VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DOLOVÁNÍ DAT V PROSTŘEDÍ SOCIÁLNÍCH SÍTÍ

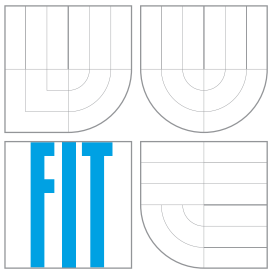
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JIŘÍ RAŠKA

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DOLOVÁNÍ DAT V PROSTŘEDÍ SOCIÁLNÍCH SÍTÍ

DATA MINING IN SOCIAL NETWORKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JIŘÍ RAŠKA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR BARTÍK, Ph.D.

BRNO 2013

Abstrakt

Tato práce se zabývala získáváním znalostí ze sociálních médií. Konkrétním cílem této práce bylo získávání názorů na úrovní rysů z uživatelských recenzí. V teoretické části byly uvedeny metody v procesu dolování názorů a zpracování přirozeného jazyka. Hlavní částí této práce byly návrh a implementace knihovny pro dolování názorů pomocí analyzátoru přirozeného jazyka Stanford Parser a lexikální databáze WordNet. Pro identifikaci rysů byla použita závislostní gramatika, implicitní rysy byly dolovány metodou CoAR a názory byly klasifikovány algoritmem typu učení s učitelem. Na závěr byly uvedeny experimenty vyhodnocující implementované řešení a příklady použití.

Abstract

This thesis deals with knowledge discovery from social media. This thesis is focused on feature based opinion mining from user reviews. In theoretical part were described methods of opinion mining and natural language processing. Main parts of this thesis were design and implementation of library for opinion mining based on Stanford Parser and lexicon WordNet. For feature identification was used dependency grammar, implicit features were mined with method CoAR and opinions were classified with supervised algorithm. Finally were given experiments with implemented library and examples of usage.

Klíčová slova

dolování dat, sociální média, zpracování přirozeného jazyka, dolování názorů, uživatelské recenze, Stanford Parser, WordNet, .NET

Keywords

data mining, social media, natural language processing, opinion mining, user reviews, Stanford Parser, WordNet, .NET

Citace

Jiří Raška: Dolování dat v prostředí sociálních sítí, diplomová práce, Brno, FIT VUT v Brně, 2013

Dolování dat v prostředí sociálních sítí

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Vladimíra Bartíka, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Raška
10. května 2013

Poděkování

Rád bych poděkoval vedoucímu mé diplomové práce Ing. Vladimíru Bartíkovi, Ph.D. za pomoc a ochotu při konzultacích. Rád bych také poděkoval mým blízkým za podporu.

Obsah

Kapitola 1

Úvod

S rozmachem sociálních médií a především s jeho všudypřítomným používáním v mnoha oblastech roste množství uživatelem vytvářeného obsahu. Tento obsah je zájmem mnoha odvětví jako psychologie, zábava, politika, obchod, zpravodajství a jiných. Aplikace dolování dat na sociální média přináší zajímavý pohled na lidské chování a interakci. Spojením dolování dat a sociálních médií můžeme lépe porozumět názorům, které lidé mají k určité problematice, identifikovat skupinu lidí a jejich vývoj v čase, naleznout vlivné osoby nebo doporučit zboží či služby jednotlivcům.

Cílem této práce je představit čtenáři oblasti sociálních médií, které jsou vhodné pro dolování. Následně bude vybrána jedna oblast a to *dolování názorů*. Čtenář bude uveden do této oblasti a budou zde zmíněny některé práce, které již podobný problém řešily. Následně bude navržena aplikace, která bude provádět dolování názorů na úrovni rysů.

Druhá kapitola, která následuje ihned po úvodu, zahrnuje základní pojmy, typy a oblasti sociálních médií pro dolování. Třetí kapitola seznámí čtenáře s problémem dolování názorů a to především z uživatelských recenzí zboží. Celý problém zde bude dekomponován na jednotlivé části, které budou rozebrány a jednotlivé přístupy srovnány. Ve čtvrté kapitole bude uvedena datová sada pro dolování. Pátá kapitola odhalí čtenáři techniky zpracování přirozeného jazyka, některé formalismy a implementace nástrojů pro zpracování přirozeného jazyka.

Šestá kapitola popíše lexikální databázi WordNet a především způsob jejího využití pro klasifikaci názorů. V sedmé kapitole bude navržena knihovna pro dolování názorů a v následující kapitole budou popsány některé zajímavé části její implementace. Devátá kapitola přibližuje čtenáři experimenty, které byly provedeny s implementovanou knihovnou, včetně jejich vyhodnocení.

Předposlední kapitola ukazuje jednoduchost použití knihovny včetně kódu, ukazuje i kompletní výstup a diskutuje praktické využití knihovny.

Kapitola 2

Dolování dat a sociální média

Tato kapitola uvede čtenáře do problematiky dolování dat a sociálních médií.

2.1 Dolování a získávání znalostí z dat

Dolování dat [5] (nebo také získávání znalostí) je proces extrakce zajímavých (netriviálních, skrytých, dříve neznámých a potenciálně užitečných) vzorů a znalostí z velkého objemu dat. Pro tentýž proces existuje řada alternativních jmen jako získávání znalostí z databází (*knowledge discovery in databases*), extrakce znalostí (*knowledge extraction*), datová archeologie (*data archeology*), bagrování dat (*data dredging*), sklízení informací (*information harvesting*) nebo *business intelligence*.

Pro ujasnění nyní shrňme, co dolování dat není. Dolování dat není provádění jednoduchých SQL dotazů, např. zjištěních počtu prodaného zboží v určitých prodejnách. Tedy musí splňovat výše uvedenou charakteristiku netriviálnosti. Dolování dat není ani používání deduktivních databází a systémů. A co tedy dolování dat je? Hledání *skrytých* a dříve *neznámých vzorů* a *znalostí*, které nejsou na první pohled vidět a musíme použít nějaký sofistikovaný postup pro jejich nalezení. Takto získaná data musí být potenciálně užitečná, tedy na základě významu těchto dat lze učinit vhodné rozhodnutí. Např. rozhodnutí o půjčce pro určitého klienta, doporučení zboží, rozpoznání potenciální hrozby atp.

Algoritmy pro dolování lze obecně rozdělit na *učení s učitelem (supervised)*, kde je klasifikační model vytvořen na základě trénovacích dat a potom je použit pro dolování. Typickými představiteli jsou *rozhodovací strom* nebo *naive bayes klasifikace*. Opakem je třída algoritmů nazývána *učení bez učitele (unsupervised)*, kde je klasifikační model vybudován na základě podobnosti a rozdílnosti dat. Do této třídy algoritmů například patří *K-means* nebo *hierarchický klastering*. Dělení na výše uvedené třídy algoritmů je provedeno na základě dat, z kterých hodláme dolovat znalosti. Pokud tato datová množina obsahuje třídní popis (class label) potom volíme první skupinu algoritmů, v opačném případě tu druhou.

Napůl cesty stojí algoritmy nazývané částečné učení s učitelem (*semi-supervised*), kde je klasifikační model vytvořen pomocí dat, která mají třídní popis. Data postrádající takový popis jsou potom použita pro vypilování klasifikačního modelu.

2.2 Sociální média

V [8] Andreas Kaplan a Michael Haenlein definovali sociální média jako skupinu internetových aplikací, které jsou vybudovány na ideologii a technologií Webu 2.0 a které umožňují

vytváření a výměnu uživatelem generovaného obsahu.

Sociální média se liší od těch tradičních tím, že umožňují snadný způsob komunikace mezi uživateli v nebývalém rozsahu nevídaném v tradičních médií. Popularita sociálních médií roste exponenciálně, což vede k vývoji sociálních sítí, blogů, mikro-blogů, lokálních sociálních sití, wiki ...

Pitam Gundecha a Huan Liu v [12] uvedli rozdělení sociálních medií do 9 kategorií na základě společných charakteristik. V následujících sekcích uvedu toto rozdělení doplněné o aktuální představitele.

2.2.1 Sociální sítě

Sociální sítě jsou webové služby, které umožňují jednotlivcům a skupinám propojení se svými přáteli a známými ze skutečného světa. Uživatelé interagují navzájem pomocí statusů, komentářů, sdílení médií, zpráv, ... Např.: Google+, Facebook, Myspace, LinkedIn.

2.2.2 Blogování

Blog je jako časopis na webu pro uživatele, kteří jsou nazýváni blogeři. Uživatelé přispívají textovým a multimediálním obsahem, jenž je chronologicky uspořádán a případně oštítkován tzv. tagy. Např.: Huffington post, Bussines insider, Engadget, Wordpress, Blogger.

2.2.3 Mikro-blogování

Mikro-blogy mají stejné použití jako blogy. Rozdílem je omezení obsahu. Například populární Twitter umožňuje publikovat obsah o maximální délce 140 znaků. Z představitelů bych jmenoval již zmíněný Twitter, dále Tumblr a Plurk.

2.2.4 Wiki

Wiki je kolaborativní editační prostředí, které umožňuje více uživatelům vytvářet webové stránky. Např.: Wikipedia, Wikitravel, Wikihow.

2.2.5 Sociální novinky

Sociální novinky jsou postaveny na sdílení a výběru nových zpráv a článků od komunity uživatelů, např.: Reddit, Digg, Slashdot.

2.2.6 Sociální záložky

Sociální záložky dovolují uživatelům skladovat, sdílet a organizovat odkazy na webové stránky, např.: Delicious, StumbleUpon.

2.2.7 Sdílení médií

Sdílení médií zastřešuje sdílení obrázků, videí, streamovaného videa a dalších medií. Např.: YouTube, Pinterest, flickr, UstreamTV, Instagram, Vine.

2.2.8 Názory, recenze a hodnocení

Základní funkcí těchto médií je sběr a publikace uživatelského obsahu ve formě subjektivních komentářů na stávající produkty, služby, zábavu, místa, atd. Např.: Epinions, Yelp, Cnet, IMDB, Polar.

2.2.9 Otázky a odpovědi

Tyto média poskytují platformu pro uživatele, kteří hledají radu. Ostatní uživatelé z komunity mohou na tyto otázky odpovědět nebo poskytnout vlastní názor. Tyto odpovědi jsou většinou posuzovány pomocí hodnocení. Např.: Yahoo answers, Stack Overflow.

2.3 Dolování v sociálních médií

Každý den je na sociálních médií vytvářeno obrovské množství uživatelem generovaného obsahu. Je velmi pravděpodobné, že se tento trend měnit nebude a proto je důležité jak pro producenty, konzumenty a především pro poskytovatele těchto služeb, aby takový obsah byli schopni jednak spravovat, ale také využít ve svůj prospěch.

Dolování v sociálních médií [2] je poměrně nová oblast ve srovnání s podobnými studiemi týkající se analýzy sociálních sítí. Nicméně aplikace využívající techniky dolování vyvinuté jak průmyslem tak i akademickou sférou jsou hojně používány v komerčním prostředí. Příkladem je Samepoint[1]. Samepoint je služba, která umožňuje dolovat a monitorovat sociální média a poskytuje zákazníkovi informace, jak je zboží nebo služba vnímána a diskutována v sociálních médií.

Pitam Gundecha a Huan Liu v [12] uvedli 6 problémů v sociálních médií, které lze řešit dolováním dat. Jsou to analýza komunity, dolování názorů a mínění, sociální doporučení, modelování vlivu, šíření a původ informací, ochrana osobních údajů a důvěra. Všech 6 oblastí ve stručnosti představím, ovšem větší pozornost bude věnována pouze jednomu tématu a to dolování názorů a mínění, proto bych zájemce o ostatní témata odkázal na již zmiňovaný článek [12] ve kterém najdou více informací a dostatek literatury, aby byli schopni do dané problematiky proniknout hlouběji.

2.3.1 Analýza komunity

Komunita je tvořena jednotlivci, kteří interagují mezi sebou častěji než s těmi, kteří nejsou součástí komunity. V závislosti na kontextu bývá komunita také označována jako skupina (group), shluk (cluster), soudržná podskupina (cohesive subgroup) nebo modul. Komunity, které se vyskytují v sociálních médií, jsou obecně děleny na implicitní a explicitní. Explicitní komunity jsou tvořeny tak, že uživatel se upíše k dané skupině. Naopak implicitní komunity jsou tvořeny přirozeně na základě interakce.

Problémy, které jsou v této oblasti řešeny, jsou utváření komunity, vývoj komunity a detekce komunity (především implicitní komunity).

2.3.2 Dolování názorů a mínění

Cílem je automatická extrakce názorů vyjádřených v uživatelsky generovaném obsahu. Takto získané znalosti potom mají široké využití, např. umožňují obchodníkům lépe porozumět, jak lidé vnímají nový produkt, značku, jakou reputaci má obchod atp.

[1] http://www.samepoint.biz/

Dolování názorů je velmi obtížné neboť přirozený jazyk, který je použit k tvorbě tohoto obsahu je víceznačný. Názor je tvořen třemi částmi: objekt, ke kterému je vyjádřen názor, samotný názor, který je vyjádřen na určitý objekt a držitel tohoto názoru.

Dolování v této oblasti lze rozdělit na tři podoblasti: dolování na úrovni dokumentu (document level), dolování na úrovni vět (sententce level) a dolování na úrovni rysů (feature level).

2.3.3 Sociální doporučení

Sociální doporučení je založeno na hypotéze, že lidé, kteří jsou sociálně propojeni, sdílí stejné nebo podobné zájmy, jsou snadno ovlivnitelní přáteli, věří a preferují doporučení právě od přátel než od někoho neznámého. Cílem je tak zvýšit kvalitu doporučení a ulehčit problém přehlcení informacemi, který je pro dnešní dobu typický. Příkladem může být doporučení knih na základě čtenářského deníku přátel.

2.3.4 Modelování vlivu

Modelování vlivu se snaží rozeznat, zda je sociální médium řízeno vlivem nebo homophily[2], což je tendence jednotlivců sdružovat se s podobnými. Pokud by v sociálním médiu převažovalo řízení vlivem, potom by bylo možné daného vlivného jednotlivce identifikovat a podněcovat ho, aby propagoval určité služby či produkty. Cílem tak může být nalezení jednotlivců, kteří mají vliv na nejvyšší počet ostatních uživatelů.

2.3.5 Šíření a původ informací

V této oblasti se zkoumají různé modely (kaskádový, prahový ...) šíření informací. Využití potom spočívá v aplikaci těchto modelů k analyzování šíření počítačových virů, zvěstí a nemocí. Hlavní úlohy této oblasti jsou: jak je informace šířena v sociálních médií, jak je ovlivňována a jaké jsou věrohodné zdroje.

2.3.6 Ochrana osobních údajů a důvěra

Uživatel sociálních sítí by chtěl mít tolik přátel a sdílet tolik informací, jak jen je to možné. Na druhé straně by si chtěl udržet své soukromí. Zde tak vznikají nové výzvy, jak ochránit osobní údaje uživatelů a zároveň jim dodat službu, kterou chtějí.

V této oblasti se výzkum věnuje takovým problémům jako mechanismy pro lepší správu a sdílení obsahu mezi uživateli nebo poukazuje na zranitelnosti soukromí uživatelů v současných sociálních sítí.

[2] http://en.wikipedia.org/wiki/Homophily

Kapitola 3

Dolování názorů a mínění

Tato kapitola je věnována problematice dolování názorů a budou zde uvedeny přístupy, techniky a algoritmy na extrakci názorů a explicitních i implicitních rysů.

3.1 Motivace

Rozvoj internetu a služeb, které umožňují uživatelům sdílet jejich názory a psát recenze, způsobuje neustálý nárust uživatelsky vytvářených recenzí. Např. na IMDB[1] pár dnů po filmové premiéře mají filmy desítky uživatelských recenzí. Mimoto mnoho recenzí je dlouhých a jen pár vět obsahuje názory vyjádřené k recenzovanému objektu. Procházení takové spousty většinou duplicitních názorů je pro uživatele zdlouhavé a neefektivní, proto se výzkumníci již nějakou dobu zabývají tím, jak automaticky z desítek, stovek či tisíců recenzí vyextrahovat názory, které tam uživatel hledá.

Takovéto nástroje pro dolování přispívají ke zvýšení user experience[2] na jedné straně. Na druhé straně pomohou prodejcům a producentům lépe sledovat názory zákazníků či uživatelů.

3.2 Úvod

Dolování názorů lze provádět na úrovni dokumentů, vět nebo rysů. V této práci se zaměřím na dolování názorů na úrovni rysů. Dolování bude prováděno na uživatelských recenzí, které budou psány v anglickém jazyce.

Recenze (review) je uživatelem vytvořený obsah textového charakteru, který je předmětem dolování. Soubor recenzí je potom nazýván korpus. Recenze je tvořena nejméně jednou větou, která obsahuje rys a názor.

Rys (feature) nebo také vlastnost je to k čemu je vyjádřen názor. Rysy lze dělit na explicitní a implicitní. Mějme recenzi:

"While light, it will not easily fit in pockets."

V této recenzi zákazník vyjadřuje svůj názor k *velikosti* pravděpodobně nějakého mobilního zařízení. I přesto, že se v textu implicitní rys *velikost* nevyskytuje, zcela jistě si ho tak v kontextu doplníme. Na druhé straně jsou rysy explicitní, které jsou v textu zmíněny a jejichž extrakce je o poznání jednodušší.

[1] http://www.imdb.com/

[2] http://en.wikipedia.org/wiki/User_experience

Názor (opinion) je vyjádřen k nějakému rysu. Obvykle můžeme určit tzv. polaritu názoru, tedy jestli se jedná o pozitivní nebo negativní názor. Případně lze názory klasifikovat do více tříd než pouze do dvou.

Názorovou větu (opinion sentence) Hu a Liu v [6] definovali následovně:

"Pokud věta obsahuje jeden nebo více rysů a zároveň jeden nebo více slov vyjadřující názor, potom je tato věta nazývána názorová věta."

3.3 Procesy v dolování názorů

Dolování názorů na úrovni rysů lze obecně rozdělit na tři částí: (1) identifikace rysů a názorů, (2) sémantická analýza a (3) shrnutí názorů. V praxi může být dělení jemnější v závislosti na tom, jaké metody chceme implementovat. V následujících podsekcích představím čtenáři různé metody v procesech dolování od samotného zpracování recenzí pomocí nástrojů NLP, přes identifikaci explicitních a implicitních rysů, ořezání kandidátních rysů až po identifikaci názorů a jejich klasifikaci.

Aktuální srovnání současného stavu (2012) v oblasti dolování názorů je diskutováno v [14], kde je v rámci možností poskytnuto srovnání jednotlivých výzkumných prací, jaké byly použity techniky dolování, jakých bylo dosaženo výsledků a jaké jsou případné nedostatky, které je potřeba do budoucna řešit.

3.3.1 Identifikace explicitních rysů

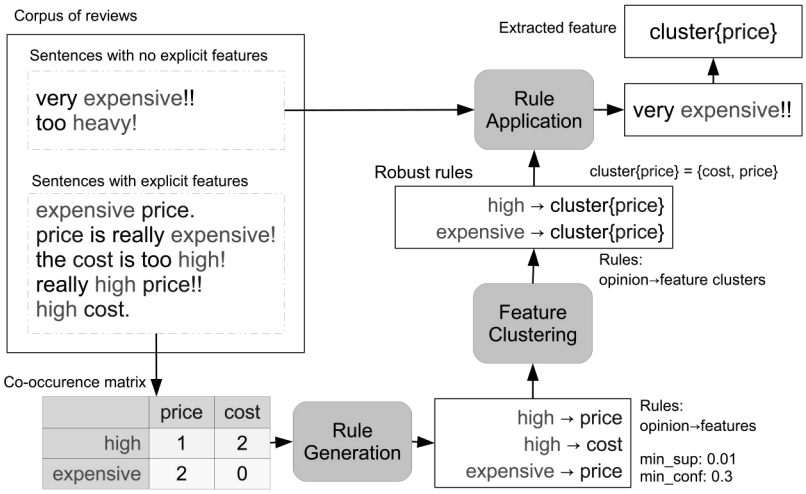
Jde o proces hledání klíčových slov, která vystihují vlastnosti recenzovaného objektu. Typickým přístupem pro identifikaci rysů a názorů je použití Part-of-Speech (PoS) [11] značkovače. Identifikace rysů je při použití PoS docela efektivní, ale velmi časově náročná. Pro snížení časové náročnosti a v některých případech i žádanou vlastností je použití vstupního seznamu hledaných rysů k určité třídě recenzovaných objektů. PoS a další nástroje pro zpracování přirozeného jazyka budou rozebrány v kapitole 5.

Hu a Liu v [6] představili kompletní systém pro dolování názoru. Hlavními dvěma fázemi jsou extrakce rysů a identifikace polarity názorů pro konkrétní produkt. Dolování rysů je podrobněji popsáno v [7]. Autoři předpokládají, že rysy jsou pouze podstatná jména a jmenné fráze. Pro zařazení slov do jednotlivých kategoriích používají PoS. Frekventované rysy jsou extrahovány pomocí asociačních pravidel a apriori. Součástí je i vypořádání se s nesouvisející rysy a provedení dvojího ořezání: ořezání frázových rysů a odstranění redundantních rysů.

V [13] Popescu a Etzioni představují OPINE, systém pro extrakci informací. Tento systém dosahuje výrazně vyšší přesnosti (o 22%) pro extrakci rysů než [6] od Hu a Liu. Dalšími odlišnostmi jsou použití inicializačních rysů pro každou třídu produktů, dolování implicitních rysů nebo hodnocení názorů rozsáhlejší škálou než pouze binární. Autoři uvádí, že dosažení vyšší přesnosti je způsobeno použitím mechanismu ohodnocení rysů a Web PMI [3] statistiky.

Pro přesnější rozpoznání slov nebo frází reprezentující rysy nebo názory se používá relace závislosti [10], což je asymetrická, binární relace mezi slovem nazývaným *hlava* a jiným slovem zvaným *modifikátor*. Slovo ve větě může mít více modifikátorů, ale každé slovo může modifikovat maximálně jedno slovo. Kořenové slovo závislostního stromu nemodifikuje

[3] http://en.wikipedia.org/wiki/Pointwise_mutual_information



Obrázek 3.1: Jak coAR pracuje, převzato z [3]

žádné slovo, je také nazýváno hlavou věty, závislostní gramatika je podrobněji řešena v části 5.2.

Za rysy se považují větné členy označené jako podstatné jméno nebo jmenná fráze. Závislostní vztahy mezi touto dvojicí jsou zkoumány a na jejich základě jsou identifikovány rysy. V [4] autoři používají 3 typy závislostních vztahů. Prvním je *verb-object* (VOB) a říká, že pokud podstatné jméno závisí na jiné komponentě se vtahem VOB potom toto podstatné jméno je kandidátem na rys. Dalšími závislostmi jsou *subject-verb* (SBV) a *head word* (HED). Kandidátní rysy jsou rozpoznány tak, že jsou nalezeny všechna podstatná jména a následně sekvenčně aplikována pravidla VOB, SBV a HED. Pokud je jedno z pravidel splněno potom je podstatné jméno přidáno do množiny kandidátních rysů.

3.3.2 Identifikace implicitních rysů

Většina prací zabývajících se dolováním názoru opomíjí implicitní rysy, v roce 2011 Hai, Chang a Kim v [3] představili jednoduchý způsob, jak extrahovat implicitní rysy tedy ty, které nejsou v recenzi zmíněny. Autoři svůj přístup nazývají *Co-occurence Association Rule Mining* a zavedli pro něj zkratku coAR.

Schéma činnosti metody coAR najdete na obrázku 3.1, kde jsou názorně ukázány jednotlivé komponenty: vstup, výstup a tok dat. CoAR pracuje ve dvou fázích generování pravidel a aplikování pravidel. V první fázi vytvoříme dvě skupiny. V jedné budou slova reprezentující rysy a v druhé názory. Tato extrakce je vykonána nad recenzemi s explicitními rysy. Následně je vytvořena matice souběžnosti (co-occurrence matrix) pro názory a rysy. Pro každý názor je vytvořeno asociační pravidlo. Na základě nastavené minimální podpory a důvěry jsou slabá pravidla ořezána.

V druhé fázi coAR dojde ke shlukování konsekventů (explicitních rysů) pravidel do shluků pro následné vygenerování více robustních pravidel. V příkladu je tedy uvedeno, že hledáme pro názor *very expensive* příslušný rys a to tak, že procházíme robustní pravidla a vybereme rys, který toto pravidlo splňuje.

Ve srovnání s ostatními metodami jako PMI nebo LRT, coAR dosahuje vyšší přesnosti. Ovšem stále je co zlepšovat, autoři poukazují například na automatické rozpoznání nejle-

pšího prahu podpory a spolehlivosti.

3.3.3 Ořezání kandidátních rysů

V textu identifikujeme velké množství rysů, bohužel ne všechny jsou ty rysy, které hledáme. Proto je nutné provést ořezání těchto kandidátních rysů a získat tak rysy relevantní. Ve většině případů jde o přidělení váhy každému rysu a následně odstranění těch rysů, které nesplňují minimální práh. V této práci představím dvě metody pro výpočet váhy každého rysu.

Jedním z nejjednodušších přístupů jak získat relevantní rysy je vypočítat frekvenci výskytu jednotlivých rysů, tato metoda je označována TF (term frequency). Čím vyšší frekvence výskytu tím více relevantní rys je. Frekvence výskytu se vypočítá jako podíl počtu recenzí s výskytem alespoň jednoho daného rysu s počtem všech recenzí. Potom pro minimální práh např. 5%, budou odstraněny ty rysy, které se vyskytují v méně než 5% recenzí. Slabinou této metody je to, že neřeší jak je daná recenze dlouhá a tudíž jak je rys relevantní v dané recenzi. Tato metoda přiřadí stejnou váhu rysu, který se nachází v recenzi o dvou větách i recenzi, která má několik odstavců a obsahuje i několik dalších rysů.

Pokročilejší je váhovací metoda TF-IDF, která spolu s frekvencí rysů počítá také s inverzní frekvencí (IDF). Metoda IDF říká, že čím častěji se rys vyskytuje v recenzi, tím méně je důležitý. IDF se vypočítá podle vzorce 3.1, kde n je počet recenzí a k je počet recenzí v kterých se vyskytuje rys t.

$$IDF(t) = 1 + log(\frac{n}{k}) \tag{3.1}$$

Výsledná hodnota TF-IDF se získá součinem TF a IDF.

3.3.4 Identifikace názorů

Názory jsou většinou vyjádřeny v podobě přídavných jmen. Hu a Liu v [6] použili tento předpoklad a za názorová slova identifikují přídavná jména, která se vyskytují ve větách spolu s jedním nebo několika rysy.

Popescu a Etzioni v OPINE[13] počítají se skutečností, že názory se vyskytují v blízkosti rysů. Pro identifikaci tak používají relace závislosti mezi rysem a názorovým slovem.

V [4] autoři hojně používají závislostní gramatiky a jako názory identifikují kromě přídavných jmen také slovesa, např. like, hate. Pokud tak najdou přídavné jméno, které je v závislém nebo řídícím vztahu k rysu potom jej identifikují jako názorové slovo. Slovesa jsou identifikovány jako názory, pokud jsou v řídícím vztahu k rysu.

V [3] autoři poukazují, že ne všechna přídavná jména a slovesa jsou názorová slova a proto pokud názorové slovo závisí na jiném slovu s příslovečnou závislostí, potom by nemělo být identifikováno jako názorové slovo.

3.3.5 Sémantická analýza

Po získání frekventovaných rysů a názorů, je našim úkolem rozpoznat polaritu, směr neboli pocit názoru. Mějme například recenzi: *"Nokia sound system is good"*. *Sound system* je rys a *good* je pocit. Cílem sémantické analýzy je klasifikace těchto pocitů (názorů). Klasifikace je většinou pouze binární (kladná/záporná). Některé práce ovšem používají i širší stupnici hodnocení. Při použití binárního hodnocení, by pocity jako beautiful, awesome, good měli

pozitivní orientaci (kladnou). Naopak pocity disappointing, bad by měli negativní orientaci (zápornou).

V již zmíněném článku [6] Hu a Liu pro výpočet orientace pocitů používají techniku, kdy nejdříve manuálně vytvoří inicializační seznam s desítkami různých pocitů a jejich orientacemi. Následně pomocí lexikonu WordNet nechávají tuto databázi růst, tak že pro seznam vyextrahovaných pocitů (názorů) z prvního kroku hledají synonyma případně antonyma k pocitům z inicializačnímu seznamu, kde je již orientace známá.

V [13] autoři klasifikují názory do třech tříd (pozitivní, negativní, neutrální). Používají Relaxation Labeling, což je klasifikační technika učení bez učitele.

3.3.6 Shrnutí názorů

Závěrečným krokem je vytvoření výstupu. V běžných nástrojích dolování názorů se používají dva přístupy (shrnutí názorů a skóre rysů). Shrnutí názorů je generování zjednodušených vět, které zachycují podstatu věci získanou z recenzí. Skóre rysů identifikuje polaritu názoru, zda je pozitivní či negativní a výstupem je rys a jeho skóre. Tento přístup se používá ve většině nástrojích pro dolování názorů.

Kapitola 4

Datová sada uživatelských recenzí

Typickým prvkem sociálních médií je umožnit uživatelům pronést svůj názor k určité problematice. Ať už jsou to zpravodajské portály, sociální sítě nebo portály pro sdílení médií vždy je nabídnuto uživateli, aby daný objekt okomentoval vlastním názorem.

Takovéto sociální média ovšem nenabízí pouze textové hodnocení, ale i hodnocení formou, která je počítači bližší a to číselné hodnocení. Běžné je tak hodnocení na stupnici 1-5, 1-10 nebo 1-100, které může být provedeno na úrovni celého objektu nebo jeho vybraných částí. Například při hodnocení nákupu na internetovém aukčním portálu je vhodné oddělit hodnocení nákupu (komunikace s prodejcem, rychlost dodaní) od hodnocení koupeného zboží.

Recenze jsou psány lidmi různé výchovy, vzdělání, zkušeností a talentu, což se odráží na kvalitě psané recenze. Pro filtraci méně kvalitních recenzí se používají čtenáři těchto recenzí, kteří si sami volí, která recenze byla pro ně přínosem a která nikoliv. Takovéto hodnocení je většinou implementováno krátkým formulářem s otázkou: "Byla výše uvedená recenze pro vás užitečná?" a s odpověďmi "ano" a "ne". Zajímavostí je, že žádný z algoritmů a technik uvedených v této publikaci nevyužívá nějakého z výše uvedených atributů pro zvýšení přesnosti.

V následujících sekcích představím 2 zdroje recenzí pro otestování výsledné implementace programu pro dolování názorů.

4.1 Zappos - recenze produktů

Zappos[1] je populární americký obchod prodávající obuv a oblečení. Kromě textové recenze může uživatel sdělit číselné hodnocení na celkový dojem, komfort a styl ve škále 1-5. Pro obuv lze provést i hodnocení velikosti, šířky a klenby. Zappos zveřejňuje svou aktuální nabídku i uživatelské recenze prostřednictvím API[2]. V příkladu 4.1 je ukázán výstup pro přesnou ilustraci, jaká data jsou vlastně k dispozici.

[1] http://www.zappos.com/

[2] http://en.wikipedia.org/wiki/Application_programming_interface

```
{
  "statusCode": "200",
  "page": 1,
  "offset": 0,
  "reviews": [
    {
        "id": "2255880",
        "date": "05/19/2010 07:26 AM",
        "name": "",
        "location": "Washington, DC",
        "otherShoes": "",
        "summary": "These slippers are very comfy! The only issue I have is
           that my foot slips off of the memory foam bottom sometimes (it
           is almost like an insert) and I stumble.",
        "shoeSize": "FULL_SIZE_SMALLER",
        "shoeWidth": "TRUE",
        "shoeArch": "NONE",
        "overallRating": "4",
        "comfortRating": "5",
        "lookRating": "5"
    }
  ]
}
```

Listing 4.1: Recenze v JSON z Zappos API

4.2 IMDB - recenze filmů

IMDB[3] (Internet Movie Database) je největší internetová databáze filmů, která je stejně jako výše uvedený Zappos vlastněna Amazonem[4]. IMDB kromě dvou milionu filmových titulů (včetně TV seriálů) obsahuje spoustu uživatelských recenzí. Ačkoliv IMDB nenabízí veřejné API, má soukromé API, které využívá jejich mobilní aplikace. Výstup z tohoto API je zobrazen na příkladu 4.2.

IMDB umožňuje hodnocení recenzí mezi čtenáři navzájem. Můžeme tak vidět atributy *user_score* a *user_score_count* , které říkají kolik uživatelů z kolika bylo spokojeno s touto recenzí.

Pro recenze na IMDB je typické, že jsou podstatně delší (až tisíce znaků) než recenze na Zappos. V příkladu je ukázána jedna z kratších recenzí.

[3] http://www.imdb.com/
[4] http://www.amazon.com/

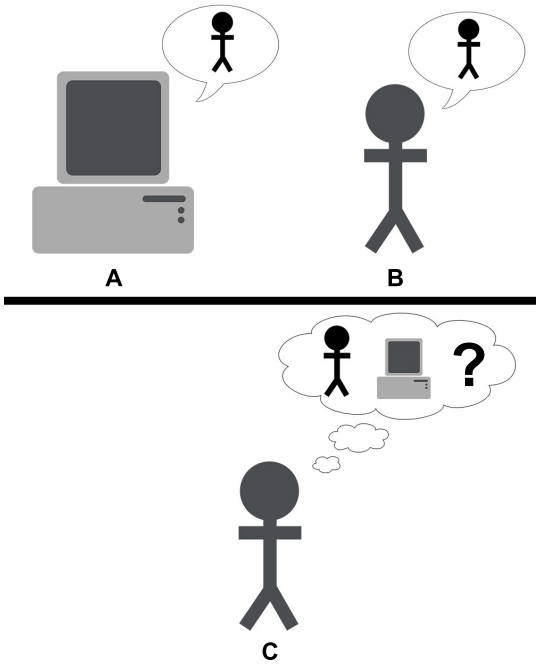
```
{
    "exp": 1356819392,
    "@meta": {
        "serverTimeMs": 39,
        "requestId": "0B3HNAAY2QDHXRVKF3FR"
    },
    "data": {
        "user_comments": [
            {
                "user_score": 125,
                "summary": "I enjoyed everything about this movie.",
                "user_location": "United States",
                "text": "My initial reaction is that this film is the best
                    romantic comedy that I've seen in years. The genre has
                    been pretty devoid of quality lately. So, I don't know
                    if that plays a part or not and I really don't care at
                    this point. I enjoyed everything about this movie. It
                    has tremendous heart and charisma and it's so very easy
                    to get caught up in to the lives of these characters. A
                    certain degree of patience is required while viewing
                    because some secondary characters that feel unnecessary
                    to the story are worth getting to know. Steve Carell's
                    character is the one everyone empathizes with and when
                    the movie shifts away from the \"A\" story you wonder
                    why and start to think that the \"B\" story is going to
                    be muddled or cliché or one to endure. Well, they're not
                     and everything comes together in a wonderful fashion.
                    The entire cast here is perfection. The overall message
                    may be one to debate but it doesn't matter because the
                    ride and this film are just so smart and so well done.",
                "date": "2011-08-04",
                "status": "G",
                "user_score_count": 156,
                "user_name": "Mill Coleman"
            }
        ],
        "total": 236,
        "tconst": "tt1570728",
        "title": "Crazy, Stupid, Love.",
        "type": "feature",
        "limit": 1,
        "year": "2011",
        "start": 1
    }
}
```


Listing 4.2: Recenze v JSON z IMDB Mobile API

Kapitola 5

Zpracování přirozeného jazyka

Tato kapitola uvede čtenáře do problému zpracování přirozeného jazyka. Ukáže dva přístupy analýzy včetně konkrétních implementacích ve volně dostupných knihovnách. Na závěr je ukázáno jak pomocí nástrojů pro zpracování přirozeného jazyka lze analyzovat uživatelské recenze a identifikovat v nich názory.



Obrázek 5.1: Ilustrace Turingova testu

5.1 Úvod

Zpracování přirozeného jazyka je obor na pomezí počítačové vědy, umělé inteligence a počítačové lingvistiky. Začátek tohoto vědního oboru sahá do 50. let 20. století, kdy Alan Turing publikoval dnes již známý článek *Computing Machinery and Intelligence* [16], kde představuje to, co je nazýváno *Turingů test* jako měřítko inteligence.

Turing tento problém popisuje na hře, kdy máme tři aktéry A,B,C viz obr. 5.1. Jeden z aktérů A a B je člověk a druhý stroj, který ovšem imituje člověka. Poslední aktér C

se snaží zjistit pomocí otázek a odpovědí, který z nich je stroj. Otázky a odpovědi jsou přenášeny přes textové kanály, takže výsledek nezávisí na schopnosti stroje vytvářet zvuk ani tak na správnosti odpovědi jako na podobnosti s typickou odpovědí člověka. Pokud aktér C není schopen rozhodnout, kdo z aktérů A a B je stroj, potom stroj úspěšně prošel *Turingovým testem*.

5.2 Závislostní gramatika

Závislostní gramatika [10] v anglické literatuře *Dependency Grammar* (dále jen DG) je alternativní gramatika k frázové strukturální gramatice (dále jen PSG). DG zachycuje vztahy mezi slovy ve větě na rozdíl od PSG, která sestavuje derivační strom věty s frázovými uzly.

Definice 5.2.1. *Frázová strukturální gramatika* (PSG) G je čtveřice $G = (N, T, P, S)$, kde

- N je konečná množina neterminálů
- T je konečná množina terminálů, $N \cap T = \emptyset$
- $P \subseteq (N \cup T)^* N (N \cup T)^* \times (N \cap T)^*$ je konečná relace, kde každé $(x, y) \in P$ je nazýváno pravidlo a je obvykle psáno ve tvaru $x \to y$
- $S \in N$ je startovací symbol

Základní myšlenkou závislostní gramatiky, je že syntaktická struktura věty se skládá z binárních relací mezi jednotlivými slovy. Pro slova v závislostní relaci platí, že jedno je hlava a druhé modifikátor. Hrana mezi těmito uzly je orientovaní ve směru od hlavy k modifikátoru, ale tato orientace se v různých literaturách může lišit. Také pojmenování uzlů relace se mezi autory používá několik:

- head - modifier
- parent - child
- governor (řídící) - dependent (podřízený)

Pro jasný výklad budu dále v textu používat pojmenování *řídící* a *podřízený*, případně zkráceně pouze *gov* a *dep*. Toto pojmenování jsem zvolil z toho důvodu, protože stejné pojmenování používá i v této práci používaný a později zmíněný analyzátor.

Obrázek 5.2: Schéma vztahu závislosti

Definic závislostních gramatik je více, pro úplnost zde uvádím definici 5.2.2 od pánů Hays (1964) a Gaifman (1965), která je uvedena v [1].

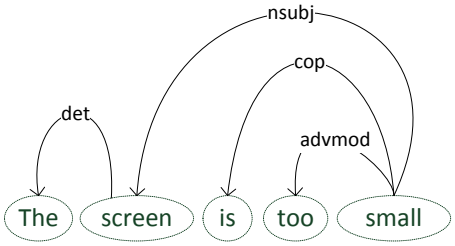
Definice 5.2.2. *Závislostní gramatika* (DG) G je čtveřice $G = (R, L, C, F)$, kde

- R je konečná množina pravidel závislostí nad pomocnými symboly C
- L je konečná množina terminálních symbolů (lexémů)

- C je konečná množina neterminálních pomocných symbolů (kategorie)
- F je funkce přiřazení ($F : L \rightarrow C$)

Relace závislosti R je binární, asymetrická, tj. jeden uzel je řídící (resp. rozvíjený) a druhý je podřízený nebo také závislý, acyklická a ireflexivní.

Příklad věty s vyznačením závislostí mezi jejími lexémy je zobrazen na obr. 5.3. V relaci závislosti platí, že každý uzel má pouze jeden řídící uzel (vyjma kořenového uzlu). Naopak každý uzel může mít jeden, více nebo žádný podřízený uzel. Všechny slova jsou spojena do grafu, který je acyklický a budeme jej nazývat strom závislostí.



Obrázek 5.3: Závislosti mezi lexémy věty

Kromě znalosti co na čem závisí, chceme také vědět jak závisí. Proto přiřazujeme závislostem štítky.

Vedle grafické podoby lze relace závislosti vyjadřovat i textově. Syntaxí je několik, já zde uvedu ty nejběžnější na příkladu relace mezi slovy *small* a *screen* z věty jejíž strom závislostí je na obr. 5.3:

- nsubj(small, screen)
- nsubj(small-5, screen-2)
- (screen NN [small] [nsubj])

V prvních dvou zápisech je první z dvou prvků v závorce *řídící* a druhý *podřízený* a před závorkou je název závislosti. První od druhého zápisu se liší pouze v přidání dalších metadat o pořadí lexému ve větě. Poslední zápis přidává metadata o slovní kategorii podřízeného slova. Řídící slovo a název závislosti jsou potom v hranatých závorkách.

V PSG je kořenový uzel derivačního stromu dán startovacím neterminálem gramatiky. V závislostní gramatice není žádný startovací neterminál. Pro kořenový uzel různí autoři používají různé notace. Např. Stanford Parser[1] používá abstraktní kořenový symbol pojmenovaný *ROOT*.

5.3 Analyzátory

Moderní nástroje pro zpracování přirozeného jazyka jsou založeny na strojovém učení, konkrétně na statistickém strojovém učení. Máme trénovací data a z nich chceme získat pravidla. V tomto případě trénovacími daty jsou texty v určitém jazyce doplněny o anotace. Pro vytvoření pravidel je takovýto soubor statisticky analyzován a na základě statistik jsou

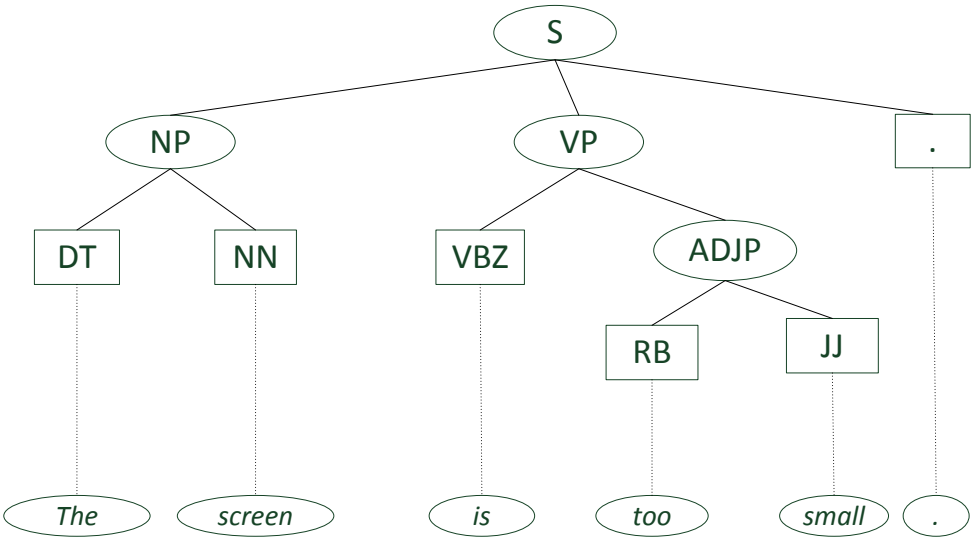
[1] http://nlp.stanford.edu:8080/parser/index.jsp

sestavena pravidla. Podstatné je, aby v trénovací množině byly zastoupeny všechny jevy a tím pádem získaná pravidla byla dostatečně robustní.

Soubor takových pravidel se potom nazývá model, který následně používají analyzátory přirozeného jazyka. Pro každý přirozený jazyk je nutné vytvořit zvláštní model, ovšem pro jazyk jako angličtina jsou většinou tyto modely již hotové. V této práci budu analyzovat anglicky psaný text, tudíž se nebudu již podrobněji zabývat jak takovýto model vytvořit, ovšem mohu čtenáře odkázat na www[2], kde je právě popsána práce s takovýmto modelem a jak jej potom použít v jednom z později uváděných analyzátorů.

5.3.1 Analyzátor větných členů

Prvním typem analyzátoru je analyzátor větných členů v anglické literatuře známý pod názvem *constituency parser*. Tento analyzátor vytváří pro vstupní text, typicky jednu větu, jeho derivační strom na základě PSG 5.2.1. Pro ilustraci mějme větu: *"The screen is too small."* a k ní její derivační strom na obr. 5.4.



Obrázek 5.4: Větný rozbor pomocí Stanford parseru

Neterminály jsou značeny ovály, terminály obdélníky. Kořenem derivačního stromu je uzel označený *S*, který značí větu. Ta je dále tvořena jmennou frází *NP* a slovesnou frází *VP*, která je tvořena adjektivní frází *ADJP*. Jednotlivé fráze jsou potom tvořeny již samotnými slovy a to jsou popořadě *DT*, *NN*, *VBZ*, *RB* a *JJ*. Tyto značky jsou nazývány Part-of-Speech tagy a jsou podrobně popsány v [15]. Těchto značek jsou téměř čtyři desítky, pro účely této práce popíši a uvedu příklady pro ty, které jsou pro tuto práci podstatné.

Přídavná jména jsou dělena do třech kategoriích *JJ*, *JJR* - *comparative* a *JJS* - *superlative*. Příkladem jsou *happy-go-lucky*, *first* nebo *unsurpassed*.

Podstatná jména jsou značena *NN* a *NNS*, kde druhá varianta značí množné číslo.

Velký počet kategorií mají slovesa a to díky časům. Jsou to *VB*, *VBD*, *VBG*, *VBN*, *VBP*, *VBZ*.

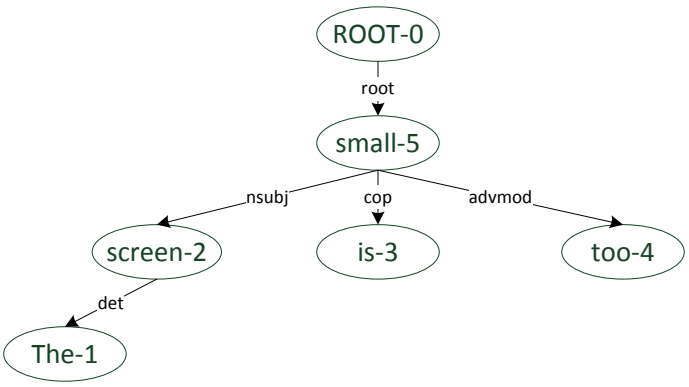
Pro doplnění značek, které jsou uvedeny na obr. 5.4, *DT* je *determiner*, který značí slova jako *a*, *an* nebo *the*. *RB* značí příslovce, např.: *quite*, *too* nebo *very*.

[2]http://www.codeproject.com/Articles/11090/Maximum-Entropy-Modeling-Using-SharpEntropy

5.3.2 Analyzátor závislostí

Analyzátor závislostí spojuje jednotlivá slova na základě jejich závisloti, kterou umí pojmenovat. Tedy na rozdíl od předchozího analyzátoru uzly stromu jsou tvořeny slovy (a čísly značícími jejich pořadí v textu) analyzované věty a ohodnocenými hranami.

Počet závislostí je opět v řádů desítek a lze je nalézt v [9]. Pro potřeby této práce zde uvedu jen některé. Mějme opět větu: *"The screen is too small."* a pro ni závislostní strom na obr. 5.5.



Obrázek 5.5: Rozbor závislostí pomocí Stanford parseru

Z obrázku vidíme, že kořenem je abstraktní kořen *ROOT* na němž závisí slovo *small* a na něm závisí tři další slova s odlišnými závislostmi, konkrétně slovo *screen* se závislostí *nusbj*, *is* se závislostí *cop* a slovo *too* se závislostí *advmod*.

Závislost *nsubj* (nominal subject) je jmenná fráze, která je syntaktickým subjektem klauzule. Řídící uzel relace nemusí být vždy sloveso, může být také přídavné nebo podstatné jméno, např.: $nsubj(small, screen)$.

Příslovečný modifikátor *advmod* je příslovce nebo příslovečná fráze, která slouží k úpravě významu slova, např.: $advmod(small, too)$.

Přímý objekt *dobj* slovesné fráze je jmenná fráze, která je akuzativem[3] slovesa. Příkladem je věta: "I like the mobile phone." se závislostí $dobj(like, phone)$.

Závislost *amod* je adjektivní modifikátor jmenné fráze, který modifikuje význam jmenné fráze. Příkladem je věta "Really high price!" se závislostí $amod(price, high)$.

Poslední závislostí je negace *neg*, která se vyskytuje mezi záporným slovem a slovem, které modifikuje. V příkladu "These shoes are not comfy." je to závislost $neg(comfy, not)$.

5.4 Analýza recenzí

Pro identifikaci kandidátních rysů v uživatelských recenzí budu využívat právě vztahů mezi jednotlivými slovy. Nyní tak na vzorových větách ukážu, jaké vztahy budu hledat, abych identifikoval největší počet rysů.

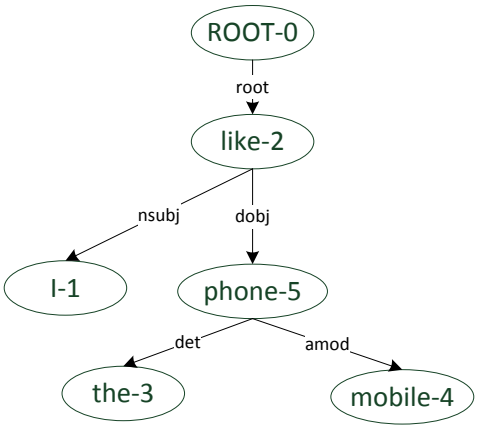
Vzorové věty vyberu z uživatelských recenzí produktů a filmů, které byly diskutovány v kapitole 4. Extrakce bude přibližně probíhat tak, že pro každé podstatné jméno ověřím, zda je v jedné z následujících závislostech, a pokud tak je, potom je takovéto slovo kandidátním rysem.

[3] http://cs.wikipedia.org/wiki/Akuzativ

5.4.1 Přímý objekt - dobj

První názorová věta zní: “I like the mobile phone.” a její strom závislostí je na obr. 5.6. V této větě máme vyjádřen názor na rys *phone*. Tento kandidátní rys má několik závislostí, pro nás je podstatná závislost $dobj(like-2, phone-5)$.

Na základě výskytu rysu v této závislosti můžeme vytvořit první pravidlo. Pokud slovo r je podstatné jméno a závisí na jiném slovu se závislostí *dobj* potom slovo r extrahuj jako kandidátní rys.



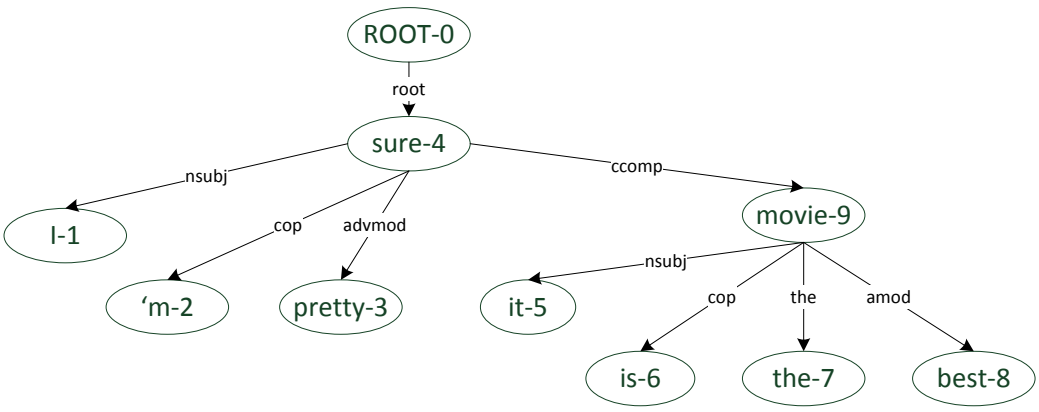
Obrázek 5.6: Strom závislostí se závislostí dobj

5.4.2 Nominální subjekt - nsubj

Nejčastějším vztahem, ve kterém se kandidátní rys vyskytuje, je vztah *nsubj*. Opět mějme větu: “The screen is too small.”, jejíž strom závislostí je na obr. 5.5. V této větě chceme identifikovat rys *screen*, který je v závislosti $nsubj(small-5, screen-2)$.

Druhé pravidlo bude formulováno obdobně jako první. Pokud slovo r je podstatné jméno a závisí na jiném slovu se závislostí *nsubj* potom slovo r extrahuj jako kandidátní rys.

5.4.3 Adjektivní modifikátor - amod



Obrázek 5.7: Strom závislostí se závislostí amod

Pro poslední vztah mějme větu: “I’m pretty sure it is the best movie.” a strom závislosti na obr. 5.7. Rysem, ke kterému se recenzent vyjadřuje, je bezesporu slovo *movie*. Toto slovo se vyskytuje v několika vztazích. Avšak podstatný je vztah $amod(movie - 9, best - 8)$.

Poslední pravidlo pro identifikaci kandidátních rysů zní následovně. Pokud slovo r je podstatné jméno a řídí jiné slovo v závislosti *amod*, potom slovo r extrahuj jako kandidátní rys.

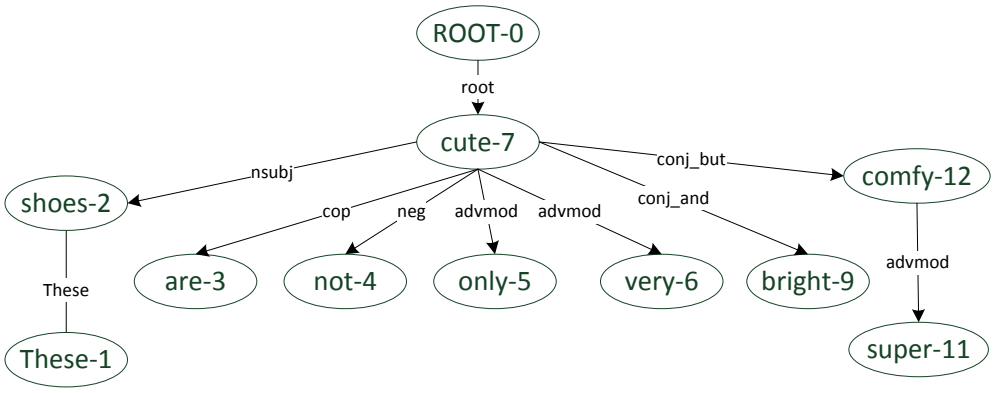
5.4.4 Negace a další modfikátory měnící význam

Vedle rysů a názorových slov jsou ve větě další slova, která mohou zcela změnit význam. Ačkoliv se nelze zaměřit na všechny, některé vzory, které jsou typické pro doménu analyzovaných recenzí, je dobré zmínit. Pro tuto činnost mějme větu: “These shoes are not only very cute and bright but super comfy.” a její strom závislostí na obr. 5.8.

Hlavním modifikátorem je negace, jejíž výskyt velmi snadno otočí význam věty. Naštěstí identifikovat negaci v závislostní gramatice je velmi snadné, protože pro ni máme speciální vztah *neg*. Například ve větě na obr. 5.8 na slově *cute* závisí slovo *not* v závislosti *neg*, které obrací jeho polaritu. Pokud tak slovo *cute* klasifikujeme jako kladné, tak při existenci této závislosti bude jeho celková klasifikace vzhledem ke kontextu záporná.

Ovšem v tomto příkladu jsou i další slova, která jsou závislá na slově *cute*. Sice ne se závislostí *neg*, ale *advmod*. Závislost *advmod* nemění význam názorového slova tak razantně jako negace, ale pouze jej upravuje. V tomto příkladu je např. slovo *cute* modifikováno slovem *very*. Pokud bychom tedy názorová slova hodnotily na stupnici o škále 1-5, potom by takovéto modifikátory posouvaly ohodnocení názorového slova výše.

I přesto, že tato věta obsahuje negaci výsledná polarita slova cute se nemění. Důvodem je právě vzor, který se ve větě nachází, *not only ... but (also)*, kde i přes existenci negace zůstává polarita názorového slova nezměněna.



Obrázek 5.8: Strom závislostí s negací a modifikátory významu

5.5 Dostupné analyzátory pro .NET

Prostředí pro implementaci jsem zvolil .NET s programovacím jazykem C# a vývojovým prostředím Visual Studio 2012. Tato volba byla podmíněna jednak znalostí této platformy a existencí nástrojů pro zpracování přirozeného jazyka, které by z této platformy mohly být snadno použité.

5.5.1 SharpNLP

SharpNLP[4] je open source projekt, který nabízí kolekci nástrojů pro zpracování přirozeného jazyka pro .NET vývojáře. Tento toolkit nabízí následující NLP nástroje:

- sentence splitter
- tokenizer
- part-of-speech tagger
- chunker
- parser
- name finder
- coreference tool
- rozhraní do lexikální databáze WordNet

SharpNLP je postaven na OpenNLP, což je NLP nástroj, který napsali v Javě Jason Baldridge, Tom Morton a Gann Bierner.

Všechny nástroje jsou řízeny maximální entropií modelu zpracovávaného knihovnou SharpEntropy. SharpNLP také obsahuje lexikální databázi WordNet, jejíž přístup zajišťuje knihovna SharpWordNet.

5.5.2 Stanford Parser

Druhým nástrojem je Stanford Parser[5] vytvořený výzkumnou skupinou *The Stanford Natural Language Processing Group* a je volně ke stažení pod licencí GNU General Public License. Stanford Parser má za sebou více než desetiletou historii a postupně na něm pracovalo více než 10 vědců. Jak funguje si lze vyzkoušet i na webu výzkumné skupiny[6]. Ačkoliv je Stanford Parser napsán v Javě, existují porty i do dalších implementačních platforem jako je .NET, Python nebo Ruby.

Stanford Parser zvládá jak PSG tak i DG, kterou prvně uvedený nástroj *SharpNLP* neumí. Vzhledem k tomu, že pro identifikaci rysů potřebujeme analyzovat závislosti mezi slovy, bude muset být pro implementaci použit Stanford Parser.

Pro demonstraci funkcí Stanford Parseru mějme větu: “I’m pretty sure it is the best movie.”. Tuto větu dáme na vstup analyzátoru, jehož výstup je zobrazen na příkladu 5.1.

[4] http://sharpnlp.codeplex.com/
[5] http://nlp.stanford.edu/software/lex-parser.shtml
[6] http://nlp.stanford.edu:8080/parser/index.jsp

```
Your query

  I'm pretty sure it is the best movie.

Tagging

  I/PRP  'm/VBP  pretty/RB  sure/JJ  it/PRP  is/VBZ  the/DT  best/JJS  movie
      /NN  ./.

Parse

  (ROOT
    (S
      (NP (PRP I))
      (VP (VBP 'm)
        (ADJP (RB pretty) (JJ sure)
          (SBAR
            (S
              (NP (PRP it))
              (VP (VBZ is)
                (NP (DT the) (JJS best) (NN movie)))))))
      (. .)))

Typed dependencies, collapsed

  nsubj(sure-4, I-1)
  cop(sure-4, 'm-2)
  advmod(sure-4, pretty-3)
  root(ROOT-0, sure-4)
  nsubj(movie-9, it-5)
  cop(movie-9, is-6)
  det(movie-9, the-7)
  amod(movie-9, best-8)
  ccomp(sure-4, movie-9)

Statistics

  Tokens: 10
  Time: 0.057 s
```

Listing 5.1: Příklad výstupu analyzátoru Stanford Parser

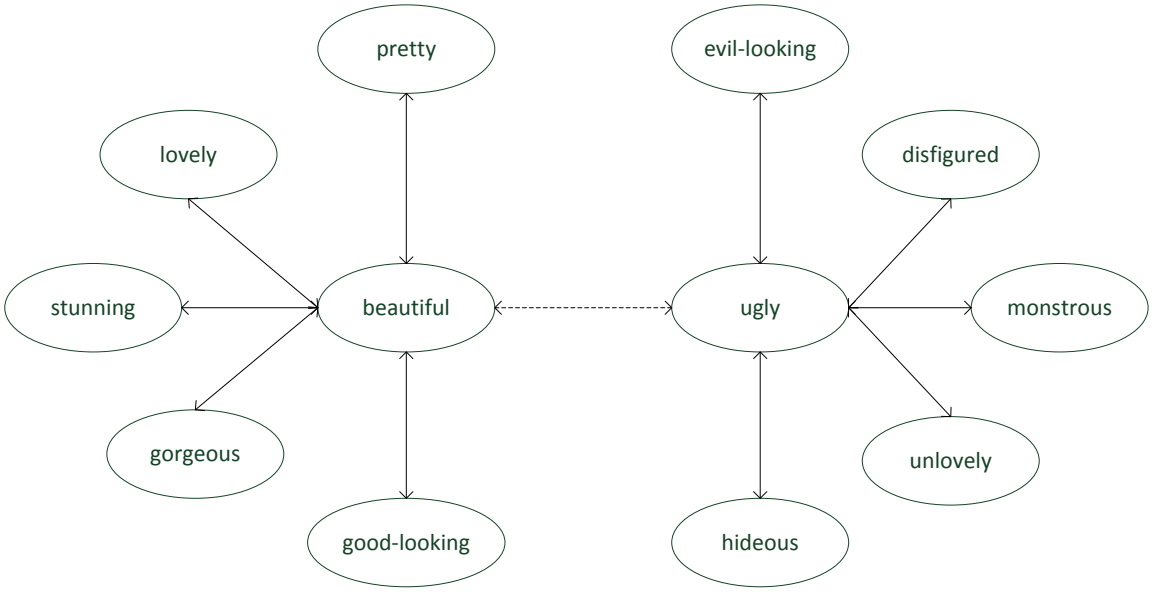
Kapitola 6

Klasifikace názorů

Názory jsou složeny z názorových slov, která jsou vyjádřena k určitému rysu. Většina názorových slov jsou přídavná jména jako *awesome*, *disappointing* nebo *beautiful*. Někdy jsou názory vyjádřeny i jako slovesa např. *like* nebo *hate*.

Pro každé názorové slovo potřebujeme zjistit jeho klasifikaci. V této práci budu názorová slova klasifikovat do 3 tříd a to pozitivní, negativní a neutrální. Třídu neutrálních názorových slov jsem zařadil především z toho důvodu, protože většina názorových slov jsou přídavná jména a přídavná jména jako *mobile* nebo *external* nelze zařadit ani do pozitivní nebo negativní třídy.

Některá názorová slova mohou mít odlišnou sémantiku v závislosti na doméně recenzí. Např. názorová slova popisující velikost nebo hmotnost nelze jednoznačně zařadit do třídy. Slovo *široký* může být recenzentem myšleno pozitivně, např. *široká škála služeb*. Oproti tomu názorová věta *kalhoty jsou v pase příliš siroké*, je recenzentem myšlena zcela negativně. Je tak zřejmé, že není možné vytvořit obecný klasifikátor názorů, na který by se dalo zcela spolehnout.



Obrázek 6.1: Struktura synonym a antonym (synonyma spojena plnou čarou, antonyma přerušovanou)

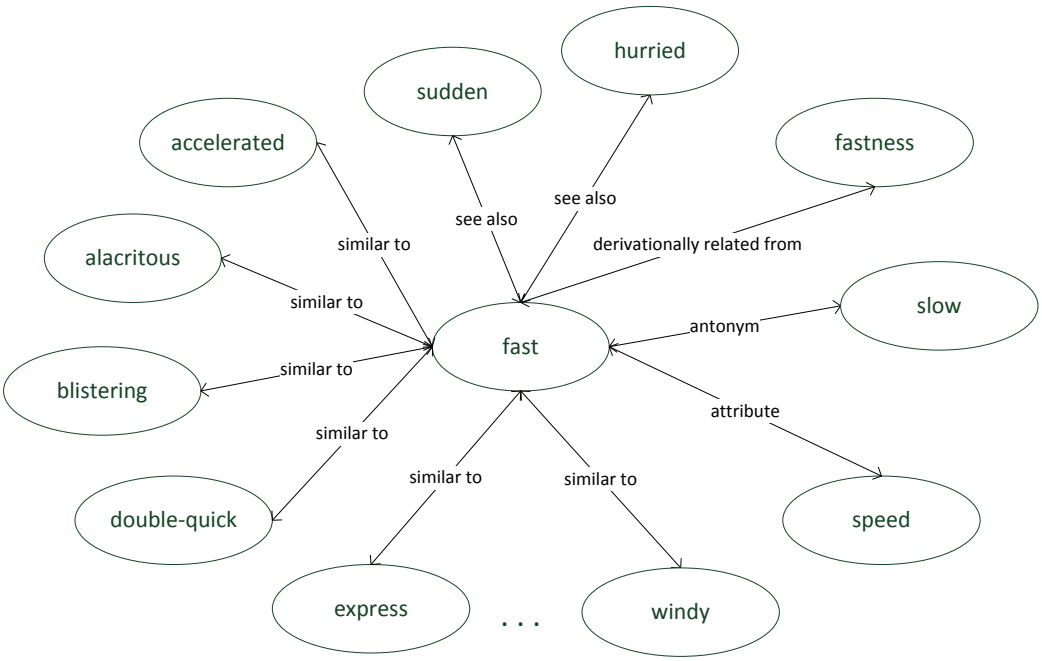
Pro rozpoznání třídy budu používat lexikální databázi WordNet 6.1, která bohužel neumí pro jakékoliv slovo určit jeho sémantickou orientaci. Na druhou stranu WordNet umí nalézt pro libovolné slovo jeho synonyma nebo antonyma. Na základě této vlastnosti mohu implementovat algoritmus, který bude mít na vstupu inicializační seznam pro každou třídu názorů. Pro vstupní slovo vypočte jeho třídu tak, že bude hledat jeho synonyma nebo antonyma taková, která jsou v jednom ze třech inicializačních seznamů a výsledná třída bude ta, v které synonymum našel nebo opačná v případě antonyma. Takhle navržený algoritmus umožní měnit inicializační seznam a například jednou zařadit názorové slovo *wide* do inicializačního seznamu s pozitivními názory a podruhé do inicializačního seznamu s negativními názory. Příklad struktury a vztahů, které je možné dopočíst z lexikální databáze je na obr. 6.1.

6.1 WordNet

WordNet[1] je lexikální databáze anglického jazyka. Je vytvářena od roku 1985 na Princetonské univerzitě.

WordNet spojuje podstatná jména, slovesa, přídavná jména a příslovce do množiny synonym, které jsou nazývány *synsety*. Každý synset vyjadřuje odlišný koncept a dohromady jsou propojeny pomocí sémantických a lexikálních relací. WordNet je volně dostupný a díky jeho struktuře často používán v počítačová lingvistice a při zpracování přirozeného jazyka.

Jako příklad uvedu synset pro přídavné jméno *fast*. Tento synset má tři sémantické relace: *see also*, *similar to*, *attribute* a dvě lexikální relace: *antonym*, *derivation related from*. Synset pro slovo *fast* včetně jeho relací je zobrazen na obr. 6.2, synset je znázorněn oválem a pro tuto ilustraci obsahuje jen jedno slovo. Ovšem v každém synsetu může být slov více.



Obrázek 6.2: Synset pro slovo fast a jeho relace s ostanimi synsety

[1] http://wordnet.princeton.edu/

6.2 WordNet pro .NET

Pro .NET platformu existuje několik implementací pro přístup k lexikonu WordNet. V této práci jsem použil implementaci[2] od Matta Gerbera z Michigan State University. Tato implementace používá WordNet ve verzi 3.0, což je jedna z posledních verzí. Tato knihovna mě zaujala především možností načtení celého lexikonu do paměti a tím pádem velmi rychlého vyhledávání. Celý lexikon zabírá v paměti přibližně 200MB, což vzhledem k dnešní dostupnosti paměti, by byla škoda této vlastnosti nevyužít.

Součástí této knihovny je i demonstrační aplikace, kterou je možné použít na seznámení s WordNetem a následně reverzním inženýrstvím zjistit API pro tuto knihovnu. Aplikační rozhraní této knihovny je tvořeno třemi funkcemi na získání synsetu podle identifikátoru nebo podle slova.

[2] https://ptl.sys.virginia.edu/ptl/members/matthew-gerber/software#wordnet

Kapitola 7

Specifikace a návrh aplikace pro dolování názorů

7.1 Rozbor zadání

Zadáním je vytvořit knihovnu pro dolování názorů. Tato knihovna by měla navenek uživateli nabízet jednu jedinou operaci, jejichž funkcí je dolování názorů. Při volání této operace jsou předány recenze a volitelně rysy. Pokud budou předány rysy potom budou hledány názory pouze na tyto rysy. Jinak při volání operace bez rysů, budou identifikovány v recenzí rysy frekventované.

Výstupem operace bude seznam rysů, kde pro každý rys bude číselně uvedeno kolikrát byl na něj vyjádřen kladný a kolikrát záporný názor. Součástí budou i jednotlivá slova vyjadřující názory a názorové věty.

Pro analýzu uživatelských recenzí bude použit Stanford Parser a identifikace rysů a názorů bude provedena pomocí závislostí mezi slovy ve větě. Kromě explicitních rysů budou identifikovány i rysy implicitní a to prostřednictvím metody coAR.

Identifikace sémantiky názoru bude provedena pomocí inicializačního seznamu názorů, který musí být konfigurovatelný a lexikální databáze WordNet. Názor bude klasifikován na kladný, záporný nebo neutrální.

Výstup této knihovny musí být dostatečně bohatý, aby uživatel mohl využít své kreativity a realizovat tak svůj nápad. Tedy nebudu se zaměřovat jen na výpis rysů a jejich celkové hodnocení nebo generování krátkého shrnutí, ale výstup bude obsahovat dostatek dat, aby uživatel této knihovny byl schopen vytvořit výstup dle jeho libosti.

7.2 Rozbor knihoven a závislostí

Před návrhem softwaru nejdříve musím projít rozhraní knihoven a jejich závislostí, které bude software používat.

7.2.1 Stanford Parser

Stanford Parser je dostupný pouze pro platformu Java. Abychom jej mohli použít v .NETu musíme jej pomocí IKVM.NET[1] přeložit do *dll* knihovny. Celý proces překladu je popsán

[1] http://www.ikvm.net/

na blogu[2] Sergeje Tihona, který v článku popisuje i použití této knihovny.

Ze souboru *stanford-parser.jar* tak vytvoříme *stanford-parser.dll*, který již můžeme použít v .NET aplikaci. Ovšem použitím IKVM.NET vznikají další závislosti a to právě na knihovnách IKVM.NET a to jsou: *IKVM.Runtime.dll*, *IKVM.OpenJDK.Text.dll*, *IKVM.OpenJDK.Util.dll* a *IKVM.OpenJDK.Core.dll*.

Poslední závislostí je model pro anglický jazyk, podle kterého bude analyzátor analyzovat uživatelské recenze. Soubor s tímto modelem je nazván *englishPCFG.ser.gz*. Při vytváření softwarové třídy *LexicalizedParser*, která zpřístupňuje operace pro analýzu, je vyžadována právě cesta k tomuto souboru.

Použití Stanford Parseru z .NETu je poměrně přímočaré a kromě občasného přetypovaní Java kolekcí na .NET generické kolekce, je práce s touto knihovnou bezproblémová.

7.2.2 WordNet

Knihovna pro WordNet od Matta Gerbera používá WordNet ve verzi 3.0, soubory s modelem pro tuto verzi jsou ke stažení na stránkách WordNetu[3]. Při konstrukci softwarové srídy *WordNetEngine*, která implementuje metody pro vyhledávání synsetů ve WordNetu, je potřeba předat jako parametr cestu k modelu.

Tato knihovna dále používá další čtyři knihovny, které je potřeba stáhnout separátně z url[4]. Jsou to *LAIR.ResourceAPIs.WordNet.dll*, *LAIR.IO.dll*, *LAIR.Extensions.dll* a *LAIR.Collections.dll*.

7.2.3 Rekapitulace závislostí

Seznam dll knihoven:

- IKVM.OpenJDK.Core.dll
- IKVM.OpenJDK.Text.dll
- IKVM.OpenJDK.Util.dll
- IKVM.Runtime.dll
- LAIR.Collections.dll
- LAIR.Extensions.dll
- LAIR.IO.dll
- LAIR.ResourceAPIs.WordNet.dll
- stanford-parser.dll

Seznam proměnných pro konfiguraci:

- Cesta k souboru s modelem pro Stanford Parser
- Cesta k adresáři s modely pro WordNet

[2] http://sergeytihon.wordpress.com/2013/02/05/nlp-stanford-parser-with-f-net/
[3] http://wordnetcode.princeton.edu/3.0/WNdb-3.0.tar.gz
[4] https://ptl.sys.virginia.edu/msg8u/NLP/Libraries/Public/

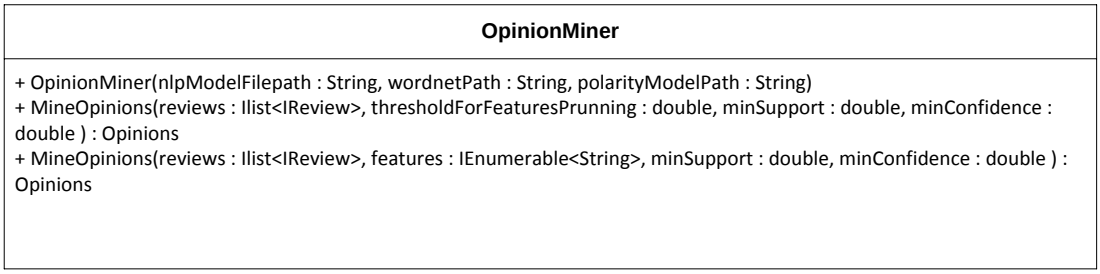
- Cesta k adresáři se soubory pro inicializaci klasifikátoru názorů
- Minimální hodnota prahu pro frekventovaných rysů
- Minimální podpora a spolehlivost pro Co-AR dolování

7.3 Návrh

Program bude navržen jako samostatná *dll* knihovna, kterou je možné použít v mobilních, desktopových, webových nebo jiných aplikacích.

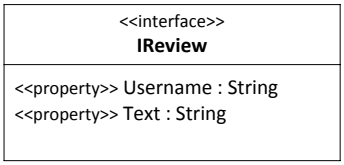
7.3.1 Návrh vstupního a výstupního rozhraní

Vstupním bodem knihovny je veřejná třída *OpinionMiner*, její UML diagram je na obr. 7.1. Při konstrukci této třídy jsou předány cesty k modelům. Konstrukce je tak poměrně složitá a při používání je nejlépe objekt třídy *OpinionMiner* konstruovat tovární metodou, která získá cesty k modelům například z konfiguračního souboru a podobně.



Obrázek 7.1: UML diagram třídy OpinionMiner

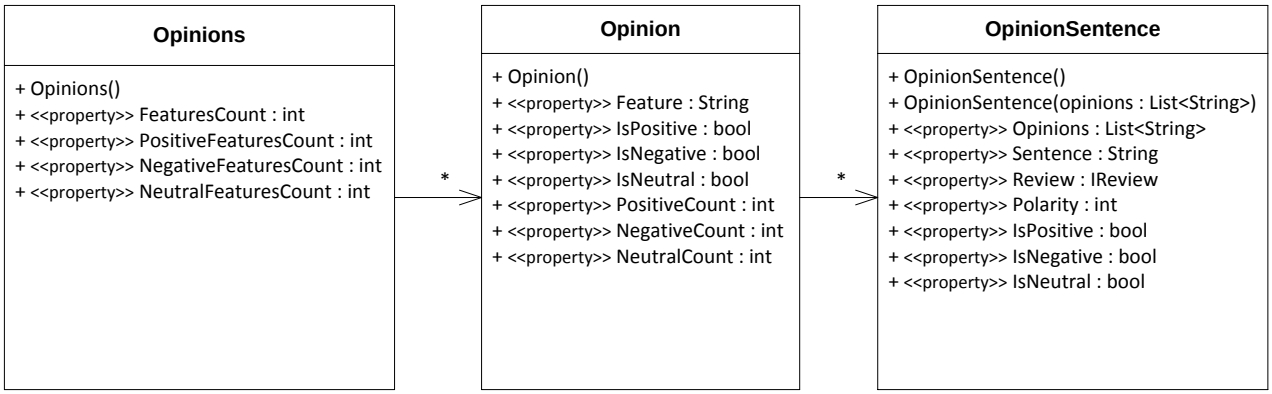
Třída má jednu metodu *MineOpinions*, která provádí dolování názorů. Této metodě lze předat požadované rysy, které chceme v recenzích hledat nebo prahovou hodnotu pro filtraci kandidátních rysů. Současně také předáváme kolekci recenzí a minimální podporu a spolehlivost. Předávané recenze jsou typu *IReview*, UML diagram tohoto rozhraní je na obr. 7.2. Rozhraní jsem zvolil z toho důvodu, protože očekávám, že uživatel této knihovny již bude mít vytvořenu třídu zabalující recenze např. z nějakého API apod., a potom mu stačí, aby už jeho hotová třída pouze implementovala property *Username* a *Text* z rozhraní *IReview*.



Obrázek 7.2: UML diagram rozhraní IReview

Metoda *MineOpinions* vrací objekt typu *Opinions*, který nese agregované hodnoty jako počet rysů a kolik z nich je pozitivních, negativních a neutrálních. Dále tato třída bude mít kolekci názorů, kde názor je tvořen třídou *Opinion*, která obsahuje jeden rys, na který byl vyjádřen názor a kolik názorů bylo klasifikováno jako pozitivních, negativních a neutrálních. Dále obsahuje bool hodnoty, zda je celkové hodnocení daného rysu pozitivní, negativní nebo

neutrální. Součástí je i detailnější pohled na recenze z kterých byly dolovány názory a to v třídě *OpinionSentence*, která bude jako kolekce umístěna v třídě *Opinion*. Třída *OpinionSentence* zachycuje názorovou větu včetně extrahovaného rysu a názoru i celé recenze, v kterých se objevuje.



Obrázek 7.3: UML diagram tříd Opinions, Opinion a OpinionSentence

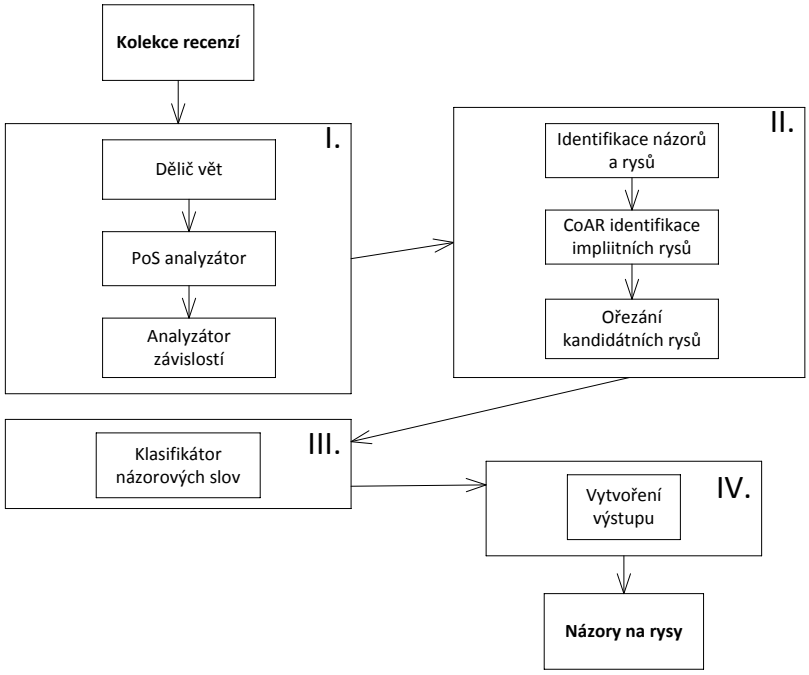
7.3.2 Návrh komponent

Problém dolování názorů je rozdělen do čtyř částí 7.4:

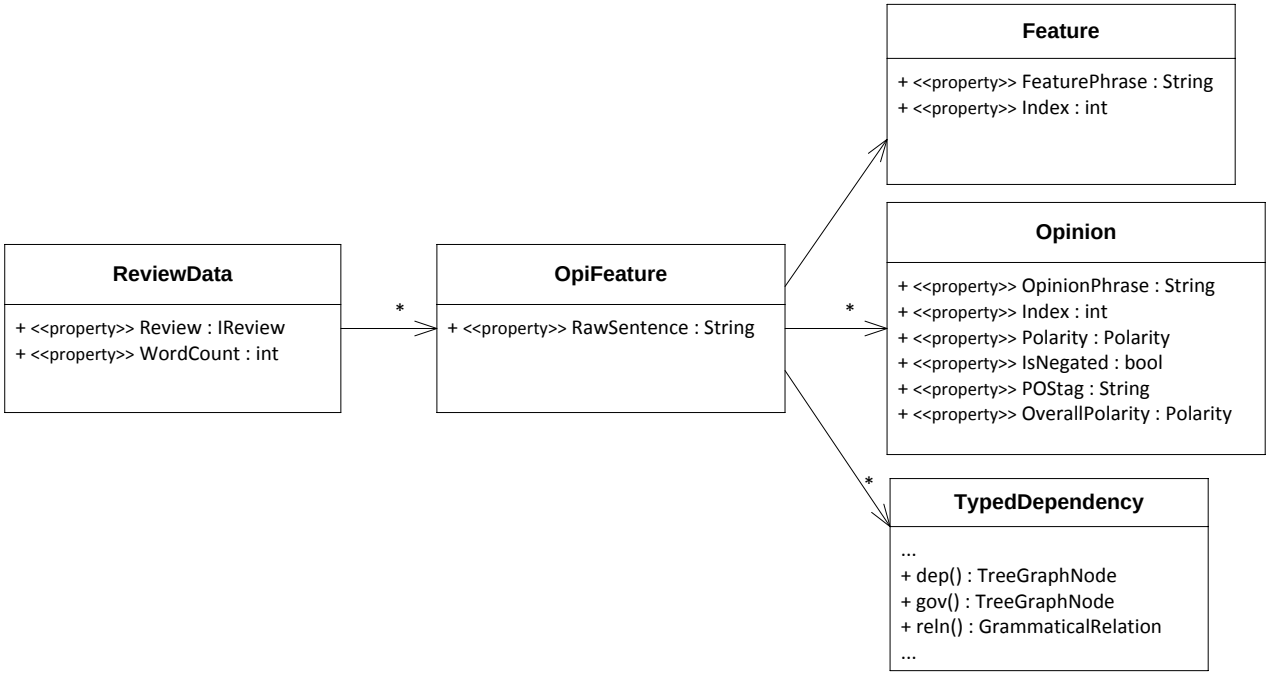
- Analýza recenzí nástroji na zpracování přirozeného jazyka
- Identifikace názorů a rysů v recenzí
- Klasifikace názorů
- Vytvoření výstupu

První komponenta pomocí Stanford Parseru řeší rozdělení recenze do vět a ty jsou následně rozděleny na jednotlivá slova. Analyzátorem jsou zjištěny PoS značky a vztahy mezi jednotlivými slovy věty. Druhá komponenta řeší identifikaci explicitních, implicitních rysů a názorových slov, která k nim náleží. Předposlední komponenta obsahuje samoučící algoritmus, který klasifikuje názorová slova do jedné ze tří tříd pomocí lexikální databáze WordNet. Čtvrtá komponenta řeší vytvoření výstupních dat z interních datových struktur, které jsou použity pro přenos mezi komponentami. Popis implementace jednotlivých komponent je diskutován v kapitole 8.

Mezi jednotlivými komponentami musí být přenášeny zpracované recenze. To znamená, že pro každou větu musí být provedena analýza závislostí a každé slovo musí mít PoS značku. Výsledkem těchto požadavků je třída ReviewData na obr. 7.5, která zapouzdřuje potřebná data každé recenze, aby mohla být zpracována jednotlivými komponentami.



Obrázek 7.4: Schéma částí knihovny



Obrázek 7.5: UML diagram tříd ReviewData, OpiFeature, Feature, Opinion, TypedDependency

Kapitola 8

Popis implementace

V kapitole č. 7 byl čtenář seznámen se vstupním a výstupním rozhraním, komponentami a strukturou interních dat. V této kapitole budou popsány stěžejní algoritmy celého procesu dolování.

8.1 Identifikace explicitních rysů

Identifikace explicitních rysů probíhá v prvním průchodu recenzemi. Nejdříve tak nad každou větou v recenzi proběhne analýza PoS značek a závislostí mezi větami. Pro každé podstatné jméno ve větě jsou zjištěny závislosti a pokud je toto podstatné jméno v závislosti dobj, nsubj nebo amod potom je shledáno jako kandidátní rys. Pseudokód pro tento algoritmus je na příkladu 8.1.

```
1 foreach Word w in Sentence s
2   if(w is Noun)
3     if(w is dependent && w is in relation dobj)
4       extract w as candidate feature;
5     else if(w is dependent && w is in relation nsubj)
6       extract w as candidate feature;
7     else if(w is governor && w is in relation amod)
8       extract w as candidate feature;
```

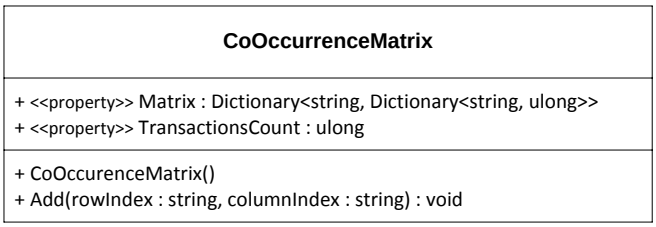
Listing 8.1: Algoritmus pro extrakci explicitních rysů

Algoritmus výše je implementován v třídě *NlpParser*, která při analýze recenzí pomocí knihovny Stanford Parser provádí také extrakci rysů. Tato třída obsahuje instanci analyzátoru (*LexicalizedParser*), vytvoření této instance je poměrně časové náročné, protože při každém vytváření dochází k načtení modelu ze souboru. Proto je tak dobré objekt třídy *NlpParser* vytvořit jen jednou jako *singleton*, aby se předešlo znovu načtení toho samého modelu při zpracování každé sady recenzí.

V uživatelských recenzí se lze se setkat s různými názvy nebo tvary slov pro stejný rys. V této práci není řešeno žádně pokročilé shlukování rysů na základě jejich podobnosti, ale pro zkvalitnění výsledku jsou rysy v množném čísle převedeny na rysy jednotného čísla. V .NETu, konkrétně v O-R Entity Frameworku se nachází služba pro převod jmen tabulek na jednotná a množná čísla. Tato funkce se nachází v třídě *PluralizationService* v jmeném prostoru *System.Data.Entity.Design.PluralizationServices*.

8.2 Identifikace implicitních rysů

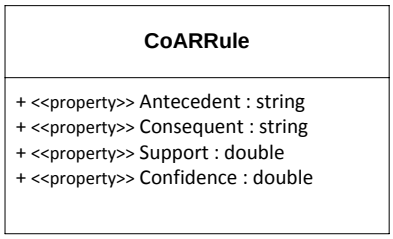
Implicitní rysy jsou identifikovány metodou CoAR, která byla popsána v části 3.3.2. Pro připomenutí metoda CoAR v prvním kroku z explicitních rysů a jejich názorů vytváří matici sounáležitosti, která má na řádcích názorová slova a sloupcích rysy. Buňky matice číselně vyjadřují počet společných výskytů dvojice rys a názor. V druhém kroku jsou z matice generovány asociační pravidla a vypočtena jejich minimální podpora a spolehlivost. Pomocí minimální podpory a minimální spolehlivosti (obě tyto hodnoty uživatel knihovny může měnit) jsou slabá pravidla zahozena. V třetím kroku výsledná asociační pravidla potom slouží k nalezení rysu pro daný názor.



Obrázek 8.1: UML diagram třídy CoOccurrenceMatrix

Matice sounáležitosti je implementována třídou *CoOccurrenceMatrix* na obr. 8.1. Samotná datová struktura uchovávající matici je implementována pomocí generické kolekce *Dictionary* typu klíč-hodnota, kde vyhledání hodnoty pomocí klíče má asymptotickou složitost $O(log(n))$. Klíčem je názorové slovo datového typu *string* a hodnotou je opět generická kolekce *Dictionary*, která představuje řádek matice. Řádek matice je tedy opět kolekce typu klíč-hodnota, kde klíčem je rys datového typu *string* a hodnotou je počet společných výskytů dvojice názor a rys. Pro potřeby výpočtu podpory jednotlivých asociačních pravidel, které jsou generovány z matice sounáležitosti, musíme vědět počet transakcí neboli dvojic názor-rys. Tuto hodnotu uchovává proměnná *TransactionsCount*.

Třída *CoOccurrenceMatrix* implementuje pouze jedinou metodu, která slouží pro přidávání dvojice názor-rys do matice. Při každém přidání je zkontrolována existence, řádku či sloupce. Pokud řádek či sloupec již existuje, potom je inkrementována hodnota udávající počet výskytů, v opačném případě je vytvořen řádek či sloupec a počet výskytů je nastaven na jeden. Při každém přidání je také inkrementováno počítadlo transakcí.

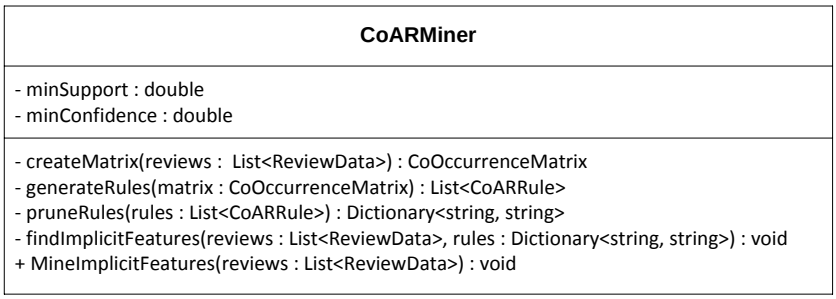


Obrázek 8.2: UML diagram třídy CoARRule

Asociační pravidlo je tvořeno levou částí zvanou antecedent a pravou částí zvanou consequent. Každé asociační pravidlo je charakterizováno podporou a spolehlivostí. Mějme pravidlo $X \rightarrow Y$. Podpora je pravděpodobnost, že transakce obsahuje názor X a zároveň rys Y. Spolehlivost je pravděpodobnost, že transakce obsahující názor X také obsahuje

rys Y. Asociační pravidlo je tvořeno třídou *CoARRule*, která uchovává výše zmíněné atributy a je zobrazena na obr. 8.2.

Veškerá logika dolování implicitních rysů metodou CoAR je implementována v třídě *CoARMiner*, jejíž UML diagram je na obr. 8.3. Její jedinou veřejnou metodou je metoda *MineImplicitFeatures*, která dostane kolekci zanalyzovaných recenzí, kde jsou již rozpoznány explicitní rysy a názory. Pro ty názory, které nemají žádný explicitní rys jsou hledány rysy implicitní, pokud nejsou nalezeny je takovýto názor zahozen. Tuto logiku sekvenčně implementují soukromé metody *createMatrix*, *generateRules*, *pruneRules* a *findImplicitFeatures*.



Obrázek 8.3: UML diagram třídy CoARMiner

Metoda *createMatrix* projde zanalyzované recenze a každou dvojici názor-rys vloží do matice sounáležitosti, kterou vytvoří a vrací jako návratovou hodnotu.

Následuje metoda *generateRules*, která dostane na vstupu matici, která byla vytvořena předchozí metodou. Výstupem je kolekce asociačních pravidel, které jsou spolu s podporou a spolehlivostí vypočteny během jednoho průchodu maticí sounáležitosti.

Slabá aplikační pravidla jsou odstraněna v metodě *pruneRules*, která ze vstupní kolekce asociačních pravidel datového typu *CoARRule* aplikací minimální podpory a minimální spolehlivosti vybere pravidla splňující toto kritérium. Výsledná pravidla jsou uložena v kolekci typu klíč-hodnota, kde je časová složitost vyhledání prvku $O(log(n))$. Klíčem je antecedent a hodnotou consequent asociačního pravidla. Vyhledávání v této kolekci bude probíhat právě pomocí hodnoty antecedent neboli názorového slova.

Aplikace asociačních pravidel probíhá v metodě *findImplicitFeatures*. Na všechna názorová slova bez rysů jsou aplikována asociační pravidla. Pro každý názor je pomocí asociačních pravidel vyhledán rys. Pokud nalezen není, potom je takovýto názor zahozen. V opačném případě je nalezený rys uložen do kolekce zanalyzovaných recenzí.

8.3 Identifikace názorových slov

Identifikace názorových slov je implementována ve třídě *OpinionIdentifier* a to dvěma způsoby. První způsob je identifikace názorů při znalosti rysu a druhým je případ, kdy rys je implicitní a tudíž neznámý.

V prvním případu názory na rysy jsou přídavná jména, která závisí na rysu nebo ho řídí a také slovesa, jenž jsou v řídícím vztahu s rysy. Pseudokód pro identifikace názorových slov je ukázán na příkladu 8.2.

Druhým případem je identifikace názorů ve větách bez rysů. V tomto případě jsou za kandidáty na názorová slova identifikována všechna přídavná jména. Kandidátní názorová slova, ke kterým není nalezen rys, jsou zahozena.

```
1 foreach Feature f and Word w in Sentence s
2   if(w is adjective && w depends on f)
3     extract w as opinion word;
4   else if(w is adjective && w governs f)
5     extract w as opinion word;
6   else if(w is verb && w governs f)
7     extract w as opinion word;
```

Listing 8.2: Algoritmus pro extrakci názorových slov

Za slovesa jsou považována slova s PoS značkami VB, VBD, VBG, VBN, VBP nebo VBZ. Přídavná jména v tomto případě mají PoS značku JJ nebo JJS.

Kromě samotných názorových slov jsou identifikovány také modifikátory měnící význam slova. Nejvýznamnějším modifikátorem je negace. Pokud názorové slovo je ve vztahu *neg* s jiným slovem potom je nastavena značka říkající, že toto slovo je negováno. V případě výskytu negace a slova *only*, které je v závislosti *advmod* s názorovým slovem, je negace zrušena.

8.4 Klasifikace názorů

Názorová slova jsou zařazena do jedné ze tří tříd: pozitivní, negativní nebo neutrální. Klasifikace probíhá samoučícím se algoritmem, který má na vstupu pro každou třídu inicializační seznam názorových slov. Klasifikátor dostane na vstupu slovo, které má zařadit do jedné ze tří tříd. Vstupní slovo zadá do lexikální databáze WordNet a hledá synonyma nebo antonyma taková, u kterých zná třídu, neboli které jsou již v jednom z inicializačních seznamů. Při nalezení synonyma, které se nachází např. v inicializačním seznamu negativních názorových slov, je vstupní slovo klasifikováno jako negativní a je přidáno do inicializačního seznamu negativních slov. Při nalezení antonyma, které se nachází např. v inicializačním seznamu negativních názorových slov, je vstupní slovo klasifikováno jako pozitivní a je přidáno do inicializačního seznamu pozitivních slov.

Při nenalezení synonyma ani antonyma v žádném z inicializačních seznamů, lze tento algoritmus rekurzivně aplikovat na synonyma a antonyma vstupního slova, ke kterým se nepodařilo nalézt jejich třídu. Pokud inicializační seznam nebude naplněn reprezentativním vzorkem názorových slov všech tříd, nelze zajisti konečnost algoritmu jinak než omezením počtu zanoření. Vzhledem k situaci, že tato knihovna umožňuje uživateli volit inicializační seznam pro všechny tři třídy, je omezenost zanoření zcela namístě. Proto tato knihovna hledá rekurzivně synonyma a antonyma jen do první úrovně zanoření. Pokud se nepodaří klasifikovat názorové slovo, potom je zařazeno do třídy neutrální. Uživatel má následně možnost inicializační seznamy doplnit o slova, které se nepodařilo klasifikovat nebo byla klasifikována nesprávně. Klasifikace názorů je implementována v třídě *OpinionClassifier*.

Kapitola 9

Experimenty a jejich vyhodnocení

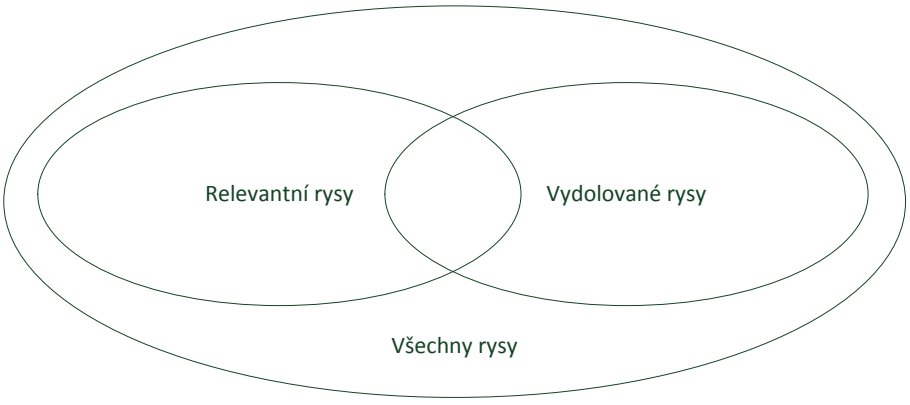
Po provedení experimentů si každý položí otázky: "Našel jsem nejvíce relevantní rysy?" nebo "Chybí mi nějaké podstatné rysy?"

Bohužel získat všechny relevantní rysy a vyhnout se těm nerelevantním je obtížné ne-li nemožné. Naštěstí lze tyto dva parametry měřit pomocí metrik přesnost a úplnost.

V této kapitole budou uvedeny postupně tři experimenty a jejich vyhodnocení včetně popisu způsobu měření.

9.1 Přesnost a úplnost

Přesnost a *úplnost* (recall) jsou základní metriky používané pro vyhodnocení výsledků dolování informací z textu. Jak je ukázáno na obr. 9.1, je předpokládáno, že pro sadu recenzí máme z množiny všech rysů vybranou množinu relevantních rysů, které se snažíme vydolovat. Množinu skutečných vydolovaných rysů však ve většině případů není možné zobrazit na množinou relevantních rysů. Právě metriky *přesnost* a *úplnost* popisují relevanci vydolovaných rysů.



Obrázek 9.1: Relevantní a obdržené rysy

Přesnost udává jak velká část z vydolovaných rysů jsou rysy relevantní. Přesnost se vypočítá podle vzorce 9.1 jako podíl velikosti průniku množin relevantních a vydolovaných rysů s velikostí množiny vydolovaných rysů.

$$precision = \frac{|Relevant \cap Retrieved|}{|Retrieved|} \tag{9.1}$$

Úplnost říká jaká část z relevantních rysů se podařila dolováním získat. Úplnost se vypočte podle vzorce 9.2 jako podíl velikosti průniku množin relevantních a vydolovaných rysů s velikostí množiny relevantních rysů.

$$recall = \frac{|Relevant \cap Retrieved|}{|Relevant|} \tag{9.2}$$

Metrika *F-measure* je harmonický průměr přesnosti a úplnosti a vypočte se podle vzorce 9.3.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{9.3}$$

9.2 Popis dat pro experiment

Pro experiment jsem náhodně vybral jeden produkt, který měl více než 100 uživatelských recenzí a k tomuto produktu jsem náhodně vybral 100 recenzí. Tyto uživatelské recenze jsem přečetl a extrahoval jsem veškeré rysy a názory. Následně jsem vypočítal celkovou třídu pro každý názor. Výsledkem je tak 16 rysů.

Pro otestování jsem provedl 3 experimenty, které jsou rozebrány v dalších částech této kapitoly. Experimenty jsou zaměřeny na vyhodnocení vydolovaných rysů.

Klasifikace je při vhodném inicializačním seznamu bezchybná, protože klasifikujeme do malého počtu tříd. A názory na rys jsou buď z podstatně větší části kladné nebo záporné, jen zřídkakdy se objevují názory na určitý rys, které by byly z poloviny kladné a z druhé poloviny záporné. Proto i menší chyby neovlivní celkovou třídu. Ve vyhodnocení experimentu tak dostává prostor pouze vyhodnocení vydolovaných rysů.

Pro jednotlivé experimenty měním určité komponenty v knihovně pro dolování názorů, tak aby bylo dosaženo lepších výsledků. Každý experiment je proveden pro tři reprezentativní prahy na ořezání rysů.

9.3 Experiment 1

9.3.1 Popis

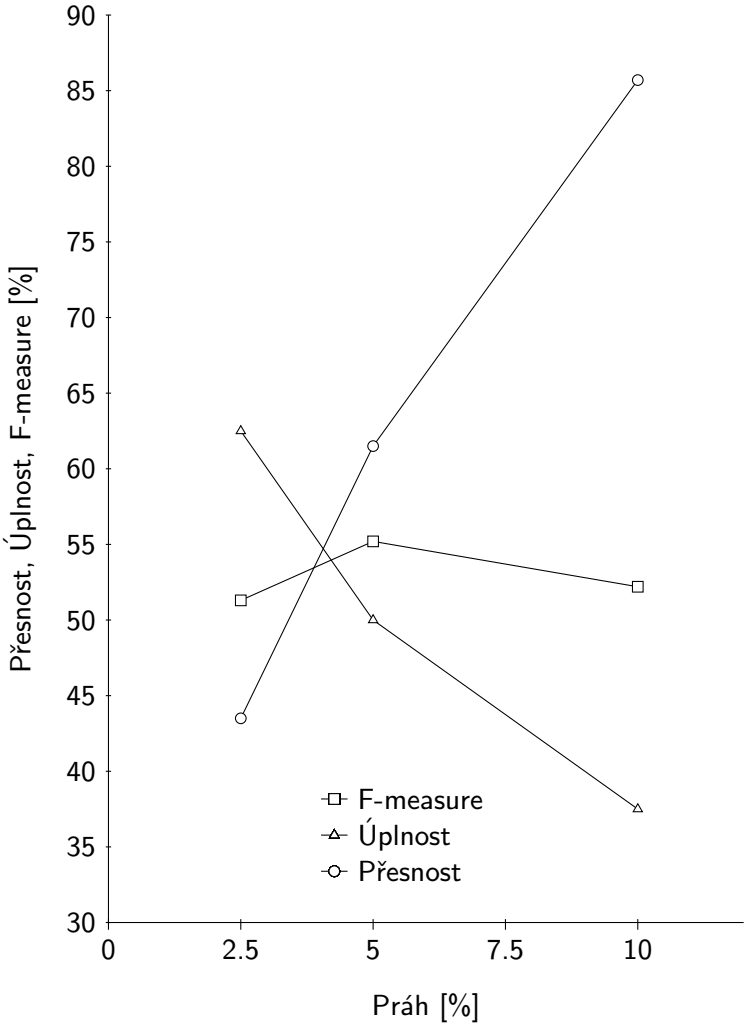
V prvním experimentu použijeme knihovnu pro dolování bez komponenty pro extrakci implicitních rysů a kandidátní rysy budeme filtrovat na základě frekvence výskytu TF. Nejen pro tento experiment, ale i pro další budou kandidátní rysy filtrovány prahovými hodnotami 2,5%, 5% a 10%.

9.3.2 Vyhodnocení

Dosažené výsledky jsou zobrazeny v tabulce 9.1.

Práh pro ořezání [%]	Přesnost [%]	Úplnost [%]	F-measure [%]
2,5	43,5	62,5	51,3
5	61,5	50	55,2
10	85,7	37,5	52,2

Tabulka 9.1: Přesnost a úplnost pro různé prahy ořezání kandidátních rysů



Obrázek 9.2: Graf závislosti přesnosti, úplnosti a F-measure na hodnotách prahu

Na obr. 9.2 je graf závislosti přesnosti, úplnosti a jejich harmonického průměru na různých hodnotách prahu. Můžeme tak vidět, že se zvyšující se hodnotou prahu se zvyšuje přesnost, ale zároveň se také snižuje úplnost. To znamená, že rysy, které získáme dolováním, jsou více relevantní; např. pro práh 10% více než 85% získaných rysů je z množiny relevantních rysů. Na druhou stranu takto získáme jen malé množství rysů. Pro práh 10% je úplnost kolem 37%. Harmonický průměr *F-measure* ukazuje pro jaký práh je kombinace přesnosti a úplnosti nejvyšší, pro tuto sadu recenzí je nejlepším prahovou hodnotou 5%.

9.4 Experiment 2

9.4.1 Popis

Pro druhý experiment jsem přidal k dolování explicitních rysů i dolování rysů implicitních metodou CoAR. Od použití této metody očekávám zvýšení počtu vyextrahovaných rysů zejména těch relevantních. Metoda CoAR vyžaduje zadání minimální podpory a spolehlivosti. Pro zvolenou sadu recenzí jsem zjistil, že nejlepší hodnoty jsou 0,005 pro minimální

podporu respektive 0,15 pro minimální spolehlivost.

9.4.2 Vyhodnocení

Výsledky druhého experimentu viz tabulka 9.2 ukázaly, že použití dolování implicitních rysů na tuto sadu recenzí nepřináší žádné valné zlepšení. Oproti prvnímu experimentu došlo k mírnému zlepšení přesnosti u nejnižšího prahu 2,5%. Zlepšení je sice nepatrné, ale na druhou stranu aspoň nedošlo k žádnému zhoršení. Výsledky prvního a druhého experimentu jsou téměř totožné z toho důvodu, že metoda CoAR kromě extrakce relevantních rysů extrahovala i rysy nerelevantní a po aplikaci ořezání kandidátních rysů nám zůstali frekventovaní rysy téměř nezměněny. Proto v dalším experimentu se zaměřím na použití jiné metody pro ořezání kandidátních rysů.

Práh pro ořezání [%]	Přesnost [%]	Úplnost [%]	F-measure [%]
2,5	45,5	62,5	52,7
5	61,5	50	55,2
10	85,7	37,5	52,2

Tabulka 9.2: Přesnost a úplnost pro 3 prahy v experimentu 2

9.5 Experiment 3

9.5.1 Popis

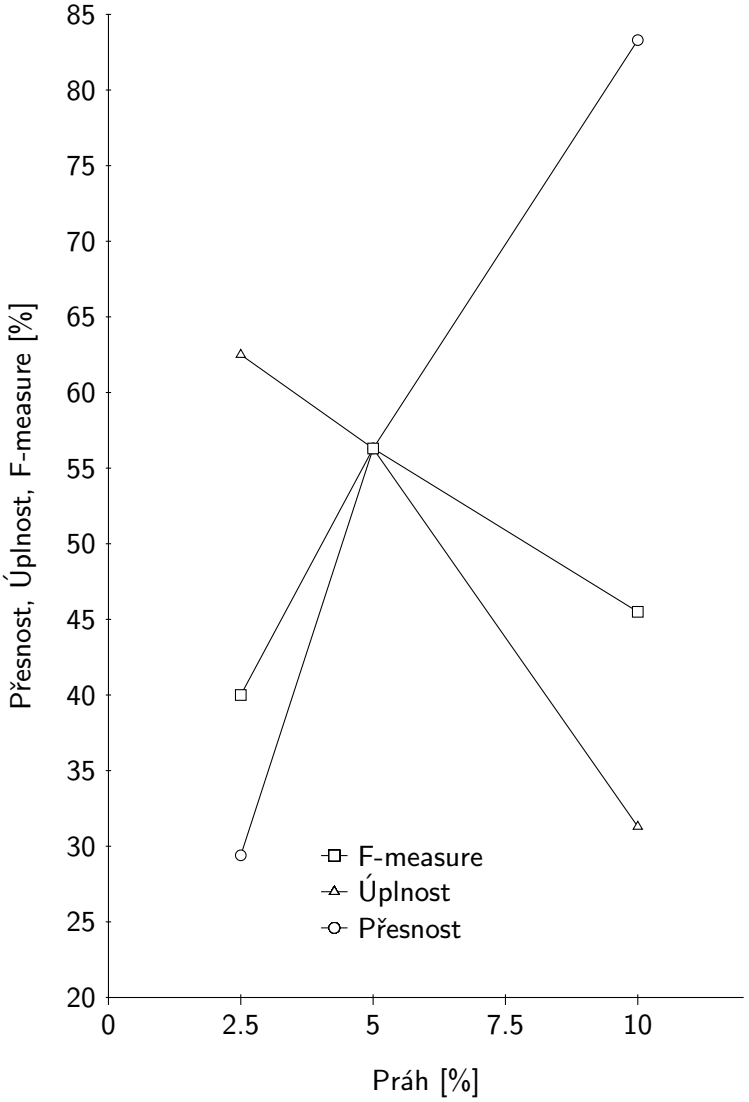
V posledním experimentu změním metodu pro ořezání kandidátních rysů. V předešlých dvou experimentech byly kandidátní rysy ořezány na základě jejich frekvence výskytu. Nyní budou ořezány na základě pomocí metody TF-IDF, která byla popsána v části 3.3.3. Slabinou předchozí metody byl výběr pouze nejvíce frekventovaných rysů. Přitom relevantní jsou i rysy které nedosahují takové frekventovanosti. Při zvyšování hodnoty minimálního prahu docházelo k snižování úplnosti.

9.5.2 Vyhodnocení

Výsledky posledního experimentu jsou shrnuty v tabulce 9.3. Opět se ukázalo, že nejlepšího výsledku se dosáhlo při prahu 5%. Při tomto experimentu bylo také dosaženo nejvyšší hodnoty F-measure a to 56,3%. Pokud porovnáme grafy z prvního a posledního experimentu, který je na obr. 9.3, můžeme vidět že prahové hodnoty měnily hodnotu F-measure pouze o jednotky procent. Avšak při použití TF-IDF je hodnota F-measure více citlivá na jakékoliv změny prahu.

Práh pro ořezání [%]	Přesnost [%]	Úplnost [%]	F-measure [%]
2,5	29,4	62,5	40
5	56,3	56,3	56,3
10	83,3	31,3	45,5

Tabulka 9.3: Přesnost a úplnost pro 3 prahy v experimentu 3



Obrázek 9.3: Graf závislosti přesnosti, úplnosti a F-measure na hodnotách prahu pro experiment 3

Kapitola 10

Příklady použití a výstupy knihovny OpiMiner

V této kapitole ukáži čtenáři jak použít knihovnu OpiMiner pro dolování názorů, která byla popsána a implementována v této práci. Následně budou ukázány příklady krátkých názorových vět a k nim výstupy z knihovny OpiMiner po provedeni dolování. Na závěr potom budou diskutovány případy použití této knihovny.

10.1 Reference

OpiMiner je postaven na analyzátoru Stanford Parser a lexikální databázi WordNet, pro použití této knihovny je potřeba kromě samotné reference na knihovnu OpiMiner.dll nutné přidat další reference. Kompletní seznam referencí a dalších závislostí byl zmíněn při návrhu knihovny v části 7.2.3.

10.2 Příklad použití

Knihovna OpiMiner má jednoduché rozhraní, následující příklad 10.1 ukazuje jak v jazyku C# vytvořit instanci třídy OpiMiner a dolovat názory.

Na 2.řádku a 3.řádku deklarujeme jmenný prostor pro používané třídy. V jmenném prostoru *OpiMiner* se nachází třída *OpinionMiner* a v jmenném prostoru *OpiMiner.Entities* se nacházejí třídy *IReview*, *Opinions*, *Opinion* a *OpinionSentence*, které neimplementují žádnou logiku a slouží jen jako schránky pro předávání dat.

Na 6. až 11. řádku jsou deklarovány a definovány proměnné nesoucí konfigurační hodnoty pro úlohu dolování jako práh, minimální podpora a minimální spolehlivost. Na 14. řádku je deklarována proměnná pro recenze. V tomto příkladu není řešeno načtení recenzí z externího zdroje.

Konečně na 18. řádku je vytvořena instance třídy *OpinionMiner*. Následně je zavolána metoda, kterou začne proces dolování *MineOpinions*. Při volání metody *MineOpinions* jsou předány konfigurační hodnoty práh, minimální podpora a minimální spolehlivost pro konfiguraci úlohy dolování.

Knihovna *OpiMiner* umožňuje nechat uživatele vybrat rysy, které chce dolovat. Tato funkčnost je uvedena na příkladu 10.2. Místo prahu pro ořezání kandidátních rysů je parametrem kolekce řetězců, pro které chceme najít a klasifikovat názory.

```
...
using OpiMiner;
using OpiMiner.Entities;
...

// prah pro prorezani kandidátních rysů - 5%
var threshold = 0.05;
// minimalni podpora pro CoAR - 1%
var minSupport = 0.01;
// minimalni spolehlivost pro CoAR - 30%
var minConfidence = 0.3;

// vytvoreni kolekce recenzi
var reviews = new List<IReview>();
...

// vytvoreni instance tridy OpinionMiner
var opinionMiner = new OpinionMiner(
    @"pathto\englishPCFG.ser.gz",
    @"pathto\WordNet",
    @"pathto\PolarityModels"
    );


// dolovani nazoru
var opinions = opinionMiner.MineOpinions(reviews, threshold, minSupport,
    minConfidence);
```

Listing 10.1: Příklad použití knihovny OpiMiner

```
// kolekce rysu, ke kterym chceme dolovat nazory
var features = new List<string>() {
  "color",
  "quality",
  "price"
  };

// dolovani nazoru
var opinions = opinionMiner.MineOpinions(reviews, features, minSupport,
    minConfidence);
```

Listing 10.2: Příklad použití knihovny OpiMiner pro dolování názorů na základě známých rysů

10.3 Příklady výstupů

Doposud se v této práci neobjevil žádný příklad ilustrující vstup a očekávaný výstup, který by měl být získán. Aby tak čtenář získal patřičný obraz o možnostech knihovny uvedu příklad vstupních názorových vět a k nim výstup.

Jako výstup dolování knihovna vrací objekt třídy *Opinions*, pro zobrazeni tohoto objektu provedu jeho serializaci do formátu JSON.

Mějme například dvě recenze: “I like the mobile phone.” a “The screen is too small.”. Pro tyto dvě recenze jsem provedl dolování a výstupem je objekt *Opinions*, který je v JSON zobrazen na příkladu 10.3.

```
{
   "OpinionCollection":[
      {
         "Feature":"phone",
         "OpinionSentences":[
            {
               "Opinions":[
                  "mobile",
                  "like"
               ],
               "Sentence":"I like the mobile phone .",
               "Review":{
                  "Username":"Anonymous",
                  "Text":"I like the mobile phone."
               },
               "IsPositive":true,
               "IsNegative":false,
               "IsNeutral":false
            }
         ],
         "IsPositive":true,
         "IsNegative":false,
         "IsNeutral":false,
         "PositiveCount":1,
         "NegativeCount":0,
         "NeutralCount":0
      },
      {
         "Feature":"screen",
         "OpinionSentences":[
            {
               "Opinions":[
                  "small"
               ],
               "Sentence":"The screen is too small .",
               "Review":{
                  "Username":"Anonymous",
                  "Text":"The screen is too small."
               },
               "IsPositive":false,
               "IsNegative":true,
               "IsNeutral":false
            }
         ],
         "IsPositive":false,
         "IsNegative":true,
         "IsNeutral":false,
         "PositiveCount":0,
         "NegativeCount":1,
         "NeutralCount":0
      }
   ],
   "FeaturesCount":2,
   "PositiveFeaturesCount":1,
   "NegativeFeaturesCount":1,
   "NeutralFeaturesCount":0
}
```

Listing 10.3: Příklad výstupu knihovny OpiMiner

10.4 Praktické využití knihovny

Při návrhu knihovny *OpiMiner* jsem dbal na to, aby výsledná knihovna nebyla pouze jednoúčelová, ale mohla být použita v široké škále aplikací. Různorodost použití je zaručena bohatým výstupem. Uživatel má k dispozici nejen výsledný rys, názor a jeho třídu, ale také větu v které byl vyřčen a nebo i celou recenzi.

Skutečným problémem může být poměrně vysoká paměťová náročnost, která je způsobena uchováváním dostatečného množství informací o každé větě. Zpracování velkého počtu rozsáhlých recenzí tak může narazit na nedostatek paměti. Tento problém by bylo možné řešit programem, který by velké množství recenzí náhodně rozdělil na jednotlivé sady, z nich by byly získány názory ovšem s nižším prahem, aby nebyly zbytečně ořezány některé rysy. Jednotlivé získané objekty názorů třídy *Opinions* by potom byly spojeny do jednoho takového objektu a ještě jednou prořezány konečným prahem. Ovšem toto řešení může při nedostatečné náhodnosti výběru recenzí pro jednotlivé sady recenzí zkreslit výsledek.

Tradičním využitím této knihovny je získání seznamu rysů a k nim celkovou třídu jejich názorů. V některých případech nás zajímají konkrétní rysy. Například u uživatelských recenzí bot nás budou zajímat rysy cena, velikost, barva. Tento případ knihovna také zvládá pokrýt a místo prahu pro ořezání rysu je předána kolekce rysů, které chceme dolovat.

Kromě zobrazení rysu a jeho klasifikace, můžeme zobrazit uživateli také jednu z vět, v které se rys objevuje včetně samotného názoru.

Jako výrobce produktu mohu pomocí knihovny *OpiMiner* získat seznam vět, v kterých uživatelé hodnotí určitý rys mého produktu kladně nebo záporně. Toho pak lze využít jak pro marketingové účely nebo pro zlepšení produktu.

S knihovnou *OpiMiner* lze generovat i kompletní shrnutí recenzí. Pro každý rys máme k dispozici několik vět, v kterých byl vyjádřen názor na tento rys. Potom nám stačí vzít pro každý rys právě jednu větu a kolekce těchto vět může tvořit shrnutí, které bude pokrývat veškeré názory na všechny rysy.

Tímto výčtem možnosti využití knihovny *OpiMiner* nekončí. Při jiném úhlu pohledu najdeme několik dalších případů, kde tato knihovna najde své místo.

Kapitola 11

Závěr

Efektivnější využití dat ze sociálních médií upoutává pozornost marketingových oddělení téměř každé společnosti. Pochopení uživatelů a jejich preferencí umožňuje společnostem lépe využít své zdroje a vede ke zvýšení konkurenceschopnosti dané společnosti.

V této práci jsem čtenáře uvedl do dolování dat v sociálních médií a seznámil je s s problematikou dolování názorů. Dolování názorů je mezioborová disciplína, která vyžaduje nejen znalosti z oblasti dolování dat, ale také z počítačové lingvistiky. Jedná se tak o komplexní problém, jehož řešení není snadné. Tato práce představila několik metod a technik, které se pro dolování názorů používají. Čtenář byl seznámen s pojmy a pracemi, jenž jsou součástí této oblasti.

Hlavní cíle této práce byly navržení, implementace a otestování knihovny pro dolování názorů, které byly úspěšně splněny. Čtenář měl tak možnost projít celým procesem návrhu a pochopit stěžejní části implementace. Výsledná knihovna pro dolování názorů je přiložena včetně zdrojových kódů v příloze této práce na CD-ROMu. Na závěr této práce byly čtenáři představeny experimenty, které byly provedeny s knihovnou, a jejich vyhodnocení.

Při vyhodnocení experimentů se ukázalo, že problémem současné implementace je ne příliš přesná identifikace rysů, možnost dalšího vývoje bych tedy směřoval tímto směrem. Nabízí se tak možnost vyzkoušení implementace jiné metody a srovnání se současnou. Ačkoliv knihovna nebyla navrhována pro snadnou rozšiřitelnost, tak díky vnitřnímu oddělení jednotlivých částí by přidání jiné metody nemělo činit výrazný problém.

Cestou dalšího vývoje by také mohla být implementace webové služby, která zpřístupní funkcionality této knihovny konzumujícím programům. S tím se také pojí optimalizace výkonu samotné knihovny. I když prostoru pro snížení časové složitosti moc není, protože tu způsobuje především zpracování recenzí, které je implementováno v knihovně třetí strany. Lze se alespoň zabývat snížením paměťové náročnosti.

Přínosem této práce bylo pro mě seznámení se s dolováním dat v sociálních médií a především s dolováním názorů. Moji pozornost zaujala oblast zpracování přirozeného jazyka a také možnosti využití lexikální databáze WordNet. Tato práce nabízí čtenáři jeden z mnoha přístupů k implementaci nástroje pro dolování názorů a může tak být použita pro rozhodnutí jakou cestou se ubírat.

Literatura

[1] Debusmann, R.: An Introduction to Dependency Grammar. 2000, hausarbeit.

[2] Geoffrey Barbier and Huan Liu: *Data Mining in Social Media*. Springer US, 2011, ISBN 978-1-4419-8462-3, 327-352 s.

[3] Hai, Zhen and Chang, Kuiyu and Kim, Jung-jae: *Implicit feature identification via co-occurrence association rule mining*. Springer-Verlag, 2011, ISBN 978-3-642-19399-6, 393-404 s.

[4] Hai, Zhen and Chang, Kuiyu and Song, Qinbao and Kim, Jung-jae: A Statistical NLP Approach for Feature and Sentiment Identification from Chinese Reviews. In *Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing*, 2010, s. 105–112.

[5] Han, J. and Kamber, M: *Data Mining: Concepts and Techniques*. Elsevier Inc., 2006, iSBN 1-55860-901-3.

[6] Hu, Minqing and Liu, Bing: Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, New York, NY, USA: ACM, 2004, ISBN 1-58113-888-1, s. 168–177.

[7] Hu, Minqing and Liu, Bing: Mining opinion features in customer reviews. In *Proceedings of the 19th national conference on Artifical intelligence*, AAAI'04, AAAI Press, 2004, ISBN 0-262-51183-5, s. 755–760.

[8] Kaplan, A. M.; Haenlein, M.: Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 2010: s. 59 – 68.

[9] de Marnee, M.-C.; Manning, C. D.: Stanford typed dependencies manual. URL http://nlp.stanford.edu/software/dependencies_manual.pdf

[10] Mel'čuk, Igor: *Dependency Syntax: Theory and Practice*. State University of New York Press, 1988.

[11] Mitkov, Ruslan (editor): *Part-of-Speech Tagging*. Oxford, UK: Oxford University Press, 2004, ISBN ISBN 0-19-823882-7, 219-232 s.

[12] Pitam Gundecha and Huan Liu: *Mining Social Media: A Brief Introduction*. 2005.

[13] Popescu, Ana-Maria and Etzioni, Oren: Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and*

Empirical Methods in Natural Language Processing, HLT '05, Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, s. 339–346.

[14] Sajin. S. Chandran and Murugappan S.: A Review on Opinion Mining from Social Media Networks. In *Proceedings of European Journal of Scientific Research*, 2012, s. 430–440.

[15] Santorini, B.: Part-Of-Speech Tagging Guidelines for the Penn Treebank Project (3rd revision, 2nd printing). Technická zpráva, Department of Linguistics, University of Pennsylvania, Philadelphia, PA, USA, 1990.

[16] Turing, A. M.: *Computing Machinery and Intelligence*. 1950, 433–460 s.