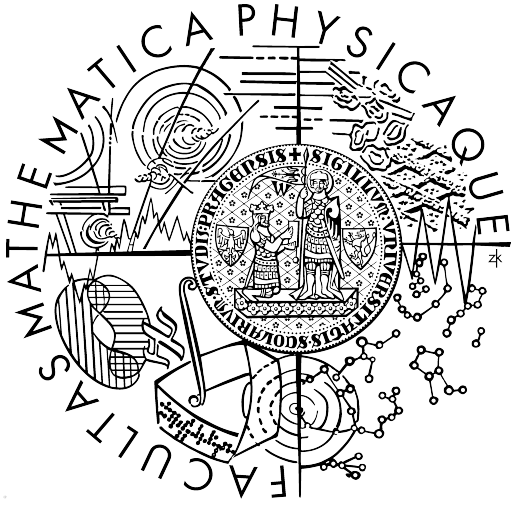
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Pavel Češka

Segmentace textu

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Pavel Pecina

Studijní program: Obecná informatika

2006

Děkuji vedoucímu své bakalářské práce Mgr. Pavlu Pecinovi za podporu a cenné rady udělované v průběhu psaní práce.
Děkuji Ústavu Českého národního korpusu za poskytnutí korpusů SYN2000 a SYN2005.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 29. května 2006

Pavel Češka

Obsah

Název práce: *Segmentace textu*
Autor: *Pavel Češka*
Katedra: *Ústav formální a aplikované lingvistiky*
Vedoucí bakalářské práce: *Mgr. Pavel Pecina*
email vedoucího: *pecina@ufal.mff.cuni.cz*
Abstrakt: *Tato bakalářská práce je zaměřena na základní předzpracování (tokenizaci a segmentaci) českého textu, zejména pro potřeby vytvoření českého internetového korpusu. Texty pro tento korpus budou automaticky získávány z Internetu, a proto samotné segmentaci předchází automatické určení kódování, čištění a rozpoznání jazyka dokumentu. Provádíme experimenty se dvěmi metodami rozpoznání jazyka a předkládáme jejich výsledky. První z metod je založena na porovnávání nejčetnějších n-gramů (podřetězců délky n) získaných z neznámého dokumentu a rozsáhlého českého korpusu. Druhá metoda využívá odhadu podmíněné pravděpodobnosti výskytu znakových trigramů získaných ze stejného korpusu. Pro širší použití je vytvořen modul pro tokenizaci a určování konců vět. Hledání konců vět je řešeno použitím seznamů českých zkratek a analýzou nejbližšího kontextu míst, která by mohla být za konce vět považována. Rozhodovací strom byl trénován na ručně označených datech. Vyhodnocení úspěšnosti bylo založeno na úsudcích nezávislé osoby a výsledky jsou předloženy v práci.*
Klíčová slova: *zpracování textu, tokenizace, segmentace, identifikace jazyka, rozhodovací strom*

Title: *Text segmentation*
Author: *Pavel Češka*
Department: *Institute of Formal and Applied Linguistics*
Supervisor: *Mgr. Pavel Pecina*
Supervisor's e-mail address: *pecina@ufal.mff.cuni.cz*
Abstract: *The bachelor thesis focuses on basic pre-processing (tokenization and segmentation) of Czech texts, mainly for purposes of Czech internet corpus. The texts for this corpus will be automatically obtained from the world wide web, therefore the segmentation is preceeded by character encoding recognition, cleaning and language identification. We performed experiments with two methods of language identification and present their results. The first method is based on comparison of the most frequent n-grams (substrings of length n) extracted from an unknown document and a large Czech corpus. The second one employs a model estimating word probabilities by conditional probabilities of trigrams estimated on the same corpus. For wider usage, we developed a module for tokenization and identification of sentences boundaries by a decision tree analysis of the nearest context of potential sentence boundaries and utilizing extensive lists of Czech abbreviations. The decision tree was trained on a set of manually processed data. Its evaluation was based on independent human judgements and results are presented in the work.*
Keywords: *text processing, tokenization, segmentation, language identification, decision tree*

Kapitola 1

Úvod

Matematická lingvistika se zabývá problémem automatického zpracování přirozeného jazyka. K mnoha problémům při této analýze lze přistupovat pomocí stochastických (statistických, pravděpodobnostních) metod. Obecně lze říci, že k úspěšnému použití těchto metod je potřeba zajistit dostatečně velké množství trénovacích dat (v našem případě se jedná o tzv. jazykový korpus). V současné době je největším českým jazykovým korpusem Český národní korpus (sečteme-li velikost jeho dvou největších publikovaných korpusů SYN2000 a SYN2005, dojdeme k číslu kolem 200 miliónů textových slov). I tento zdánlivě rozsáhlý korpus je pro řadu metod nepostačující, a proto vznikla myšlenka vytvoření českého internetového korpusu, který by měl obsahovat kvalitní české texty automaticky získávané z Internetu. Předpokládaná velikost tohoto korpusu by měla být v řádu miliard slov.

Internet je multijazykové prostředí, kde se navíc vyskytují i texty nekvalitní, nevhodné pro zařazení do korpusu (např. texty bez diakritiky apod.). Abychom mohli neznámý text z Internetu zařadit do korpusu, musíme rozpoznat jeho znakovou sadu (kódování) a určit, zdali je v požadovaném jazyce (v našem případě v češtině). Abychom zvýšili kvalitu vytvářeného korpusu, odstraníme nekvalitní části textu – například navigační menu, záhlaví a zápatí webových stránek atd. Text dále hierarchicky členíme strukturními značkami na menší celky – dokumenty, odstavce, věty, textová slova (tokeny). Tokeny mohou být slovní tvary, interpunkce, čísla a další zvláštní znaky. Takto předzpracovaný text již může být postoupen k dalšímu zpracování, např. morfologické analýze.

Cílem této bakalářské práce je provést co nejkvalitněji základní předzpracování textů a dokumentů v přirozeném jazyce a vyhodnotit jeho úspěšnost. Zvláštní zřetel bude kladen na identifikaci jazyka a rozpoznání začátků a konců vět.

V kapitole 2 je popsána příprava textů ke zpracování. V kapitole 3 je uveden popis rozpoznávání jazyka. Kapitola 4 se věnuje rozpoznávání textových slov (tokenizaci). Rozpoznávání začátků a konců vět (segmentaci) se podrobně věnuje kapitola 5. Kapitola 6 shrnuje a diskutuje dosažené výsledky.

Kapitola 2

Předzpracování textu

Volba pořadí jednotlivých kroků

Dokument před započetím tokenizace a segmentace projde fázemi předzpracování v tomto pořadí:

- rozpoznání kódování dokumentu
- čištění dokumentu
- rozpoznání jazyka (češtiny)

Pořadí těchto kroků bylo zvoleno záměrně, aby co nejlépe vyhovovalo potřebám zpracování dat z Internetu. Jelikož některé znakové sady zakódují jeden znak do sekvence bytů, je prvním krokem této práce rozpoznání kódování, abychom mohli text správně interpretovat. Než dojde k rozpoznání jazyka, odstraníme z dokumentu nekvalitní části textu (např. popisky obrázků, položky menu…). Tyto texty mohou být v jiném jazyce než českém, či se může jednat o názvy výrobků, společností atd., které by zhoršovaly identifikaci jazyka dokumentu.

Rozpoznání kódování

Abychom mohli znaky vstupního dokumentu správně interpretovat, musíme znát znakovou sadu (kódování), kterou používá. Dle normy HTML [8] by nás při stahování internetové stránky měl server informovat o jejím kódování. Nejpřímější cestou je využití parametru *charset* hlavičkového pole *Content-Type* HTTP protokolu. Například následující hlavička HTTP protokolu oznamuje použití kódování
ISO-8859-2:

Content-Type: text/html; charset=ISO-8859-2

Pokud tento parametr není uveden, norma HTTP protokolu stanovuje výchozí kódování ISO-8859-1. Toto doporučení je ale neužitečné, protože některé servery nemají dovoleno parametr *charset* posílat a jiné nejsou nastaveny tak, aby tento parametr odesílaly. Z tohoto důvodu může informaci o kódování obsahovat přímo HTML stránka za použití tohoto META elementu:

<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-2">

Mnohdy ale stránka informaci o kódování neobsahuje a může se dokonce stát i to, že uvedené kódování je chybné. Nemůžeme se tedy spolehnout na znalost kódování a musíme jej automaticky rozpoznat. Pro češtinu existuje kódování několik, z nichž nejrozšířenější jsou následující tři:

- CP 1250
- ISO 8859–2
- UTF–8

Pro první dvě uvedená kódování jeden znak odpovídá jednomu bytu. Na rozdíl od nich kódování UTF–8 může jeden znak kódovat do 1 až 6 bytů, přičemž délku každé této sekvence zjistíme podle hodnoty prvního bytu v této sekvenci a následujícího klíče:

- 0 – 127 = 1 byte
- 128 – 191 = chybná hodnota
- 192 – 223 = 2 byty
- 224 – 239 = 3 byty
- 240 – 247 = 4 byty
- 248 – 251 = 5 bytů
- 252 – 253 = 6 bytů
- 254 – 255 = speciální byte o délce 1 bytu

Zbylé byty v sekvenci určující kód pro jeden znak již mohou nabývat všech hodnot. Rozpoznání kódování UTF–8 je založeno na překontrolování, zdali je možné vstupní posloupnost interpretovat jako text kódovaný v této znakové sadě (tj. žádná sekvence nebude začínat bytem z rozmezí hodnot 128 – 191 a poslední sekvence na konci souboru bude ukončená). Je-li tomu tak, budeme text považovat za zapsaný pomocí znakové sady UTF–8, i když je zde velmi malá pravděpodobnost, že text může být zakódován jinak. Potom by ale tento text interpretovaný pomocí kódování UTF–8 tvořil nesmyslný text, který by později neprošel kontrolou jazyka a do korpusu proto nebyl zařazen.

K rozpoznání kódování CP 1250 a ISO 8859–2, která kódují každý znak vstupního řetězce vždy do jednoho bytu, je použit jednoduchý pravděpodobnostní model. Byty s hodnotou 0 – 127 odpovídají znakové sadě ASCII a jsou pro obě kódování stejná. V bytech s hodnotou 128 – 255 se tato kódování odlišují pouze pro znaky **ž**, **š**, **ť**, **Ž**, **Š** a **Ť**. Program nejdříve spočítá, kolik z bytů s hodnotou 128 – 255 by v daném kódování reprezentovalo písmena české abecedy, a u kterého kódování bude tento počet vyšší, pro to se rozhodne a bude jej považovat za znakovou sadu dokumentu.

Čištění dokumentu

Mnohá internetová stránka obsahuje položky menu, popisky obrázků atd. Tyto části textu jsou většinou jen několikaslovné, kvalitu získaného korpusu by snižovaly. Kvalitu textu můžeme poměřovat jak z hlediska jazykového, kdy chceme, aby dokument obsahoval celé věty, tak z hlediska obsahového, kdy odstraňujeme části, které netvoří tematický obsah dokumentu.

Jednoduchý algoritmus implementovaný v této práci vychází zejména z hlediska obsahového. Pokud není počet slov na řádce dostatečný, celá řádka bude z dokumentu odstraněna. Aby nedocházelo k nežádoucímu odstraňování nadpisů, algoritmus prochází text s okénkem o velikosti dvou řádek a nadpisy před souvislým textem ponechává.

Kapitola 3

Rozpoznávání jazyka

Předzpracovaný dokument již můžeme se znalostí znakové sady, v níž je kódován, správně interpretovat, zbývá nám určit, jestli je napsán v českém jazyce. K tomu můžeme použít následující přístupy řešení.

A) Rozpoznávání podle abecedy

Český jazyk používá unikátní abecedu, v jejímž rozpoznání by určení jazyka mohlo spočívat. Unikátnost abecedy zapříčiňují především akcentované znaky. V českém jazyce se jedná o tyto znaky: á, č, ď, é, ě, í, ň, ó, ř, š, ť, ú, ů, ý, ž. V porovnání s jazykem slovenským z nich jako unikátní zbývají pouze znaky ě, ř, ů, navíc s relativně nízkou četností (hodnoty jsou získány z korpusů SYN2000 a SYN2005) [5], [6]:

- ě 1,3292%
- ř 1,0102%
- ů 0,4219%

Dohromady tedy 2,7613%. Pravděpodobnost, že stoznakový řetězec českého textu nebude obsahovat žádný z těchto znaků, činí $(1 - 0{,}027613)^{100}$. Což je po vyčíslení více než 6% a tuto pravděpodobnost nemůžeme zanedbat. Rozpoznání češtiny od slovenštiny by v těchto případech nebylo možné, a proto tuto metodu pro určení jazyka nepoužijeme.

B) Pravděpodobnost trigramů

Další z metod pro rozpoznání jazyka vychází z pozorování, že přirozené jazyky používající abecedu jsou velmi nenáhodné a konzistentní v používání jednotlivých písmen, ale i řetězců písmen. Řečeno jinými slovy, každý jazyk používá unikátní nebo velmi charakteristickou abecedu, písmena této abecedy se vyskytují s překvapivou stálostí frekvence a navíc frekvence dvojic, trojic, čtveřic a pětic písmen jsou pro přirozené jazyky charakteristické a mezi různými jazyky dostatečně odlišné, jak tvrdí Beesley [3].

Abychom mohli rozpoznávat jazyk neznámého dokumentu, nejdříve spočteme frekvence n-gramů (podřetězců délky n znaků) u dostatečně velkého korpusu, jehož

jazyk známe a metodou maximální věrohodnosti odhadneme jejich pravděpodobnostní rozdělení. Pro každý jazyk, který chceme rozpoznávat, zvlášť.

$$P(w_1 w_2 w_3) = \frac{C(w_1 w_2 w_3)}{N}$$

Ke každému slovu neznámého textu přidáme na jeho počátek a konec podtržítko (reprezentující mezeru mezi slovy) a rozdělíme na n-gramy. Každému n-gramu přiřadíme jeho pravděpodobnost pro každý jazyk. Pravděpodobnost toho, že slovo je napsáno některým z těchto jazyků, zjistíme prostým vynásobením pravděpodobností jednotlivých n-gramů:

$$P(w_1 w_2 \ldots w_n) = P(\_w_1 w_2 \ldots w_n\_) = P(\_w_1) * P(w_1 w_2) * \ldots * P(w_{n-1} w_n) * P(w_n\_)$$

Pro každé slovo určíme nejpravděpodobnější jazyk a z výsledků u všech slov určíme jazyk neznámého textu. Touto metodou se podrobněji zabýval Beesley [3].

Pro naše účely, kdy nás zajímá pouze to, zda je text v českém jazyce nebo zda je nevhodný pro zařazení do korpusu, není tato metoda nejvhodnější. Samozřejmě není v našich silách odhadnout pravděpodobnosti všech n-gramů ve všech lidských jazycích, které můžeme na Internetu nalézt. Ale tuto skutečnost můžeme obejít stanovením hranice pravděpodobnosti, která bude pro přijetí slova jako českého určující. Musíme brát v úvahu i délku slova, protože pravděpodobnosti n-gramů jsou v řádu tisícin procent a dlouhá slova budou mít nižší pravděpodobnost. Ale ani toto zohlednění délky nám nepomůže, jak ilustrují následující příklady stejně dlouhých slov:

sťalo	***_prone_***
P(_sť) = 0, 000 365 323 . 10^{-3}	*P(_pr) = 4, 671 096 838 . 10^{-3}*
P(sťa) = 0, 019 720 248 . 10^{-3}	*P(pro) = 2, 990 866 765 . 10^{-3}*
P(ťal) = 0, 002 538 875 . 10^{-3}	*P(ron) = 0, 149 317 989 . 10^{-3}*
P(alo) = 0, 501 650 855 . 10^{-3}	*P(one) = 0, 246 478 723 . 10^{-3}*
P(lo_) = 1, 340 773 844 . 10^{-3}	*P(ne_) = 0, 528 208 320 . 10^{-3}*
P(_sťalo_) = 0, 000 000 012 . 10^{-15}	*P(_prone_) = 0, 271 589 355 . 10^{-15}*
hlavu	***_nasty_***
P(_hl) = 0, 522 953 104 . 10^{-3}	*P(_na) = 4, 603 066 018 . 10^{-3}*
P(hla) = 0, 590 694 543 . 10^{-3}	*P(nas) = 0, 123 329 693 . 10^{-3}*
P(lav) = 0, 522 925 125 . 10^{-3}	*P(ast) = 0, 750 463 009 . 10^{-3}*
P(avu) = 0, 184 986 945 . 10^{-3}	*P(sty) = 0, 145 673 552 . 10^{-3}*
P(vu_) = 0, 364 405 313 . 10^{-3}	*P(ty_) = 0, 735 032 309 . 10^{-3}*
P(_hlavu_) = 0, 010 889 075 . 10^{-15}	*P(_nasty_) = 0, 045 617 479 . 10^{-15}*

3. 1 Pravděpodobnosti slov podle metody B)

Při porovnávání pravděpodobností pouze s jedním jazykovým modelem lze jen stěží rozhodnout, jestli je text napsán v jazyce českém, či nikoliv. Určení hranice, která by toto posuzovala, je nemožné, jak ukazují výše zmíněné příklady.

C) Porovnání nejčetnějších n-gramů

I tato metoda vychází z konzistentního používání n-gramů v přirozených jazycích. Nahlíží na tento problém z jiného úhlu a všímá si, že pořadí prvních 200 až 400 nejčetnějších n-gramů (pro n = 1, ..., 5) je pro dva dokumenty napsané ve stejném jazyce velmi podobné.

Pro každý jazyk, který budeme chtít rozpoznávat, vytvoříme pořadí četností všech n-gramů jazyka. Důležité pro další zpracování bude pouze pořadí 400 nejčetnějších, které seřadíme od nejčetnějšího po nejméně četný a nazveme profilem jazyka. K rozpoznání jazyka neznámého dokumentu budeme potřebovat též jeho profil. U každého n-gramu profilu neznámého dokumentu změříme rozdíl mezi pořadím v profilu jazyka a profilu neznámého textu. Prostým sečtením všech rozdílů u 400 nejčetnějších n-gramů získáme vzdálenost jednotlivých profilů.

	Profil neznámého textu	**Profil jazyka**	**Rozdíl pořadí**
nejčetnější	ní_	_po	1
	_a_	ní_	1
	_pr	_a_	2
	_ne	_ne	0
	_ww	_pr	nenalezen = maximum
nejméně četný	...	...	...
			součet = vzdálenost profilů

3. 2 Vzdálenost profilů

Za jazyk neznámého dokumentu prohlásíme profil s nejmenší vzdáleností. Více se o této metodě můžete dozvědět v článku Cavnara a Trenkleho [4].

Použití tohoto postupu pro rozpoznání češtiny však opět nepřineslo požadovanou úspěšnost. Původní metoda počítá s využitím několika jazykových profilů a jejich porovnáváním, o což my neusilujeme, jak bylo zmíněno již dříve. Určení hranice, kdy se ještě jedná o český text a kdy už nikoliv, je nemožné kvůli neostré hranici, zejména pro češtinu a slovenštinu bez diakritiky. Využití přesně tohoto postupu je tedy pro naši práci nevhodné, ale mohlo by sloužit jako základ pro metodu novou.

Hlavní nedostatky řešení C)

Největším nedostatkem tohoto řešení je nezohledňování pořadí u n-gramů se stejnou frekvencí, což zejména u kratších textů může způsobovat značné zkreslení výsledku. Několik desítek n-gramů bude mít pouze 1, 2 či 3 výskyty a pořadí mezi n-gramy se stejnou četností bude určeno náhodně, nezávisle na četnosti. N-gramy s tímto řídkým výskytem navíc nemusejí být typické pro český jazyk a v profilu vyrobeném z obsáhlého korpusu budou zaujímat místa daleko za 400. příčkou. Zde však z důvodu krátké délky vstupních dat obsadí příčky mnohem vyšší.

Dalším aspektem, kterému původní metoda C nepřikládá váhu, je přesná míra odlišnosti frekvencí nejčetnějších n-gramů u porovnávaných profilů. Bude-li mít určitý n-gram stejnou frekvenci u obou profilů, není to určující – záleží pouze na rozdílu v pořadí v těchto profilech.

Jelikož metoda C počítá výsledný rozdíl profilů z nejčastějších 400 n-gramů neznámého textu, nekontroluje tím použití typických n-gramů češtiny, což lze považovat za poslední vážný nedostatek. Půjde-li například o český text bez diakritiky, rozdíl jeho profilu a profilu češtiny měřený metodou použitou v řešení C bude malý, ale tento text je pro zařazení do našeho korpusu naprosto nevhodný.

D) Vylepšená metoda porovnání nejčetnějších n-gramů

První ze zvolených řešení vychází z metody porovnávání nejčetnějších n-gramů a upravuje ji, aby lépe vyhovovala rozpoznání pouze jediného jazyka – češtiny. Odstraňuje všechny výše zmíněné nedostatky a zaměřuje se zejména na odlišení textů psaných ve slovenském a českém jazyce bez diakritky od češtiny s diakritikou. Přitom však zůstává funkční i pro odfiltrování všech jiných jazyků než češtiny.

K odstranění prvního problému bylo použito zkrácení délky profilu, která bude pro porovnávání důležitá. Hranice porovnávaných n-gramů byla experimentálně stanovena na 1/8 počtu všech n-gramů neznámého textu, maximálně však 400 n-gramů. Není zde nic speciálního na 1/8 počtu všech n-gramů, až na to, že tato hranice odstraní všechny n-gramy s nízkým výskytem a velikost zbytku je stále dostatečně průkazná k určení toho, zdali je dokument český. Zároveň se ukázalo, že je lepší stanovit délku porovnávaného profilu podle rozmanitosti textu (tj. počtu různých n-gramů), než odstraněním n-gramů s nízkým výskytem. K hodnotě této hranice bylo dospěno krátkým průzkumem, někdo může samozřejmě vytvořit propracovanější statistiku a tuto mez určit přesněji.

Zbylé dva problémy byly vyřešeny volbou nové metriky pro porovnávání profilu češtiny a profilu neznámého dokumentu. Můžeme si všimnout, že metrika popsaná Cavnarem a Trenklem [4], kdy pro každý n-gram z profilu neznámého textu určuji jeho odchylku v pořadí od nejčastějších n-gramů pro češtinu, má jinou hodnotu, než je ta, kterou získáme vypočítáním odchylky 400 nejčastějších n-gramů českého jazyka od pořadí těchto n-gramů v neznámém textu. Autoři svou volbu metriky nevysvětlují, můžeme se jen domnívat, že toto řešení zvolili proto, že u kratších textů,

které mají profil menší než 400 n-gramů, jsou vypočítány odchylky všech n-gramů, čímž získáme dostatek informací k určení jazyka i z velmi krátkých textů. Tato informace z více n-gramů je pro nás důležitá, neboť budeme moci porovnávat delší profil neznámeho dokumentu s profily jazyků, které budeme rozpoznávat, a tím i s větší jistotou vybrat z těchto jazyků ten, jenž má nejmenší odchylku profilů. Pro naše účely je ovšem lepší použít druhou z popisovaných metrik, kterou lze jinými slovy popsat jako odchylku četností nejčetnějších českých n-gramů (jejichž počet bude roven výše zmiňované 1/8 celkového počtu n-gramů, maximálně však 400) od četností těchto n-gramů v neznámém textu.

Nebudeme též odchylku profilů počítat z pořadí jednotlivých n-gramů v profilu, ale podle odchylky četností jejich výskytů podle následujícího vztahu:

$$\frac{|P_{czech}(\text{n-gram}) - P_{unknown}(\text{n-gram})|}{P_{czech}(\text{n-gram})}$$

Bude-li tato hodnota větší než 1 (tj. výskyt n-gramu v neznámém textu bude vyšší než dvojnásobek výskytu v češtině), definujeme tuto hodnotu rovnu 1. Též si můžeme všimnout, že pokud má některý n-gram nulový výskyt, bude hodnota odchylky rovněž maximální – rovna jedné. Z tohoto vztahu je patrná snaha o normování velikosti odchylky u každého jednotlivého n-gramu. Ze všech těchto odchylek spočítáme aritmetický průměr, který nám určí s jakou pravděpodobností je daný dokument napsán v českém jazyce. Toto číslo, které bude z intervalu <0; 1>, nazveme P-hodnotou dokumentu.

Nastavení prahu pro rozpoznání češtiny

Abychom mohli neznámý dokument s vypočítanou P-hodnotou přijmout za český nebo jej zamítnout, musíme stanovit hranici, která o zařazení rozhodne. O stanovení této hranice jsme se nejdříve pokusili analýzou P-hodnot 20 dokumentů, z nichž je 9 českých včetně diakritiky, 4 české bez diakritiky, 2 slovenské bez diakritiky, 3 bosenské bez diakritiky, 1 text chorvatský bez diakritiky a 1 text anglický. Texty lze nalézt na přiloženém CD ve složce *langid_1*. O P-hodnotách těchto dokumentů referuje následující tabulka:

bosnian1	0,577	cze3	0,418
bosnian2	0,551	cze4	0,434
bosnian3	0,607	cze5	0,433
croatian	0,582	cze6	0,381
cze_ascii1	0,525	cze7	0,326
cze_ascii2	0,484	cze8	0,488
cze_ascii3	0,510	cze9	0,422
cze_ascii4	0,507	english	0,700
cze1	0,342	slovak_ascii1	0,528
cze2	0,300	slovak_ascii2	0,528

3. 3 P-hodnoty testovaných dokumentů I

Testované české dokumenty mají P-hodnotu nejvýše 0,488, nečeské dokumenty mají P-hodnotu nejméně 0,484. Jazyky, které mají nejpodobnější histogram četností n-gramů češtině, jsou čeština bez diakritiky a slovenština bez diakritiky. Proto jsme provedli další analýzu textů v těchto jazycích a navíc zohlednili délku dokumentu. Texty lze nalézt na přiloženém CD ve složce *langid_2*.

čeština		**čeština bez diakritiky**		**slovenština bez diakritiky**	
délka	*P-hodnota*	*délka*	*P-hodnota*	*délka*	*P-hodnota*
054	0,496	052	0,379	051	0,562
059	0,640	055	0,478	054	0,623
060	0,472	062	0,412	069	0,542
065	0,502	068	0,624	073	0,432
083	0,448	083	0,598	074	0,434
085	0,464	087	0,481	084	0,397
100	0,468	104	0,493	091	0,483
144	0,494	133	0,522	151	0,534
202	**0,383**	**224**	**0,499**	**216**	**0,533**
221	**0,306**	**231**	**0,512**	**243**	**0,423**
500	**0,379**	**537**	**0,524**	**522**	**0,536**

3. 4 P-hodnoty testovaných dokumentů II

Pro rozpoznávání textů kratších než 200 znaků se tento algoritmus nehodí, nedokáže rozpoznat češtinu od češtiny bez diakritiky a slovenštiny bez diakritiky. Pro delší texty již vykazuje dobré výsledky s prahem nastaveným od hodnoty 0, 420 (pro vyšší jistotu, že testem neprojde nečeský dokument) až po hodnotu 0, 450 (která propustí více českých dokumentů, ale může se mezi nimi vyskytnout i jiný).

E) Metoda podmíněné pravděpodobnosti trigramů

Mějme slovo, u kterého chceme určit jeho pravděpodobnost výskytu. Přidejme na začátek a na konec slova 2 podtržítka, abychom pak podle nich mohli u trigramů rozpoznat začátek a konec slova. Pravděpodobnost výskytu tohoto slova, tj. n-tici $w_1w_2...w_n$ můžeme vyjádřit pomocí podmíněné pravděpodobnosti tímto způsobem:

$$P(w_1w_2\dots w_n) = P(w_n \mid w_1w_2\dots w_{n-1}) * P(w_{n-1} \mid w_1w_2\dots w_{n-2}) * \dots * P(w_3 \mid w_1w_2) * P(w_2 \mid w_1) * P(w_1) \doteq$$

$$\doteq P(w_3 \mid w_1w_2) * P(w_4 \mid w_2w_3) * \dots * P(w_n \mid w_{n-2}w_{n-1}) =$$

$$= \frac{P_3(w_1w_2w_3)}{P_2(w_1w_2)} * \frac{P_3(w_2w_3w_4)}{P_2(w_2w_3)} * \dots * \frac{P_3(w_{n-2}w_{n-1}w_n)}{P_2(w_{n-2}w_{n-1})}$$

kde P značí pravděpodobnost výskytu n-tice, P_2 pravděpodobnost výskytu dvojice, P_3 pravděpodobnost výskytu trojice.

Nyní můžeme u každého slova vstupního dokumentu určit pravděpodobnost jeho výskytu. Bude-li tato pravděpodobnost malá, budeme slovo považovat za nečeské. Jazyk celého dokumentu budeme posuzovat podle poměru českých slov ke všem

slovům, která obsahuje. K tomu, abychom rozhodli, jestli je dokument český, musíme stanovit tyto dvě hranice.

Určení parametrů pro rozpoznání češtiny

První parametr (uznání slova jako českého) byl stanoven z analýzy českého textu bez diakritiky, slovenského textu bez diakritiky a anglického textu. Každému slovu jsme vypočítali jeho pravděpodobnost a podle ní jsme slova setřídili. Mnoho slov s nejvyšší pravděpodobností můžeme považovat za česká (v tabulce jsou znázorněna tučně). Hraniční hodnotou, po kterou se v textu tato slova nejvíce vyskytují, je přibližně hodnota -2,100 (všechny hodnoty pravděpodobností jsou zlogaritmovány přirozeným logaritmem).

Anglický text

a	**-1,392**	promise	-1,908	information	-2,028
to	**-1,475**	sit	-1,911	for	-2,031
do	**-1,481**	novice	-1,917	programs	-2,041
by	**-1,514**	**let**	**-1,936**	systém	-2,057
ve	**-1,663**	them	-1,936	force	-2,063
program	**-1,743**	**most**	**-1,941**	into	-2,065
problem	-1,790	**i**	**-1,967**	prefer	-2,066
same	-1,818	real	-1,978	**no**	**-2,066**
me	-1,848	material	-1,988	**reference**	**-2,071**
s	**-1,849**	balance	-1,991	**list**	**-2,073**
set	**-1,864**	since	-1,997	master	-2,076
once	-1,888	**port**	**-2,002**	**man**	**-2,077**
many	-1,898	mini	-2,008	tiny	-2,091
de	-1,905	any	-2,018		

3. 5 Pravděpodobnosti slov anglického textu

Český text bez diakritiky

se	**-1,193**	**svou**	**-1,656**	**s**	**-1,849**
na	**-1,215**	**ve**	**-1,663**	myslenky	-1,863
jak	**-1,316**	poslednich	-1,687	**z**	**-1,876**
a	**-1,392**	**povinnosti**	**-1,744**	musi	-1,889
jsem	**-1,454**	**ale**	**-1,760**	**o**	**-1,915**
pro	**-1,456**	**nej**	**-1,807**	podelit	-1,954
to	**-1,477**	**moc**	**-1,827**	naladen	-1,967
do	**-1,481**	**taky**	**-1,827**	**i**	**-1,967**
tak	**-1,544**	posledni	-1,837	nove	-2,059
bude	**-1,562**	**nikoli**	**-1,842**	**dlouho**	**-2,075**
si	**-1,565**	**mohl**	**-1,848**		

3. 6 Pravděpodobnosti slov českého textu bez diakritiky

Slovenský text bez diakritiky

na	**-1,215**	**domu**	**-1,656**	**taky**	**-1,827**
je	**-1,292**	**po**	**-1,673**	**s**	**-1,849**
a	**-1,392**	**sebou**	**-1,693**	**spravit**	**-1,896**
to	**-1,475**	**jeden**	**-1,703**	**o**	**-1,915**
praci	-1,495	**co**	**-1,746**	**tu**	**-1,947**
v	**-1,529**	pracou	-1,748	**ani**	**-2,036**
tak	**-1,544**	**velmi**	**-1,753**	**zda**	**-2,046**
bude	**-1,562**	**ale**	**-1,760**	dnoch	-2,074
za	**-1,596**	**kde**	**-1,782**	moze	-2,098
mi	**-1,635**	**ze**	**-1,816**		

3. 7 Pravděpodobnosti slov slovenského textu bez diakritiky

Druhý parametr jsme určili po analýze poměru „českých“ a „nečeských“ slov u 11 českých textů s diakritikou, 11 českých textů bez diakritiky a 11 slovenských textů bez diakritiky o různé délce. Jeho hodnota je 0,450. Vypočítané poměry „českých“ a „nečeských“ slov u těchto dokumentů můžete vidět v následující tabulce. Texty, které byly špatně identifikovány, jsou vyznačeny tučně:

čeština		**čeština bez diakritiky**		**slovenština bez diakritiky**	
délka	*poměr*	*délka*	*poměr*	*délka*	*poměr*
054	0,250	052	0,500	051	0,625
059	0,400	055	0,625	**054**	**0,400**
060	0,444	062	0,583	069	0,769
065	0,181	068	0,500	073	0,667
083	0,308	**083**	**0,385**	074	0,727
085	0,333	087	0,500	084	0,625
100	0,067	104	0,467	**091**	**0,438**
144	0,238	**133**	**0,273**	151	0,556
202	0,222	224	0,621	216	0,452
221	0,514	**231**	**0,447**	243	0,581
500	**0,451**	537	0,479	522	0,529

3. 8 Hodnota kvality češtiny dokumentů podle metody E)

Vyhodnocení úspěšnosti identifikace jazyka

Z 261 544 dokumentů stažených z Internetu v roce 2003 z 9 různých zpravodajských serverů jsme náhodně vybrali 1 547 dokumentů. Ty jsme nechali zpracovat a setřídili je podle výsledků metody D) pro rozpoznání jazyka a z tohoto seznamu jsme rovnoměrně vybrali 107 dokumentů (tak abychom pokryli celou škálu výsledných skóre). U nich jsme potom bez znalosti výsledku metody D) a E) manuálně určili jazyk. Poté jsme je nechali zpracovat těmito metodami a spočítali úspěšnost identifikace českého jazyka.

Vylepšená metoda porovnání nejčetnějších n-gramů

Ze všech 107 dokumentů tato metoda velmi těsně (s P-hodnotami 0,45301, 0,45349 a 0,45491) označila 3 české dokumenty jako nečeské. Úspěšnost této metody na testovaných datech je tedy 97,20%.

Metoda podmíněné pravděpodobnosti trigramů

Tato metoda označila 9 českých dokumentů za nečeské. Úspěšnost metody na testovaných datech činí 91,59%. Přesto ji nelze jednoduše považovat za horší než metodu předchozí. Chybně určené dokumenty jsou sice napsány v českém jazyce, ale obsahují velmi mnoho cizích jmen (názvy cizích osob, apod.). Navíc i jejich obsahová kvalita je velmi malá. Spojíme-li s požadavkem česky psaného textu i nároky na obsahovou kvalitu, jeví se tato metoda vhodnější.

Kapitola 4

Tokenizace

Pod pojmem tokenizace se rozumí identifikace tokenů – jednotlivých slov, interpunkce a mezer. Na tokenizaci obvykle navazuje segmentace, která se stará o rozpoznávání začátků a konců vět. V této kapitole se budeme zabývat tokenizací, o identifikaci vět pojednává kapitola následující.

Definice slova

Problém rozdělení vstupního textu na slova, interpunkci a mezery se na první pohled může jevit velice jednoduchým, což ovšem není úplně pravda, jak dokládá i Hajič v [2]:

„Jakkoli triviální se tento úvodní problém může zdát, není tomu tak; již jen definice toho, co to je "slovo" je někdy nejasná: je byl-li, pracovals, technicko-hospodářský nebo naň jedno slovo, nebo dvě? Je New York nebo Kostelec n./Č. lesy jedno slovo, nebo dvě (resp. pět slov)? Obvykle se volí nějaký relativně dobře definovatelný kompromis. Zdá se, že z hlediska dalšího zpracování je vhodné v nejasných případech za slovo brát jednotku co nejkratší.“

Za slovo v této práci budeme považovat souvislou posloupnost písmen z české abecedy doplněnou o číslice 0–9 a písmena jiných národních abeced (tato množina je přesně definována v souboru *letters.dat* a lze ji modifikovat), která se mohou objevit v českých textech u jmen osob, míst apod., což odpovídá i definici slova v Českém národním korpusu SYN2000 (v novějším korpusu SYN2005 je slovo již definováno složitěji).

Příklad:
Řetězec technicko-hospodářský rozdělíme do tří slov: "technicko", "-" a "hospodářský".

Standard CSTS

Pro popis rozdělení textu na tokeny, segmenty apod. se používá formát CSTS založený na standardu SGML, který používá i Pražský závislostní korpus verze 1.0 (Prague Dependency Treebank 1.0)[1]. Dle tohoto standardu je každý úsek textu uvozen otevírací značkou ve formátu <značka> a ukončen buď značkou ve formátu

</značka> nebo jinou otevírací značkou na stejné úrovni. Podle jejich účelu můžeme značky rozdělit na značky správní a strukturní.

Správní značky

Tvoří hlavičku dokumentu, která obsahuje administrativní údaje o těle dokumentu. Pro naše účely budou postačovat následující značky:

- <csts> – jazyk dokumentu
- <doc> – identifikátor dokumentu
- <a> – informace o modu, typu, žánru, zdali je text veršován, druh média, na kterém byl publikován, pohlaví autora, jazyk, rok vydání, 1. rok vydání apod.
- <c> – samotný obsah dokumentu

Strukturní značky

Strukturní značky člení text do hierarchicky uspořádaných odstavců, vět a tokenů.

- <p> – odstavec
- <s> – věta
- <f> – slovo, ne však interpunkce
- <d> – interpunkce
- <D> – informace o tom, že mezi předcházejícím a následujícím tokenem není mezera

Použijeme i několik lingvistických značek, které popisují morfologickou interpretaci tokenu (jedná se o atributy značky <f>). Token <f> může být doplněn jedním z následujících atributů:

- abbr – zkratka psaná malými písmeny
- cap.abbr – zkratka s prvním písmenem velkým
- upper.abbr – zkratka se všemi písmeny velkými
- mixed.abbr – zkratka složená z písmen i číslic
- cap – první písmeno je velké
- mixed – slovo složeno z písmen i číslic
- num – slovo je číslo zapsané číslicemi
- upper – všechna písmena jsou velká

Příklad:
<csts lang=cs>
<doc>
<a>

**
<c>
* *
<s>
<f cap.abbr>Ing.
<f cap>Novák
<f>z
<f upper>ČVUT
<f>navštívil
<f>v
<f>roce
<f num>2006
<f>již
<f num>2
<D>
<d>.
<f>konferenci
<D>
<d>.
</s>
* *
</c>
</doc>
</csts>

Token <enter>

Pro potřeby zaznamenání informace o lámání textu v původním dokumentu (tzv. newline), která bude sloužit pro speciální případ identifikace konců vět při segmentaci, byla zavedena nová pomocná značka <enter>, která slouží jen pro potřeby segmentace, a po zpracování textu tokenizací a segmentací se ve výstupním souboru již neobjeví a formát CSTS nebude porušen.

Identifikace čísel

Tokenizer mimo souvislou posloupnost znaků 0 – 9 identifikuje i tyto základní tvary čísel a reprezentuje je pomocí jediného tokenu:

- číslo, kterému bez merezy předchází znaménko (plus či mínus)
- desetinné číslo zapsané bez mezer mezi celou částí, desetinnou čárkou a desetinnou částí

Implementace tokenizace

Po definování pojmu slovo je již implementace tokenizace snadným úkolem. Vstupní text je dle této definice rozdělen na jednotlivé tokeny, kterým je přiřazena odpovídající strukturní značka, případně doplněná o atributy .

Kapitola 5

Segmentace

Co je to věta

Pod pojmem segmentace budeme rozumět rozdělení textu, který již prošel tokenizací (je rozdělen na textová slova – tokeny), na věty. Jako mnoho pojmů z lingvistiky lze i větu chápat více způsoby. Definice věty jsou pro naše účely málo formální, jak ukazuje například [7]:

„...věta je promluva sdělná, kterou se mluvčí aktivně a způsobem zanechávajícím po stránce formální dojem obvyklosti a subjektivní úplnosti staví k nějakému faktu nebo ke skupině faktů.“

Nebudeme se snažit o formálnější definici věty, ale pokusíme se určit znaky, za kterými konec věty můžeme očekávat (věty těmito znaky obvykle ukončujeme). Tyto znaky budeme hledat a pokusíme se rozhodnout, jestli v konkrétním případě konec věty označují, či ne. Tuto definici můžeme poté použít pro vyhodnocení úspěšnosti našeho algoritmu .

Symboly označující možný konec věty

Česká věta musí končit jedním z těchto symbolů:

- tečka **.**
- vykřičník **!**
- otazník **?**
- uvozovka “
- pravá závorka **)**

Ne vždy však tento symbol konec věty znamená.

- Tečka může být použita jako součást elipsy (výpustky), která může stát uprostřed věty, ale i na jejím konci. Za konec věty též nepovažujeme tečku u řadových číslovek či tečku za zkratkou (nestojí-li na konci věty, kde plní

obě funkce – atribut číslovky, či zkratky a konec věty). Tečka taktéž nekončí větu v přímé řeči, kde tuto funkci přebírá uvozovka.

- Vykřičník za konec věty nepovažujeme uvnitř přímé řeči a nesmíme zapomenout na matematický význam vykřičníku (faktoriál), který též konec věty neznačí.

- Jediným problémem u otazníku je otázka v přímé řeči, kde funkci konce věty přebírá uvozovka.

- Uvozovka značí konec věty jedině jako znak pro ukončení přímé řeči.

- Pravou (uzavírací) závorku považujeme za konec věty, pokud je uvnitř závorek celá věta

Za větu budeme považovat i nadpisy (u článků, kapitol atd.), kde je pro nás však rozlišení konce věty velmi obtížné. Pokud bychom chtěli postupovat korektně, museli bychom mezi symboly znamenající možný konec věty zařadit i mezeru mezi slovy (či uvažovat o konci věty za každým slovem). Rozpoznávání nadpisů tímto způsobem by bylo velice náročné a neefektivní, proto budeme o konci věty uvažovat pouze za pomocným tokenem <enter>, který tokenizer vložil tam, kde na vstupu načetl znak nové řádky (\n).

Rozhodovací strom (Decision Tree)

Jedním z možných řešení problému segmentace textu by bylo popsání všech pravidel, kterými se rozpoznání konců vět řídí, a popsání výjimek z těchto pravidel, případně výjimek z těchto výjimek. Jako výsledek bychom obdrželi strom podmínek, podle kterého bychom určili, zda se za místem možného výskytu konce věty konec věty nachází, či nikoliv. Tímto způsobem získaný strom by dobře popisoval lidské uvažování při řešení tohoto problému, ale jeho implementace přináší několik potíží. Jak máme strom vytvořit, abychom se neptali na tutéž věc dvakrát? Jaké je nejvýhodnější pořadí pravidel? Bude náš strom dobře vyhodnocovat i případy, se kterými jsme nepočítali? Nevytvoříme pravidlo, které bude při použití zbytečné či dokonce chybné?

Pro vyřešení nastíněných otázek jsme se rozhodli využít prediktivního modelu nazývaného rozhodovací strom (Decision Tree), jenž se používá na řešení problémů, kde na malé množině dat, u které známe její vlastnosti i klasifikaci, nalezneme pravidla ke klasifikaci dalších dat s vlastnostmi ze stejných domén. Rozhodovací strom je speciální případ stromové struktury, ve které listy reprezentují klasifikaci (v našem případě jednu z hodnot: je konec věty; není konec věty) a větve konjunkci vlastností, které vedou k tomuto ohodnocení.

Atributy rozhodovacího stromu

Nejdříve musíme zvolit vlastnosti (atributy), které považujeme za důležité při ohodnocování potenciálních konců vět. Všechny tyto vlastnosti budou nabývat pouze dvou hodnot, a proto vytvořený rozhodovací strom bude binární. Při volbě těchto atributů jsme brali v úvahu nejbližší kontext místa, kde uvažujeme o konci věty, a podobu aktuálního tokenu.

- Prvních pět atributů slouží k rozpoznání symbolu označujícího možný konec věty.
- Volba pátého tokenu má pomoci při identifikaci konce věty na konci odstavce či celého dokumentu.
- Šestý a sedmý atribut si všímá velikosti počátečního písmena u následujícího slova, protože víme, že nová věta začíná velkým písmenem.
- Osmý a devátý token pomáhá identifikovat konec přímé řeči a výpustku.
- Desátý atribut pomáhá při identifikaci pořadových čísel ve sportovních výsledcích.
- Jedenáctý atribut slouží k identifikaci pořadových čísel.
- Dvanáctý až čtrnáctý atribut slouží k rozpoznání zkratek z výše zmíněných kategorií zkratek.
- Patnáctý atribut slouží k rozeznání zkratek a pořadových čísel.
- Šestnáctý atribut napomáhá identifikovat konce vět u tokenu <enter>.

Atributy pro rozhodovací strom

- 1. aktuální token je vykřičník
- 2. aktuální token je otazník
- 3. aktuální token je uvozovka nebo pravá závorka
- 4. aktuální token je <enter>
- 5. další token není (aktuální token je poslední)
- 6. další token je <f cap>
- 7. další token je <f upper> nebo <f mixed>
- 8. další token je <D>
- 9. druhý další token je <d>[.!?)"]
- 10. druhý další token je <f num>
- 11. druhý předchozí token je <f num>
- 12. druhý předchozí token je <f abbr>, který určitě neoznačuje konec věty
- 13. druhý předchozí token je <f abbr>, který může končit větu
- 14. druhý předchozí token je <f abbr>, který však je identický apelativu
- 15. předchozí token je <D>
- 16. předchozí token je <d>[.!?)"]

Zejména v kódování UTF–8 existuje více druhů uvozovek. Proto jsou všechny rozpoznávané uvozovky a pravé závorky uloženy v souboru *quotes.dat*, který lze modifikovat.

Rozeznávání zkratek

Nejzajímavějším případem je však rozeznávání zkratek. Předně je nutné si uvědomit, že krátkých zkratek (jedno a dvojpísmenných) je velmi mnoho, a lepší než je všechny hledat, bude na problém nahlédnout z druhé strany a sepsat všechna jedno a dvojpísmenná česká slova, která mohou zakončovat větu. Doplněk množiny těchto slov na množině všech jedno a dvojpísmenných znakových řetězců nám poskytne seznam těchto zkratek. Delší zkratky budeme dále muset rozdělit do několika skupin.

- První skupinou jsou zkratky, které mohou být uprostřed věty a neoznačovat konec věty, ale mohou stát i na konci věty, kde bude tečka za zkratkou plnit dvojí roli – grafického atributu zkratky a konce věty. Mezi tyto zkratky řadíme například: **apod., atd., atp., DrSc., mld.**

- Další speciální skupinou zkratek budou slova, která jsou víceznačná – mohou značit zkratku, anebo též české apelativum umístěné na konci věty. Například: **mat., max., níž., předl., soch., nám.**

- Do poslední skupiny zkratek zařadíme všechny zbylé, tj. slova, která následována tečkou jsou vždy zkratkami a nemohou stát na konci věty. Například: **čes., exp., chem., labor., zejm., zříc.**

Příklad hodnoty atributů a klasifikace (zvýrazněna tučně) u tokenů, za kterými uvažujeme o konci věty:

<f cap>Slavia
<f>se
<f>na
<f num>2
<D>
<d>. ## 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 ***0***
<f>místo
<f>v
<f>lize
<f>musí
<f>probudit
<enter> ## 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 ***1***
<f cap>V
<f>dalším

Algoritmus trénování rozhodovacího stromu – ID3

Ke trénování rozhodovacího stromu byl použit ID3 algoritmus vyvinutý Rossem Quilanem v roce 1983. Základní myšlenkou tohoto algoritmu je konstruování výsledného stromu metodou odshora dolů, kdy v každém kroku hledáme ze všech atributů ten, který je nejvíce užitečný pro klasifikaci dané množiny. Abychom mohli tento atribut najít, musíme zvolit metriku zisku informace, kterou budeme poměřovat. Aby se při každé klasifikaci prováděl minimální počet rozhodování, budeme se snažit o minimalizaci počtu porovnávaných atributů (tj. hloubky stromu). Potřebujeme tedy takové dělení množiny dat, které nám poskytne co nejvíce vyvážené rozdělení.

Abychom mohli definovat metriku, musíme nejdříve objasnit pojem entropie. Ten můžeme chápat jako míru homogenity množiny dat. Entropii množiny S s frekvencemi klasifikací P(i) spočteme následovně:

$$Entropy(S) = -\sum_{i=1}^{n} P(i) * \log_2 P(i)$$

Atribut, který bude nejvhodnější pro rozdělení množiny dat, je ten, jenž nám poskytne největší redukci entropie vztaženou k rozdělení množiny podle daného atributu. Informaci získanou z množiny S za pomoci atributu A vypočítáme takto:

$$G(S, A) = Entropy(S) - \sum_{v=1}^{n}(|S_v|/|S|) * Entropy(S_v)$$

kde S_v značí podmnožinu S s hodnotou atributu A rovnou V.

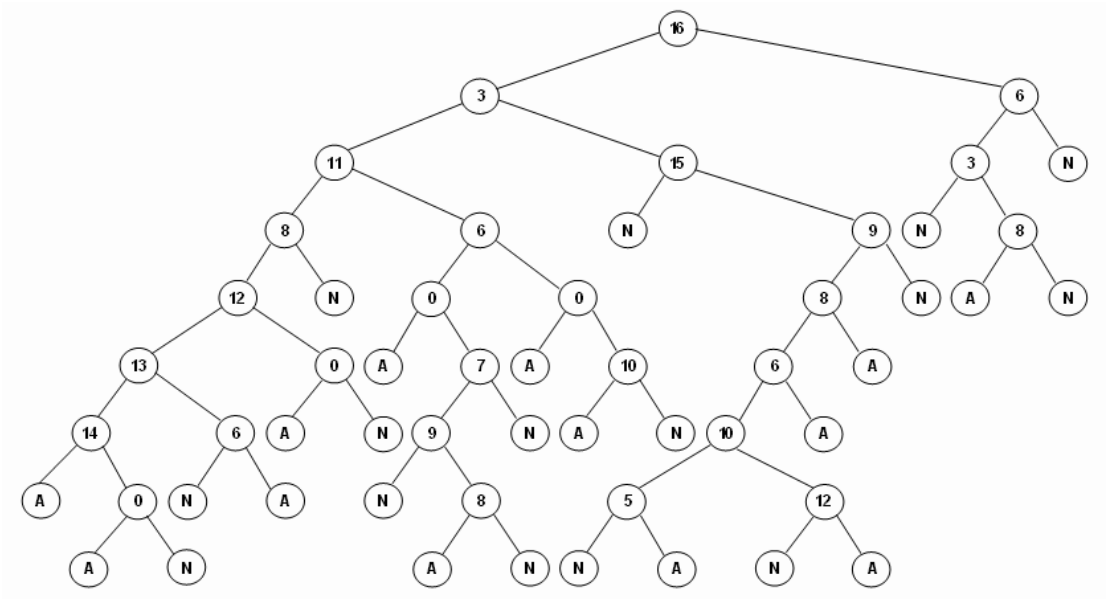
V každém kroku tedy vybereme atribut s největším ziskem informace z atributů, které jsme ještě na cestě z kořene do daného uzlu nepoužili. Myšlenku implementace ID3 algoritmu ukazuje tento zjednodušený kód:

```
ID3 ( trénovací množina S, množina atributů A, hodnoty atributů V)
return rozhodovací strom
{
if (S je prázdná) then return null;
N := new uzel;
if (A je prázdná) then označ N nejběžnější hodnotou klasifikace
else if (všechny instance S mají stejnou klasifikaci) then označ N touto klasifikací
    else {
        vypočítej zisk informace pro každý ze zbývajících atributů,
        najdi atribut A s maximální G(S,A);
        if (maximální G(S,A) < 0) then označ N nejběžnější klasifikací z S
        else {
            označ N atributem A;
```

```
                    foreach hodnotu V atributu A do {
                        rekurzivně zavolej ID3 algoritmus
                        N1 := (S_A|V, A, V), kde S_A|V, je podmnožina S,
                        kde atribut A má hodnotu V;
                        if (N1 != null) then vytvoř odkaz z N do N1 označený V;
                    }
                }
            }
return N;
}
```

Výsledný rozhodovací strom

Ke trénování rozhodovacího stromu bylo použito testovacích dat o celkové velikosti 3984 případů, která byla ručně klasifikována. Majoritní část testovacích dat pochází z internetových článků Lidových novin (3947 případů), zbylých 37 případů bylo k trénovacím datům přidáno po evaluaci prvotního rozhodovacího stromu. Výsledný rozhodovací strom má následující podobu:



5. 1 Výsledný rozhodovací strom

Uzly označené čísly 0–16 označují testovaný atribut, levá větev znamená nesplnění podmínky, pravá její splnění.

List označený A značí konec věty, list označený N označuje místo, které jako konec věty neoznačíme.

Vyhodnocení úspěšnosti segmentace

Z 261 544 dokumentů stažených z Internetu v roce 2003 z 9 různých zpravodajských serverů jsme náhodně vybrali 1 547 dokumentů (stejných jako u identifikace jazyka). Ty jsme nechali nasegmentovat. Z nich jsme extrahovali všechny pozice, které by mohly být považovány za konce vět (těch bylo 87 180). Z tohoto množství jsme opět udělali náhodný výběr a získali 1 001 pozic, které jsme ručně ověřili, zda se opravdu jedná o konce vět. V testu bylo chybně označeno 16 případů. Úspěšnost vyhledání konců vět na testovaných datech je tedy 98,40%.

Analýza chybně ohodnocených případů

Nejvíce chyb se vyskytuje ve sportovních výsledcích. Je-li číslo následováno bez mezery tečkou a následující slovo začíná velkým písmenem, tak se podle rozhodovacího stromu jedná o začátek nové věty.

Příklad:
...26. Tomík (Žemlička), 45. Havelka (Hnát, Netík), 51. <s> Novák (Bělohlav, Nedvěd), 52. Král (Chabada, Kašpařík)...

Tento případ je totiž zaměnitelný s výskytem čísla na konci věty, který rozhodovací strom ohodnotí správně. Bez další hlubší lingvistické analýzy tento problém nevyřešíme a s tímto chybným ohodnocením musíme počítat.

Další chybou je označení poslední tečky z elipsy uprostřed věty za její ukončení.

Příklad:
...k článku: NÁZOR: Možnost lacino koupit stíhačky Ota Ulč ... <s> na prostoře 2.600 akrů vzniklo pozoruhodné parkoviště a část...

Tento případ nastává kvůli mnoha případům z trénovacích dat, kde za tečkou ukončující větu následuje pomocný token <enter>. Za nositele ukončení věty byla v tomto případě zvolena tečka, která má poté stejné ohodnocení atributů jako poslední tečka z elipsy. Tento případ bychom mohli vyřešit přidáním atributu: je následující token <enter>?, který by tyto případy rozlišil.

Poslední častou chybou je nerozlišování případu novinářské signatury od obvyklého zakončení závorky uprostřed věty následované slovem začínajícím velkým písmenem. Novinářská signatura je správně označena za konec věty, ale druhý případ taktéž, což je však chyba. Mezi těmito jevy nemůžeme bez hlubší lingvistické analýzy rozhodnout stejně jako u případu sportovních výsledků.

Příklad:
...Zpravodajství / 26. února 2003 / (mr, čtk) <s> Tůma: V kuklách jsme do banky nemohli...

...Důvtipná otázka: "Kdo řekl, Být, či nebýt?, a) <s> Julius Caesar, b) Hamlet, c) Klement Gottwald...

Kapitola 6

Závěr

Tato bakalářská práce se zabývala kompletním předzpracováním textu s důrazem na texty získané z Internetu. Zpracování jsme rozdělili do těchto kroků:

- určení znakové sady,
- čištění dokumentu,
- rozpoznání jazyka,
- rozdělení textu na textová slova (tokeny),
- určení začátků a konců vět.

Při rozpoznávání jazyka jsme vycházeli z metod jiných autorů (pravděpodobnost trigramů, porovnání nejčetnějších n-gramů), které jsme upravili pro rozpoznávání pouze jediného jazyka a popsali metodu novou, založenou na podmíněné pravděpodobnosti trigramů. Segmentace textu byla vyřešena vytvořením rozhodovacího stromu z vhodně zvolených vlastností okolí znaků, které můžeme za konce vět považovat.

Výsledky práce splnily očekávání a její přínosy jsou zejména tyto:

1. Rozebrání pěti různých metod pro rozpoznání jazyka. Diskutovali jsme jejich vhodnost pro rozlišení mezi českým a nečeským textem, dvě z nich implementovali a vyhodnotili jejich úspěšnost.

2. Definování tří rozdílných typů zkratek pro účely určování konců vět a vytvoření jejich seznamů o délce 14, 91 a 752 zkratek.

3. Stanovení 17 vlastností kontextu znaků, za kterými se vyskytují konce vět. Z těchto vlastností jsme na ručně označených trénovacích datech o celkové velikosti 3984 případů vytvořili rozhodovací strom, který konce vět klasifikuje. Výsledný rozhodovací strom jsme aplikovali na testovací data a vyhodnotili jeho úspěšnost.

4. Byla vytvořena variabilní aplikace, která umožňuje komfortní a komplexní předzpracování textů.

Předzpracování textu v šíři, která je v práci prezentována, jsme se věnovali jako první. Předpokládáme, že vytvořená aplikace bude používána nejen pro tvorbu korpusů z Internetu, ale i pro usnadnění každodenní práce matematických lingvistů.

Literatura

[1] The Prague Dependency Treebank 1.0, 1996, http://ufal.mff.cuni.cz/pdt

[2] Hajič Jan: *Statistické modelování a automatická analýza přirozeného jazyka (morfologie, syntax, překlad)*, ÚFAL & CKL MFF UK

[3] Beesley R. Kenneth: *Language Identifier: A Computer Program for Automatic Natural-Language Identifikation of On-line Text*, Languages at Crossroads: Proceedings of the 29th Annual Conference of the American Translators Association, 1988

[4] Cavnar B. William, Trenkle M. John: *N-Gram Based Text Categorization*, In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, str. 161–169, 11–13. dubna 1994

[5] Ústav Českého národního korpusu FF UK: *Český národní korpus – SYN2000*, 2000.

[6] Ústav Českého národního korpusu FF UK: *Český národní korpus – SYN2005*, 2005.

[7] Sborník přednášek z 1. sjezdu československých profesorů filozofie, filologie a historie, 3. – 7. dubna 1929

[8] Specifikace HTML 4.01, 1999, http://www.w3.org/TR/html4/

Přílohy

Obsah přiloženého CD

evaluation_langid	
files	*soubory použité k vyhodnocení identifikace jazyka*
evaluation.ods	*přehled vyhodnocení identifikace jazyka*
evaluation_segmentation	
evaluation.con	*přehled vyhodnocení segmentace*
ID3	
id3.cpp	*ID3 algoritmus*
data.id3	*data pro ID3 algoritmus*
training_data_addition.seg	*trénovací data – dodatek ve tvaru segmentace*
training_data_base.seg	*trénovací data – základ ve tvaru segmentace*
training_data_addition.txt	*trénovací data – dodatek v plaintextu*
training_data_base.txt	*trénovací data – základ v plaintextu*
langid_1	*testovací soubory pro identifikaci jazyka*
langid_2	*testovací soubory pro identifikaci jazyka*
abbr_may_s.dat	*seznam zkratek, které mohou končit větu*
abbr_negative_2.dat	*seznam slov délky 2, která mohou ukončovat větu*
abbr_not_s.dat	*seznam zkratek, které nemohou končit větu*
abbr=apel.dat	*seznam homonymních zkratek*
letters.dat	*výčet rozpoznávaných písmen abecedy*
quotes.dat	*výčet rozpoznávaných uvozovek a závorek*
__cze__.lm	*jazykový model pro metodu E)*
cze_top400.lm	*jazykový model pro metodu C)*
textseg.pl	*hlavní program*
Cleaner.pm	*modul pro čištění dokumentu*
Concordance.pm	*modul pro konkordanci*
EncoDetect.pm	*modul pro detekci znakové sady*
LangID1.pm	*model prorozpoznávání jazyka, metoda C)*
LangID2.pm	*model prorozpoznávání jazyka, metoda E)*
Segmenter.pm	*modul pro segmentaci*
Tokenizer.pm	*modul pro tokenizaci*
readme.txt	*nápověda*

Frekvence nejčetnějších n-gramů z korpusů SYN2000 a SYN2005

Unigramy		Bigramy		Trigramy	
_	0,170642	e_	0,019291	_po	0,006100
o	0,067986	_p	0,017874	ní_	0,005368
e	0,065888	a_	0,017358	_a_	0,004892
a	0,057554	_s	0,015931	_ne	0,004724
n	0,053516	o_	0,013356	_pr	0,004671
t	0,045069	_n	0,013221	_na	0,004603
s	0,038570	_v	0,013061	_je	0,004408
i	0,037621	í_	0,012338	ch_	0,004361
l	0,037417	st	0,011148	_se	0,004332
v	0,035747	u_	0,010832	na_	0,003992
k	0,031127	i_	0,010607	_př	0,003766
r	0,031003	_j	0,009146	se_	0,003744
d	0,030641	y_	0,008824	_do	0,003544
m	0,028140	_z	0,008579	_v_	0,003248
p	0,027439	_t	0,008500	ou_	0,003220
u	0,026289	_a	0,008489	ho_	0,003092
í	0,024369	m_	0,008286	pro	0,002991
c	0,021064	po	0,008254	ost	0,002821
h	0,019550	ch	0,008177	em_	0,002805
á	0,017768	ní	0,008098	je_	0,002716
j	0,017420	_d	0,007951	la_	0,002578
z	0,017071	_k	0,007907	_st	0,002554
y	0,015350	ov	0,007830	_za	0,002539
b	0,014025	_m	0,007657	to_	0,002385
ě	0,013292	ro	0,007642	ce_	0,002346
ř	0,010102	na	0,007437	e_s	0,002276
é	0,009324	en	0,007361	ně_	0,002218
ž	0,009008	ne	0,006611	_to	0,002194
č	0,007967	_o	0,006604	že_	0,002089
ý	0,007476	je	0,006579	sta	0,001970
š	0,007002	le	0,006295	a_p	0,001939
ů	0,004219	se	0,006006	ova	0,001936
f	0,002627	to	0,005981	e_p	0,001872
g	0,002507	t_	0,005961	_vy	0,001872
ú	0,001084	te	0,005955	_by	0,001863
ň	0,000577	ho	0,005945	le_	0,001858
x	0,000561	ko	0,005886	pře	0,001834
w	0,000404	la	0,005840	e_v	0,001808
ť	0,000385	pr	0,005774	ých	0,001781
ó	0,000270	ou	0,005760	li_	0,001775
ď	0,000255	l_	0,005715	_ko	0,001775
q	0,000040	al	0,005629	ím_	0,001723
		é_	0,005521	_ro	0,001704
jiný	0,000268	od	0,005478	ter	0,001703
		ra	0,005394	_že	0,001666
		_b	0,005268	ky_	0,001665
		do	0,005261	e_n	0,001649
		li	0,005229	_ve	0,001648
		no	0,004864	né_	0,001622
		ta	0,004852	_ta	0,001612

Seznamy zkratek

Uvedené seznamy obsahují tří a vícepísmené zkratky. V seznamu III jsou všechny zkratky zapsány malými písmeny

I Zkratky, které mohou stát na konci věty

apod., atd., atp., atpod., CSc., DrSc., etc., hod., Ltd., mld., pod., spol., str.

II Zkratky, které jsou ekvivalentní jinému českému slovu

Admin., alt., arch., bal., bar., Brit., brok., bus., car., cit., část., dal., dam., dat., dep., depo., dia., dis., div., dok., dol., dom., dun., dup., Gen., gen., hod., hon., hovor., Ind., Ital., Jan., Jar., jun., kap., kat., Led., led., lic., mach., Man., man., mat., max., med., mil., min., Mor., neb., něm., neživ., níž., noc., nos., par., Pat., pat., Per., per., Plen., plk., plyš., pod., pop., pro., pros., prs., předl., přít., ret., sel., Sen., sen., soch., sov., sup., supr., Tel., term., tis., trs., typ., Val., val., vel., ves., vid., vil., viz., vol., zprav.

III Zkratky, které vždy označují konec věty

A

abbr., abs., abt., acc., accrd., accred., acct., ack., acq., acrd., acs., act., acv., add., adj., ads., advert., advt., affil., aftn., agcy., agd., agrd., agrt., agst., agt., agt., agy., akad., ala., alas., alg., amdt., amer., amph., amt., angl., ann., ans., ansd., ant., antr., app., appd., appln., appro., approx., appx., apr., aprx., arith., ariz., ark., arr., arrangt., arrd., art., asd., ashp., ass., assd., assmt., assn., assn., assoc., asst., astd., att., att., attn., aug., aus., auth., aux., avdp., ave., avg., avoir., avoird.,

B

back., barg., barl., bbl., bca., bdl., bds., benef., bfcy., bgt., bkg., bkge., bkpt., bkr., bkrpt., bks., blk., blvd., bot., bpc., brl., bro., bros., bsh., bvt., bvtd., bxs.,

C

calif., can., can., canc., cap., cap., caps., carr., cash., cat., cca., cert., cge., cgo., círk., citosl., cld., clk., cml., cmm., cncld., cnl., cog., col., coll., coll., collat., colo., com., comm., comml., comp., compt., con., cond., conn., cons., consgt., consid., cont., contd., contg., conv., cor., corp., cred., crt., csk., csv., ctf., ctge., ctl., cub., cul., cum., cum., cur., curr., curt., cvt., cwt.,

Č

čes., čís.,

D

dbk., dble., dcg., dcl., dcm., deb., dec., dec., decd., decim., def., deft., deg., del., del., deld., delv., dely., dem., dept., desp., destn., det., dft., dict., dicta., diff., dig., dim.,

dir., dis., disb., disc., disct., dist., divd., divs., dkg., dkl., dkm., dld., dlr., dlvd., dly., doc., doc., dop., dopol., doz., dpt., dram., dstn.,

E
encl., end., eng., entd., equ., esq., esqre., est., event., evid., evtl., exec., exch., exch., exor., exp., exrx., ext., extd.,

F
fac., fco., fdg., ffy., fid., fig., fin., fir., fla., fol., folg., forg., franc., frt., fur., fut., fwd., fwdd., fyz.,

G
gai., gal., gaz., gdn., gent., gov., govt., gtd., gtt.,

H
hab., hack., hhd., hist., hrs.,

CH
chaf., char., chd., chem., chg., chg., chge., chq., chtt.,

I
iii., ill., imp., inc., inc., ince., incl., indm., inf., ing., ing., inj., ins., insce., inst., instr., insur., int., inv., irred., irrev., ishld., iss., itg.,

J
jap., jedn., jnl., jnr., jnt., jos., jour., judr., junc.,

K
kans., knuž., kpt., krim., kupř.,

L
lab., lab., labor., lat., lbr., ldg., ldgs., lds., leg., liq., lir., lkg., lkge., long., lqdr.,

M
mag., machin., maint., mal., manfr., manuf., mar., marg., mark., mass., max., mdse., mem., meml., memo., mentd., messrs., měú., mfg., mfr., mga., mgmt., mgr., mcht., mchy., mid., mich., minn., misc., miss., mjr., mkr., mkt., mng., mngmt., mngr., mom., mont., mos., mrs., msd., mst., mtg., mtge., mtge., mthly., mudr., mun., mvdr.,

N
nábř., nakl., např., nar., nár., násl., nati., naut., nav., nazv., nebr., nedok., neg., negb., negl., nem., něm., neodb., neskl., nespis., nev., niz., nol., nom., nomin., non., npor., nprap.,

O
oblg., oblig., obst., obv., occ., occupon., odb., odd., odpol., odst., ofd., off., off., offic., okla., oph., opn., opt., ord., oreg., org., ors., outg., outstg., ozn.,

P
paeddr., para., parc., parcs., parlars., part., partn., pass., patd., pay., pay., payt., pce., pces., pckge., pckges., pcks., pcl., pcs., pen., pfd., pharm., pharmdr., phdr., phmr., pchs., písm., pkg., pkt., pkts., plf., pltf., podst., pol., pomn., popř., poř., posl., pozn., ppd., ppr., pps., ppt., prec., pref., pref., prem., pres., pres., prev., príd., prin., priv., prob., proc., prod., prof., prof., prop., provo., prox., prp., přek., překl., příč., příjm., přír., přísl., pts., pty., pub., pubn., pubr., pur., purst., pvt.,

Q
qlty., qnty., qrly., qto., qtr., qual., quar., quot.,

R
rcd., rcpt., rct., rds., rec., rec., recd., rect., red., redem., redemon., redisc., reexp., ref., reg., regd., reimp., reins., rem., remitt., reop., rep., repr., req., reqd., res., resp., rest., retd., rev., rev., revon., rež., rfg., riv., rkp., rly., rmdr., rndr., rom., rozk., rpt., rsdr., rst., rtn., rts., rts., rubr.,

Ř
řec., řidč., řím.,

S
sail., savs., scp., sec., sect., secy., senr., sep., seq., ser., sfce., sgd., shd., shipt., shpg., shpg., shpt., shr., shs., shtg., sch., sched., schr., sig., sign., sim., sir., sit., sitrep., sits., skg., sld., slg., soc., soc., spec., srov., srov., stč., stge., stk., stk., stn., strd., sub., suby., suc., success., suff., sufft., sund., supt., sur., surr., surv., survor., synd.,

Š
špan., švýc.,

T
tab., tal., tbe., tee., tel., tel., tem., temp., tenn., tens., terr., territ., testor., tex., tfer., tgm., thlic., thrin., tot., tpt., trans., trav., tre., treas., tree., trf., tripl., trlr., tzn., tzv.,

U
ugt., ull., ult., unabr., unan., unatt., unconf., ung., unkn., unm., unpd., usu.,

V
vac., valuon., vdr., veh., vii., viii., vlt., voy., vtg., vyd.,

W
warr., wash., wghhd., wght., whas., whatsr., whby., whf., whge., wholes., whr., whrin., whse., wis., witht., witned., wtd., wtr., wyo.,

X
xcp., xii., xiii., xiv., xix., xli., xlii., xliii., xliv., xlix., xlv., xlvi., xlvii., xlviii., xvi., xvii., xviii., xxi., xxii., xxiii., xxiv., xxix., xxv., xxvi., xxvii., xxviii., xxx., xxxi., xxxii., xxxiii., xxxiv., xxxix., xxxv., xxxvi., xxxvii., xxxviii.,

Y
yrs.,

Z
zájm., zák., zákl., záp., zast., zast., záv., zdrav., zejm., zkr., zříc., zvl.,

Ž
žst.