. Podobně pro všechny ostatní prvky kontingenční tabulky. Očekávaná hodnota na pozici (l, m) je tedy $n \cdot P(C = l)P(I_t = m)$ a pozorovaná hodnota $k_{l,m}$.
Rozdíl mezi očekávanou a pozorovanou hodnotou je interpretován následovně:

$$\chi^2 = \sum_{l \in Class} \sum_{m \in \{0,1\}} \frac{(k_{l,m} - n \cdot P(C = l)P(I_t = m))^2}{n \cdot P(C = l)P(I_t = m)}$$

4.3.3 Vzájemná informace (MI-Score)

Tato míra vychází z teorie informace a je vhodná i pro multinomiální modely dokumentu. Pokud X a Y jsou diskrétní náhodné proměnné nabývající hodnot x pro X a y pro Y, je jejich vzájemná informace definovaná jako:

$$MI(X, Y) = \sum_x \sum_y P(x, y) log \frac{P(x, y)}{P(x)P(y)}$$

kde $P(x) = P(X = x)$ a $P(y) = P(Y = y)$.
Podobně jako χ^2, určuje MI-Score závislosti náhodných proměnných X a Y, tedy jak se liší hodnoty $P(x, y)$ a $P(x) \cdot P(y)$. V případě, že jsou na sobě

náhodné veličiny nezávislé, pak $P(x,y)/P(x)\cdot P(y) = 1$ a tedy $MI(X,Y) = 0$. Naopak, čím více jsou náhodné veličiny závislé, tím větší hodnota MI-Score.
Nyní se vraťme k ohodnocování termů z korpusu. Nechť je t opět fixní term, I_t náhodná proměnná nabývající hodnot $0, 1$ pro zvolený dokument a C náhodná proměnná nabývající hodnot z množiny $Class$. Hodnota MI-Score pro term t je definována jako:

$$MI(I_t, C) = \sum_{l \in Class} \sum_{m \in \{0,1\}} \frac{k_{l,m}}{n} log \frac{k_{l,m}/n}{(k_{l,0} + k_{l,1}) \cdot (\sum_{i \in Class} k_{i,m})/n^2}$$

Kapitola 5

Evaluace výsledků klasifikace

V předchozích částech práce byla představena řada metod strojového učení a předzpracování dat. Cílem této kapitoly je otestovat všechny přístupy na reálných datech a vybrat ty nejlepší.

5.1 Definice pojmů

Pro účely testování definujme několik zásadních pojmů. Pro zjednodušení uvažujme pouze dvě klasifikační třídy – pozitivní a negativní.

Matice záměn (confusion matrix)
Cílem testování je většinou učit v kolika případech se klasifikátor shoduje s učitelem. Pokud se systém s učitelem neshoduje, může být pro nás zajímavá informace o třídě, přiřazené systémem. Tyto údaje je zvykem zobrazovat v *matici záměn*. Tabulka 5.1 zachycuje situaci, kdy jde o klasifikaci do dvou tříd „+“ a „-“. TP (true positive) je počet příkladů, které systém správně zařadil do třídy „+“, FP (false positive) počet chybně zařazených příkladů do třídy „+“, TN (true negative) a FN (false negative) analogicky.

	Klasifikace systémem	
Správné zařazení	+	-
+	TP	FN
-	FP	TN

Tabulka 5.1: Matice záměn

Celková správnost (overall accuracy)
Jedná se nejjednodušší charakteristiku, která určuje pouze relativní počet správných určení systému v případě, že je každý příklad klasifikován do jedné ze tříd. Z matice záměn ji lze vypočítat jako:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Celkovou chybu lze vypočítat jako:

$$Err = 1 - Acc = \frac{FP + FN}{TP + TN + FP + FN}$$

Přesnost, úplnost a F-míra
Někdy nemusí být informace o úspěšnosti dostatečná, proto se definují další ukazatele, zohledňující pokrytí příkladů – přesnost (precision), úplnost (recall) a jejich kombinace F-míra:

$$\text{Přesnost} = \frac{TP}{TP + FP}$$

$$\text{Úplnost} = \frac{TP}{TP + FN}$$

$$\text{F-míra} = \frac{2 \cdot \text{Přesnost} \cdot \text{Úplnost}}{\text{Přesnost} + \text{Úplnost}} = \frac{2TP}{2TP + FP + FN}$$

Závislost celkové správnosti na počtu atributů
Pro učení je jedním z rozhodujících parametrů počet atributů. Závislost celkové správnosti na počtu atributů je možné zachytit ve formě grafu. Atributy jsou vybírány jako prvních n prvků seznamu atributů, seřazeného sestupně podle hodnot χ_2 nebo MI-score.

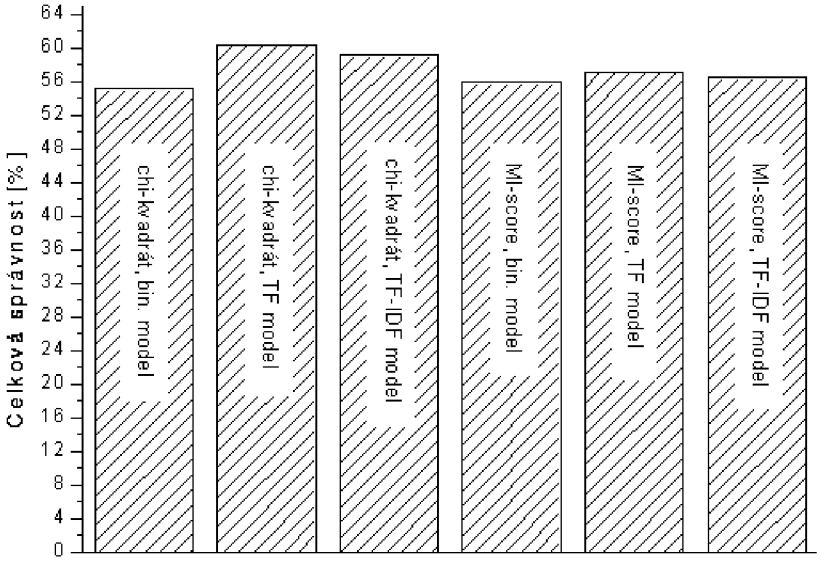
Z grafu závislosti je vidět, že s narůstajícím počtem atributů až do jisté hranice přesnost stoupá, potom již jen stagnuje nebo dokonce klesá.

5.2 Výsledky experimentů

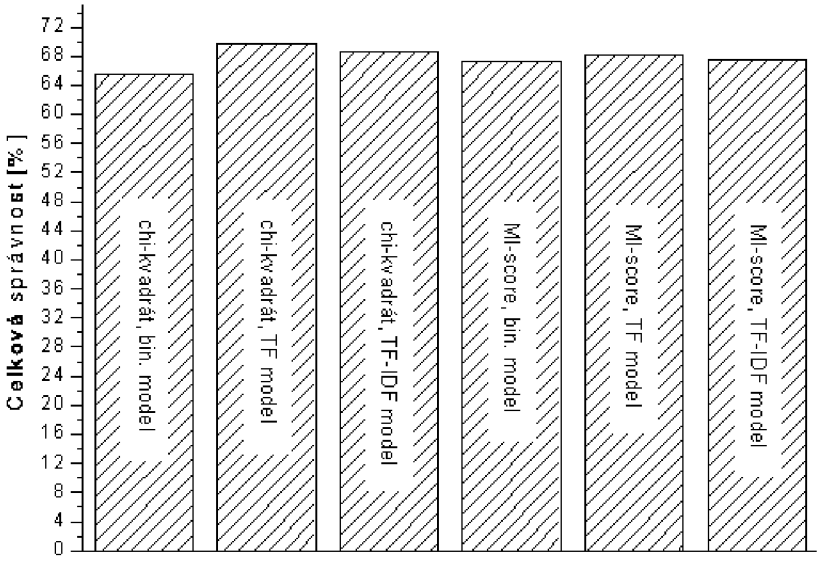
Pro účely testování bylo z korpusu vybráno náhodně 3500 trénovacích příkladů, 1500 testovacích a byla použita 10ti složková křížová validace [22]. K testování metod Naivního Bayesovského klasifikátoru, algoritmu C4.5 a k-nejbližších sousedů byl použit balík nástrojů *Weka* [47], pro testování metod Support Vector Machines nástroj *LIBSVM* [7].

Nejprve jsou porovnány běžně používané metody předzpracování a všechny testované algoritmy na datech, která tvoří příklady, sestávající

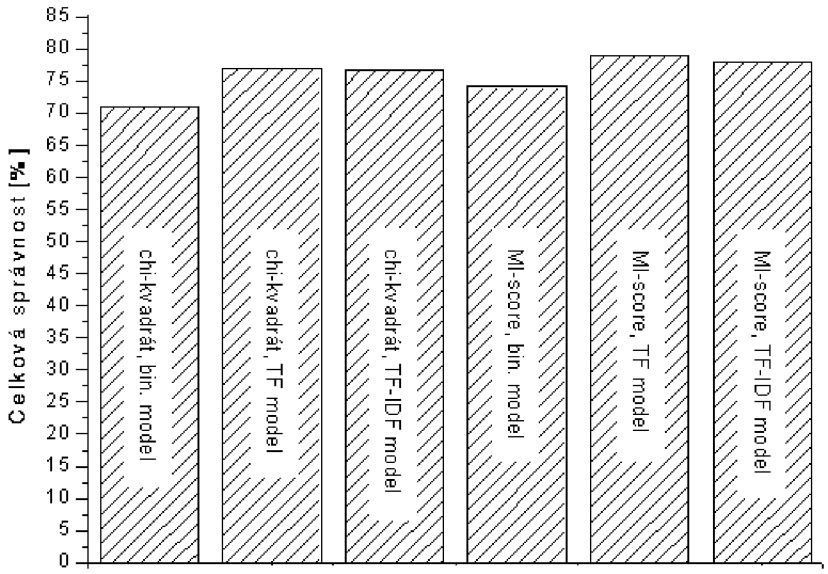
pouze z jednotlivých webových stránek, potom je vybrána nejlepší varianta a na ní otestovány obě nové metody, navržené v této práci (rozšíření příkladu o odkazované dokumenty a shlukování termů na základě slovníku).



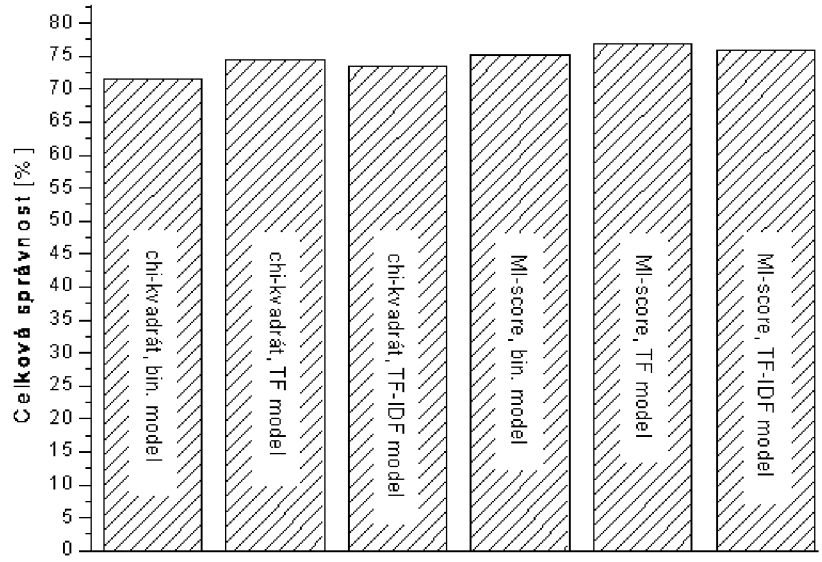
Obrázek 5.1: Naivní Bayesovský klasifikátor



Obrázek 5.2: Algoritmus C4.5



Obrázek 5.3: Support Vector Machines



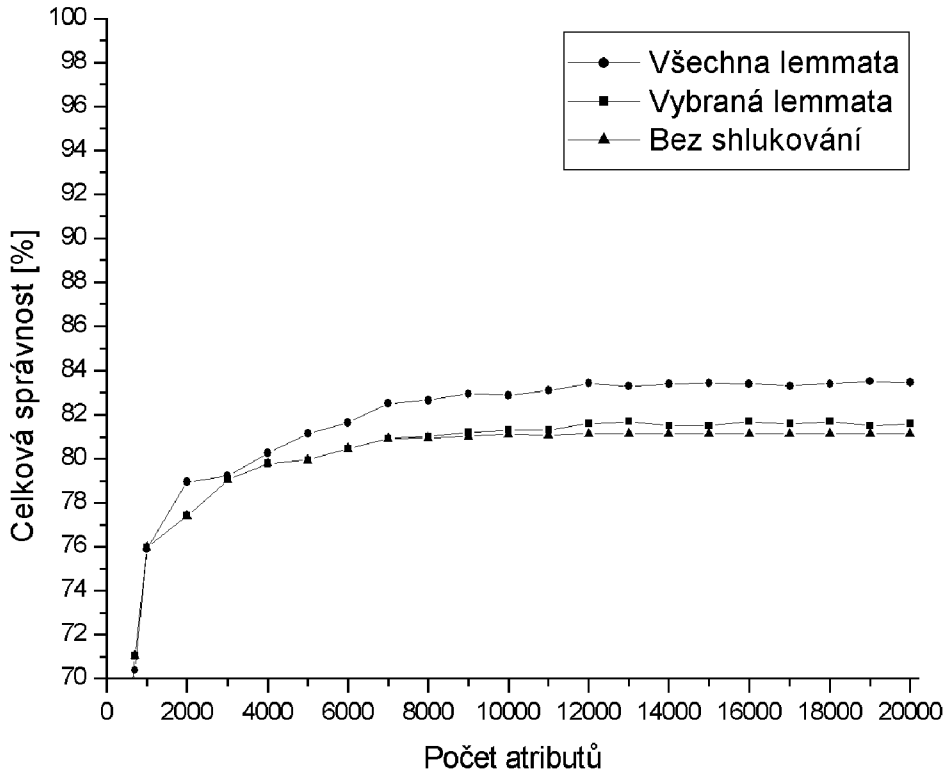
Obrázek 5.4: *k*-nejbližších sousedů

Na obrázcích 5.1-5.4 jsou znázorněny grafy celkových správností pro algoritmy naivní Bayesovský klasifikátor, algoritmus C4.5, Support Vector Machines a *k*-nejbližších sousedů se všemi kombinacemi reprezentací vek-

torových modelů a výběru atributů, popsaných v předchozí kapitole. Počet atributů byl zvolen na 3000.

Z grafů je vidět, že nejlépe dopadl algoritmus Support Vector Machines s reprezentací vektorového modelu pomocí TF a výběrem atributů mírou MI-score. Byla tak dosažena správnost 79.04% Pro další experimenty byly zvoleny tyto parametry jako výchozí.

Na obrázku 5.5 jsou zachyceny grafy závislostí celkové správnosti na počtu atributů pro metodu bez shlukování, se hlukováním na základě stejných lemmat a se shlukováním na základě stejných lemmat po odstranění vybraných slovních druhů.



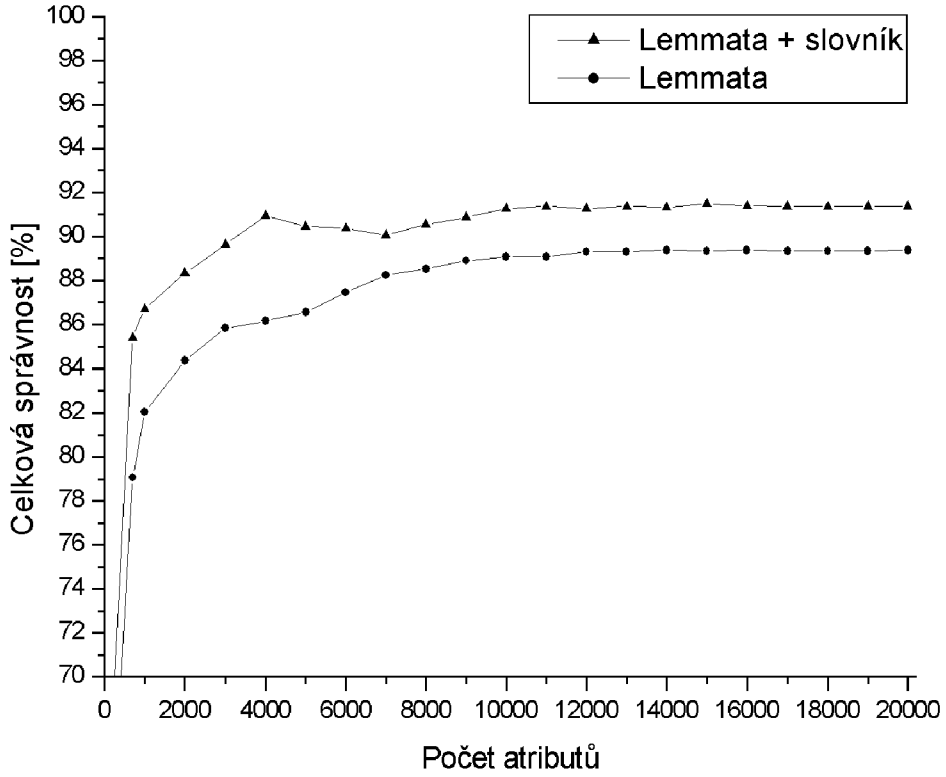
Obrázek 5.5: Metody shlukování

Je vidět, že ve všech případech roste úspěšnost až do počtu atributů okolo 12000. Poté se již výrazně nemění. Ze tří zobrazených metod podle očekávání nejhůře dopadl přístup bez shlukování a nejlépe metoda shlu-

kování podle stejných lemmat s úspěšností 83,4%. Metoda, využívající pouze atributů vybraných slovních druhů zklamala a dopadla hůře.

Nyní již se konečně dostáváme k výsledkům metod, navržených v této práci. Za výchozí parametry byly zvoleny ty nejlepší z předchozích experimentů – tzn. algoritmus Support Vector Machines, TF model dokumetu, selekce atributů pomocí MI-score a shlukování na základě stejných lemmat.

Obrázek 5.6 zachycuje křivky učení pro metodu rozšiřující dokumenty o odkazované stránky se shlukováním podle slovníku a se shlukováním pouze podle lemmat. Z grafu je vidět poměrně velký nárůst úspěšnosti, především po rozšíření dokumentu o odkazované stránky (z 83,4% na 89,3%). Shlukování podle slovníku poté zvýšilo přesnost na 91,6%. Vzhledem k tomu, že již úspěšnost 83,4 je velmi dobrá, lze považovat i zlepšení o několik procent za úspěch.



Obrázek 5.6: Rozšíření o odkazované dokumenty

V tabulce 5.2 je zachycena matice záměn výsledného klasifikátoru, v tabulce 5.3 přesnost, úplnost a F-míra. Je vidět, že pro většinu kategorií funguje klasifikace velmi dobře. Problém je především u kategorií, které se překrývají.

Kategorie	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**
a – Cestování	***102***	*0*	*0*	*1*	*2*	*1*	*0*	*1*	*0*	*0*	*0*
b – Erotika	*1*	***103***	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*1*	*0*
c – Hry	*1*	*1*	***96***	*0*	*1*	*0*	*0*	*2*	*0*	*1*	*0*
d – Informační a inzertní servery	*1*	*0*	*0*	***93***	*0*	*2*	*0*	*0*	*1*	*3*	*3*
e – Kultura a umění	*1*	*1*	*0*	*2*	***90***	*1*	*2*	*0*	*1*	*5*	*1*
f – Lidé a společnost	*3*	*5*	*0*	*0*	*5*	***93***	*0*	*1*	*0*	*7*	*0*
g – Počítače a internet	*2*	*0*	*0*	*3*	*0*	*1*	***97***	*0*	*0*	*1*	*0*
h – Sport	*1*	*0*	*2*	*0*	*2*	*1*	*0*	***94***	*0*	*0*	*2*
i – Věda a technika	*0*	*1*	*0*	*0*	*0*	*3*	*0*	*1*	***96***	*1*	*1*
j – Volný čas a zábava	*1*	*2*	*1*	*3*	*0*	*0*	*4*	*0*	*1*	***82***	*1*
k – Zpravodajství	*1*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	***99***

Tabulka 5.2: Matice záměn

Kategorie	**Přesnost**	**Úplnost**	**F-míra**
Cestování	*0.957*	*0.894*	*0.924*
Erotika	*0.971*	*0.911*	*0.94*
Hry	*0.941*	*0.969*	*0.954*
Informační a inzertní servery	*0.902*	*0.911*	*0.906*
Kultura a umění	*0.865*	*0.891*	*0.877*
Lidé a společnost	*0.815*	*0.911*	*0.887*
Počítače a internet	*0.932*	*0.932*	*0.932*
Sport	*0.97*	*0.949*	*0.959*
Věda a technika	*0.932*	*0.969*	*0.95*
Volný čas a zábava	*0.872*	*0.811*	*0.84*
Zpravodajství	*0.979*	*0.925*	*0.951*

Tabulka 5.3: Přesnost, úplnost, F-míra

Kapitola 6

Určení klíčových slov

Pod pojmem *klíčová slova* slova dokumentu si lze představit slova, která jsou pro daný dokument důležitá. Pro automatické určování klíčových slov však potřebujeme přesnější definici.

Jednu z definic poskytuje např. [27]: „Keyword is a word which occurs in a text more often than we would expect to occur by chance alone". Jedná se tedy o slova, jejichž četnost je v dokumentu statisticky významnější než v nějakém univerzu dokumentů.

Klíčovými slovy se však nemusí rozumět jen jednotlivé termy. Princip klíčových slov lze např. zobecnit na formule v predikátové logice prvního řádu [32]. V citované práci bylo použito metod induktivního logického programování (ILP), pomocí nichž byly z textů extrahovány logické formule v predikátové logice prvního řádu, splněné v sadě pozitivních příkladů a nesplněné v sadě negativních příkladů.

6.1 Klíčová slova a klíčovost

V minulosti se pojmem klíčovosti zabývala řada lingvistů. Nejznámějším přístupem je tzv. *propoziční analýza* [20]. Metoda rozděluje text do propozičních komponent (propozic), přičemž nezáleží na tom, jak jsou jednotlivé propozice vyjádřeny. Jedna propozice může být např. zachycena dvěma různými jazyky v případě, že je označen stejný koncept. Za klíčové propozice jsou potom považovány ty, které odkazují k většině ostatních propozic v textu.

Na propoziční analýzu svým přístupem navazuje Michael Hoey [14]. Jeho metoda nepracuje s propozicemi, nýbrž s celými větami. Podobně jako u propoziční analýzy autor hledá vztahy mezi větami. Klíčovou větou je pro něho ta, která se s dostatečnou četností opakuje v předchozím textu. Opakování však nemusí být doslovné, ale může se jednat např. jen o opakovaní stejné gramatické struktury.

Při určování klíčových slov je důležité rozlišit mezi jazykovou, myš-

lenkovou, kulturní a textovou klíčovostí [27]. Například slovo „jest" může být klíčové v současném textu, ale jen stěží v českém textu 19. století. Dále se v práci budeme zabývat pouze textovou klíčovostí, která k jazykovým, myšlenkovým a kulturním hlediskům nepřihlíží.

6.2 Automatické určení klíčových slov

Většina automatických metod určení klíčových slov je založena na opakování termů. Pokud se term vyskytuje významně relativně častěji ve zkoumaném dokumentu než v referenčním kospusu, je považován za klíčové slovo [27]. Referenční korpus je v takovém případě tvořen pokud možno velkým množstvím dokumentů v témže jazyce a stylu.

Pokud bychom tento princip chtěli použít i při určování klíčových slov webové stránky, narážíme opět na problém malého rozsahu. V typické stránce se jedno slovo opakuje v průměru 1.9krát [46], četnosti výskytu tedy většinou není možno využít.

6.2.1 Charakteristická slova

Určení klíčových slov v této práci je řešeno podobně jako v [29], jako navazující úloha na určení domény stránky. Pro každou doménu bylo nasbíráno velké množství dat, jež bylo použito k nalezení charakteristických slov pro danou doménu. Podobně jako při výběru atributů v kapitole 4 bylo použito statistických metod, konkrétně míry MI-Score. Ta umožnila najít pro každou kategorii termy, které se v ní vyskytují významně častěji než v ostatních kategoriích.

Opět nechť t je fixní term, I_t náhodná proměnná nabývající hodnot 0, 1 pro zvolený dokument, C náhodná proměnná nabývající hodnot z množiny $Class$, $c \in Class$ jedna z klasifikačních tříd a $l \in Class, m \in \{0,1\}$. Hodnotu MI-score pro term t a třídu c definujeme jako:

$$MI(I_t, C, c) = \sum_l \sum_m \frac{k_{l,m}}{n} log \frac{k_{l,m}/n}{(k_{l,0} + k_{l,1}) \cdot k_{c,m}/n^2}$$

Dále nechť $S_c = [l_1, l_2, \ldots, l_{max}]$ je seznam všech lemmat korpusu z třídy c, seřazený sestupně podle hodnot MI-score. Množinu charakteristických slov pro doménu c definujeme jako

$$CH_c = \{l_1, l_2, \cdots, l_n\}$$

kde $n \in \mathbb{Z}^+, n \leq max$ je vhodně zvolená konstanta.

6.2.2 Algoritmus výběru klíčových slov

Nejprve je třeba, aby algoritmus z dokumentu vybral významná slova, která se mohou stát potencionálními klíčovými slovy. Při jejich určování již do jednoho dokumentu nejsou zahrnuty odkazované stránky, neboť mohou obsahovat klíčová slova, která nejsou pro dotazovanou stránku relevantní. Z tohoto důvodu je rozsah většiny stránek velmi malý a není účelné počítat relativní četnosti výskytů jednotlivých termů, nebo jiné charakteristiky na nich založené.
Potencionální klíčová slova definujeme jako termy, pro které platí jedna z následujících podmínek:

1. Vyskytuje se v nadpisu libovolné úrovně.

2. Četnost v dokumentu je alespoň 2.

Při výběru klíčových slov z množiny potencionálních klíčových slov se berou v úvahu pouze ta, která se vyskytují v množině charakteristických slov pro doménu stránky. Vzhledem k tomu, že některá stránka může být výrazně delší, než je ze statistického hlediska obvyklé, může nastat situace, kdy je seznam klíčových slov příliš dlouhý. V takovém případě je vhodné jej zkrátit.

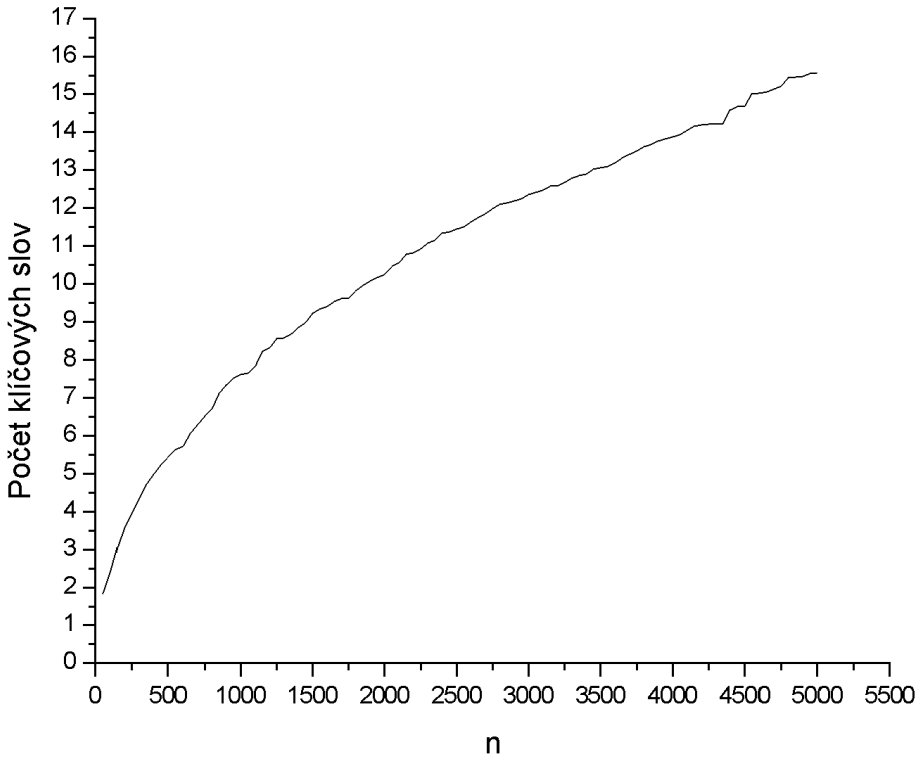
Podle [27] by měl být obvykle počet klíčových slov v rozsahu mezi 3 až 10 výrazy. Proto je seznam klíčových slov uspořádán sestupně podle hodnot MI-score a v případě nutnosti zkrácen na požadovanou délku. Ta je stanovena jako 1,5 násobek maximální doporučené délky, tedy na patnáct termů. Algoritmus pracuje schematicky následujícím způsobem:

```
VSTUP: webová stránka

1. urči doménu stránky
2. ze stránky vyber potencionální klíčová slova
3. z potencionálních klíčových slov vyber ta,
   která jsou charakteristická pro danou doménu
4. vybraná slova uspořádej do seznamu sestupně
   podle hodnot MI-score
5. seznam ořízni na maximální délku 15 slov

VÝSTUP: uspořádaný seznam klíčových slov
```

6.2.3 Volba velikosti množin charakteristických slov

Při budování množin charakteristických slov pro jednotlivé kategorie narážíme na problém volby velikosti množin, tedy volbu vhodné konstanty n. V případě malých hodnot n budou seznamy klíčových slov stránek příliš krátké nebo dokonce nebude možné žádné klíčové slovo najít. V případě velkých hodnot n se naopak v seznamu budou vyskytovat slova, která pro stránku klíčová nejsou.

Volba vhodné hodnoty n se zdá být obtížná. Jednou z možností, jak určit její přibližnou hodnotu je sledování průměrného počtu klíčových slov na stránku. Na obrázku 6.1 je graf závislosti průměrného počtu klíčových slov dokumentu na hodnotě n.



Obrázek 6.1: Závislost průměrného počtu klíčových slov na n

Pokud se budeme řídit doporučením 3-10 klíčových slov na stránku, omezí se nám rozsah hodnot n na přibližně 200-2000. Jen podle průměrného počtu klíčových slov by byla volba vhodného n obtížná, proto byl proveden

následující experiment:

1. Z korpusu bylo náhodně vybráno 30 dokumentů.
2. U každého dokumentu napočítán seznam klíčových slov pro hodnoty $n \in \{200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000\}$.
3. Pro každý dokument a hodnotu n ručně ohodnocena relevance klíčových slov.

Cílem bylo najít takovou hodnotu n, při které jsou klíčová slova stále ještě relevantní a přitom žádné z důležitých klíčových slov stránky v seznamu nechybí. Experiment ukázal, že maximálního počtu klíčových slov při zachování relevance bylo v průměru dosaženo při hodnotě $n = 800$. Velikost množin charakteristických slov pro všechny domény byla tedy nastavena na tuto experimentálně stanovenou hodnotu.

Kapitola 7

Závěr

Cílem této práce bylo navrhnout a implementovat systém, který pro libovolnou českou webovou stránku určí její žánrovou doménu a případně klíčová slova. Byly zde představeny dva nové přístupy, řešící problém malého rozsahu stránek, které umožnily klasifikaci s přesností 91%. Určení klíčových slov bylo řešeno jako navazující úloha, využívající znalosti domény.

Pro klasifikaci bylo použito několik algoritmů strojového učení, z nichž nejlépe dopadl algoritmus Support Vector Machines s lineární kernel funkcí, následovaný metodou k-nejbližších sousedů. Pro vektorovou reprezentaci slov byl použit frekvenční model, který v závěsu s modelem TF-IDF výrazně překonal binární model. Z velkého množství slov, vyskytujících se v trénovacích datech, bylo třeba vybrat jejich podmnožinu, tvořící složky vektorů. Byly otestovány míry χ^2 a MI-score, ze kterých lépe dopadla míra MI-score. Pro určení počtu atributů byl proveden experiment s hodnotami od 50 do 20000. Ukázalo se, že při hodnotách vyšších než 12000 již ke zlepšení přesnosti nedochází.

Jako jedno z řešení malého rozsahu webových stránek byla představena metoda rozšiřující dokumenty o některé, z nich odkazované, stránky. Díky tomuto rozšíření přesnost klasifikace výrazně vzrostla.

V práci byly použity dvě úrovně shlukování slov – klasický přístup shlukování na základě stejných lemmat a v práci navržená metoda shlukování založená na slovníku, vytvořeném pomocí statistických korpusových nástrojů. Obě metody dokázaly zvýšit přesnost o několik procent. Především u shlukování podle slovníku lze považovat i malé zlepšení za velký úspěch, neboť již přesnost před shlukováním byla velmi dobrá.

Součástí katalogu stránek není jen jedenáct použitých kategorií, ale je dále hierarchicky členěn do podkategorií. Vzhledem k vysoké úspěšnosti klasifikace do top-level kategorií by mohla být provedena další klasifikace v rámci jednotlivých tříd. Problém by však mohl nastat například u kategorie Sport, která má stovky podkategorií.

Funkční klasifikační systém včetně modulu pro určení klíčových slov je umístěn na přiloženém DVD. Jeho nevýhodou je závislost na morfologickém analyzátoru Ajka a desambiguátoru Desamb, které nejsou volně šiřitelné. V budoucnosti by mohlo být vhodné nahradit je nějakými Open Source nástroji (v případě, že budou existovat).

Druhou nevýhodou je malá rychlost systému. Prodleva je způsobená jednak stahováním dat z internetu a dále rychlostí stávajícího desambiguátoru. V případě nasazení systému v reálné aplikaci by bylo možné snížit dobu stahování dat na minimum například použitím proxy serveru, paralelním stahováním ve více vláknech a dávkovým zpracováním klasifikovaných stránek. Přestože je desambiguace výpočetně náročná, hlavním důvodem malé rychlosti desambiguátoru je jeho implementace v Prologu. Řešením by mohla být reimplementace do jiného programovacího jazyka.

V současné době mohou být klíčovými slovy určenými systémem pouze jednotlivé termy. Ve skutečnosti však lze za klíčová slova považovat i víceslovné výrazy, označující jeden pojem. Úprava by pro systém znamenala nahrazení charakteristických slov jednotlivých domén charakteristickými n-gramy.

Literatura

[1] Kranti Kumar Ravi Arul Prakash Asirvatham. Web page categorization based on document structure.

[2] Necip Fazil Ayan. Using information gain as feature weight.

[3] Petr Berka. *Dobývání znalostí z databází*. Academia, 2003.

[4] Douglas Biber. The multi-dimensional approach to linguistic analyses of genre variation: An overview of methodology and findings. pages 331–345. Springer Netherlands.

[5] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[6] Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann Publishers, 2003.

[7] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. Technical report, Department of Computer Science National Taiwan University, Taipei 106, Taiwan, 2007.

[8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[9] Susan T. Dumais and Hao Chen. Hierarchical classification of Web content. In Nicholas J. Belkin, Peter Ingwersen, and Mun-Kew Leong, editors, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR, 2000. ACM Press, New York, US.

[10] Usama M. Fayyad and Keki B. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8:87–102, 1992.

[11] A. Finn and N. Kushmerick. Learning to classify documents according to genre, 2003.

[12] Robert K. France. Weights and measures: an axiomatic model for similarity computations. Technical report, Virginia Tech, 1994.

[13] Luigi Galavotti, Fabrizio Sebastiani, and Maria Simi. Feature selection and negative evidence in automated text categorization. In Marko Grobelnik, Dunja Mladenić, and Natasa Milic-Frayling, editors, *Proceedings of the ACM KDD-00 Workshop on Text Mining*, Boston, US, 2000.

[14] Michael Hoey. *Patterns of Lexis in Text*. Oxford University Press, 1991.

[15] Barbora Hladká Jan Hajič. Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. *Proceedings of the Conference COLING - ACL '98*, Montreal, Canada, 1998.

[16] Jussi Karlgren and Douglass Cutting. Recognizing text genres with simple metrics using discriminant analysis. In *Proceedings of the 15th. International Conference on Computational Linguistics (*COLING 94*)*, volume II, pages 1071 – 1075, Kyoto, Japan, 1994.

[17] Brett Kessler, Geoffrey Nunberg, and Hinrich Schütze. Automatic detection of text genre. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 32–38, Somerset, New Jersey, 1997. Association for Computational Linguistics.

[18] Adam Kilgarriff. Thesauruses for natural language processing. *Proc NLP-KE*, 2003.

[19] Adam Kilgarriff, Pavel Rychlý, Pavel Smrž, and David Tugwell. *The Sketch Engine in Practical Lexicography: A Reader*. 2008.

[20] Van Dijk T.A. Kintsch, W.A. Toward a model of text comprehension and production. *Psychological Review*, 1978.

[21] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.

[22] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145, 1995.

[23] H. P. Luhn. The automatic creation of literature abstracts. Technical report, April 1958.

[24] Hiroshi Motoda Makoto Tsukada, Takashi Washio. Automatic web-page classification by using machine learning methods. In *web intelligence : research and development*. Maebashi City , JAPON (23/10/2001), 2001.

[25] Charles T. Meadow. *Text Information Retrieval Systems*. Academic Press, Inc., Orlando, FL, USA, 1992.

[26] Miroslav Spousta Michal Marek, Pavel Pecina. Web page cleaning with conditional random fields. *Proceedings of the Web as Corpus Workshop (WAC3)*, September 2007.

[27] Christopher Tribble Mike Scott. *Textual Patterns: Key words and corpus analysis in language education*. Philadelphia: John Benjamins, 2006.

[28] D. Mladenic and M. Grobelnik. Feature selection for classification based on text hierarchy.

[29] D. Mladenic and M. Grobelnik. Assigning keywords to documents using machine learning, 1999.

[30] Dunja Mladenic. Turning yahoo to automatic web-page classifier. In *European Conference on Artificial Intelligence*, pages 473–474, 1998.

[31] John M. Pierre. On automated classification of web sites. Linkoping University Electronic Press, Sweden.

[32] Lubomír Popelínský and Jiří Materna. Keyness and relational learning. In *International conference Keyness in Text*, pages 61–63, 2007.

[33] T. Mitchell R. Michalski, J. Carbonell. *Machine Learning – An Artificial Intelligence Approach*. Springer Verlag, Berlin, Germany, 1984.

[34] Pavel Rychlý. *Korpusové Manažery a jejich efektivní implementace*. Ph.D. thesis, Faculty of Informatics, Masaryk Univerzity, Brno, 2000.

[35] Jones Sparck S. E. Robertson. Relevance weighting of search terms. *Journal of the American Society for Information Science*, pages 27(3):129–146, 1976.

[36] Marina Santini. State-of-the-art on automatic genre identification. In *Tech. Rep ITRI-04-03*. University of Brighton, 2004.

[37] Marina Santini. Some issues in automatic genre classification of web pages. In *ctes des 8 Journées internationales d'analyse statistiques des donnés textuelles*. University of Brighton, 2006.

[38] Radek Sedláček. *Morphemic Analyser for Czech*. Ph.D. thesis, Faculty of Informatics, Masaryk Univerzity, Brno, 2005.

[39] Katsuhiko Momoi Shanjian Li. A composite approach to language/encoding detection. *9th International Unicode Conference*, San Jose, California, 2001.

[40] Web source. http://cs.wikipedia.org/wiki/jazykový_korpus.

[41] Web source. http://en.wikipedia.org/wiki/support_vector_machines.

[42] Miguel Sousa. http://www.adhesiontext.com/.

[43] Benno Stein Sven Meyer zu Eissen. Genre classification of web pages. 2004.

[44] Radim Řehůřek. *Aplikace metod strojového učení Support Vector Machines na klasifikaci dokumentů v přirozeném jazyce pomocí dvou tříd*. Master thesis, Faculty of Informatics, Masaryk Univerzity, Brno, 2004.

[45] Pavel Šmerk. *Towards Morphological Disambiguation of Czech*. Ph.D. thesis proposals, Faculty of Informatics, Masaryk Univerzity, Brno, 2007.

[46] Ada Wai-Chee Fu Wai-Chiu Wong. Incremental document clustering for web page classification. Chinese University of Hong Kong, July 2000.

[47] Ian H. Witten and Eibe Frank. Data mining: Practical machine learning tools and techniques. Technical report, Morgan Kaufmann, San Francisco, 2005.

[48] Y. Yang and X. Liu. A re-examination of text categorization methods. In *22nd Annual International SIGIR*, pages 42–49, Berkley, August 1999.

Dodatek A

Značkový systém morfologického analyzátoru Ajka

k1 – substantivum

x	Speciální vzor
P	půl

g	Rod
M	Rod mužský životný
I	Rod mužský neživotný
N	Rod střední
F	Rod ženský
R	Rodina (příjmení)

n	Číslo
S	Číslo jednotné
P	Číslo množné
D	Duál
R	Označení rodiny

c	Pád
1	Pád první
2	Pád druhý
3	Pád třetí
4	Pád čtvrtý
5	Pád pátý
6	Pád šestý
7	Pád sedmý

w	Stylistický příznak
A	Archaismus
B	Básnicky
C	Pouze v korpusech
E	Expresivně
H	Hovorově
K	Knižně
O	Oblastně
R	Řidčeji
Z	Zastarale

z	Typ tvaru
S	S příklonným -s

k2 – adjektivum

e	Negace
A	Afirmace
N	Negace

g	Rod
M	Rod mužský životný
I	Rod mužský neživotný
N	Rod střední
F	Rod ženský

n	Číslo
S	Číslo jednotné
P	Číslo množné
D	Duál

c	Pád
1	Pád první
2	Pád druhý
3	Pád třetí
4	Pád čtvrtý
5	Pád pátý
6	Pád šestý
7	Pád sedmý

d	Stupeň
1	Pozitiv
2	Komparativ
3	Akuzativ

w	Stylistický příznak
A	Archaismus
B	Básnicky
C	Pouze v korpusech
E	Expresivně
H	Hovorově
K	Knižně
O	Oblastně
R	Řidčeji
Z	Zastarale

z	Typ tvaru
S	S příklonným -s

k3 – zájmeno

x	Druh zájmena (x)
P	Osobní
O	Přivlastňovací
D	Ukazovací
T	Vymezovací

y	Druh zájmena (y)
R	Reflexivní
Q	Tázací
R	Vztažné
N	Záporné
I	Neurčité

p	Osoba
1	První osoba
2	Druhá osoba
3	Třetí osoba
X	Jedna z předchozích

g	Rod
M	Rod mužský životný
I	Rod mužský neživotný
N	Rod střední
F	Rod ženský

n	Číslo
S	Číslo jednotné
P	Číslo množné
D	Duál

c	Pád
1	Pád první
2	Pád druhý
3	Pád třetí
4	Pád čtvrtý
5	Pád pátý
6	Pád šestý
7	Pád sedmý

w	Stylistický příznak
A	Archaismus
B	Básnicky
C	Pouze v korpusech
E	Expresivně
H	Hovorově
K	Knižně
O	Oblastně
R	Řidčeji
Z	Zastarale

z	Typ tvaru
S	S příklonným -s

k4 – číslovka

x	Druh číslovky (X)
C	Základní
O	Řadová
R	Druhová
G	Gramatika
H	Gramatika

y	Druh číslovky (y)
N	Záporná
I	Neurčitá

g	Rod
M	Rod mužský životný
I	Rod mužský neživotný
N	Rod střední
F	Rod ženský

n	Číslo
S	Číslo jednotné
P	Číslo množné
D	Duál

c	Pád
1	Pád první
2	Pád druhý
3	Pád třetí
4	Pád čtvrtý
5	Pád pátý
6	Pád šestý
7	Pád sedmý

w	Stylistický příznak tvaru
A	archaismus
B	básnicky
C	pouze v korpusech
E	expresivně
H	hovorově
K	knižně
O	oblastně
R	řidčeji
Z	zastarale

z	Typ tvaru
S	tvar s příklonným -s

k5 – sloveso

e	Negace
A	Afirmace
N	Negace

a	Vid
P	Perfektivum
I	Imperfektivum
B	Obouvidé

m	Typ (Mód)
F	Infinitiv
I	Infinitiv prézentu
R	Imperativ
A	Příčestí činné (minulé)
N	Příčestí trpné
S	Přechodník přítomný (současnost)
D	Přechodník minulý (dřívější děje)
B	Indikativ futura

p	Osoba
1	První osoba
2	Druhá osoba
3	Třetí osoba

n	Číslo
S	Číslo jednotné
P	Číslo množné

w	Stylistický příznak tvaru
A	archaismus
B	básnicky
C	pouze v korpusech
E	expresivně
H	hovorově
K	knižně
O	oblastně
R	řidčeji
Z	zastarale

z	Typ tvaru
S	tvar s příklonným -s

k6 – příslovce

e	Negace
A	Afirmace
N	Negace

x	Druh zájmenného příslovce (x)
D	Ukazovací
T	Vymezovací
M	Způsobové
S	Stavové

y	Druh zájmenného příslovce (y)
Q	Tázací
R	Vztažné
N	Záporné
I	Neurčité

d	Stupeň
1	Pozitiv
2	Komparativ
3	Superlativ

w	Stylistický příznak tvaru
A	archaismus
B	básnicky
C	pouze v korpusech
E	expresivně
H	hovorově
K	knižně
O	oblastně
R	řidčeji
Z	zastarale

z	Typ tvaru
S	tvar s příklonným -s

k7 – předložka

c	Pád
1	Pád první
2	Pád druhý
3	Pád třetí
4	Pád čtvrtý
5	Pád pátý
6	Pád šestý
7	Pád sedmý

k8 – spojka

x	Druh spojky
C	Souřadící
S	Podřadící

z	Typ tvaru
S	tvar s příklonným -s

k9 – částice

z	Typ tvaru
S	tvar s příklonným -s

k0 – citoslovce

kA – zkratka

kY – by, aby, kdyby

m	Vztah k slovesnému módu
C	kondicionál

p	Osoba
1	První osoba
2	Druhá osoba
3	Třetí osoba

n	Číslo
S	Číslo jednotné
P	Číslo množné

w	Stylistický příznak tvaru
A	archaismus
B	básnicky
C	pouze v korpusech
E	expresivně
H	hovorově
K	knižně
O	oblastně
R	řidčeji
Z	zastarale