Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Analýza sentimentu v sociálních sítích

Plzeň, 2014 Karel Zíbar

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 6. května 2014

Karel Zíbar

Poděkování

Tímto bych chtěl poděkovat panu Doc. Ing. Josefu Steinbergerovi Ph.D., zadavateli mé bakalářské práce, za odborné vedení, podnětné návrhy a vstřícný přístup. Dále bych chtěl také poděkovat svojí rodině za veškerou podporu.

Abstract

This thesis deals with an application of a designed lexicon–based method for sentiment analysis on posts and comments publicated on social network Facebook. It also gives a brief view on basic methods of machine learning which are used for this kind of analysis as well. This work also shows basic operations with a chosen Java (free-licensed) framework for working with Facebook API, introduces method used for analysis and finally describes the demo application which was created.

Abstrakt

Tato práce demonstruje použití navržené slovníkové metody pro analýzu sentimentu na textech z postů či komentářů zveřejněných na sociální síti Facebook. Naleznete zde popis této metody i její výsledky při otestování na reálných datech. Dále je zde ukázáno několik základních metod z oblasti umělé inteligence, které se pro tento typ analýzy také používají. V neposlední řadě v této práci naleznete popis používaného free Java frameworku umožňujícího komunikaci s Facebook API a jeho použití při tvorbě výsledné aplikace.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

1.1 Motivace

Analyzování sentimentu má široké uplatnění především v případě zjišťování názorů široké veřejnosti. Většina firem má dnes na internetu své stránky (ať už na Facebooku nebo jinde), kam mohou běžní uživatelé psát své názory např. na nový výrobek dané firmy a je tak velmi snadné získat rozsáhlá data. Extrahování informace o sentimentu z těchto dat se pak stává rychlejší a především levnější alternativou marketingových průzkumů či uživatelských anket. Výhodou tohoto on-line přístupu je také to, že není potřeba žádat uživatele o údaje jako věk, zaměstnání, rodinný stav atd. protože většina lidí tyto položky zveřejňuje na svém profilu a je tak snazší je získat. Díky tomu je pak možné udělat si přesnější obrázek o jednotlivých uživatelích případně určitých skupinách těchto uživatelů. Tyto údaje pak mají velkou hodnotu například pro marketingová oddělení firem, která pak mohou lépe připravovat reklamní kampaně. Příkladem použití tohoto postupu je někdy uváděno předvídání volebních výsledků což může zpřesnit výsledky předvolebních průzkumů.

Více informací o tom, co je to „sentiment“ se dočtete v kapitole 2.

1.2 Cíle práce

Hlavním cílem je navržení a následné otestování vlastní metody pro analýzu sentimentu. Následně je pak zapotřebí vytvořit demo program, který bude schopný z požadované Facebookové stránky získat texty ze zveřejněných publikací (alias postů) a určit, zda-li se jedná z hlediska sentimentu spíše o negativní, neutrální nebo pozitivní text. Aplikace by pak měla tyto získané informace zobrazit uživateli v přehledné podobě a umožnit jejich filtraci na základě jejich polarity z hlediska sentimentu.

Důležitou součástí této práce je také navržení a následné otestování vlastní metody pro analýzu sentimentu (více v kapitole 4).

2 Analýza sentimentu

Analýza sentimentu je podobor spadající pod problematiku zpracování přiro-zeného jazyka (NLP - Natural Language Processing). To je obor, který je v poslední době velmi aktivní oblastí výzkumu a vývoje. Není zde žádná jednotná shrnující definice, ale dalo by se říci, že se jedná o obor na pomezí informatiky, umělé inteligence, lingvistiky, akustiky a dalších vědních oborů, který zkoumá problémy zpracování a generování textů či mluveného slova. To vyžaduje do jisté míry porozumění počítače přirozenému jazyku. V praxi pak může například nějaký interaktivní informační terminál odpovídat plynuleji a hlavně přesněji na dotazy uživatele, překladač textu lépe překládat...

Cílem analýzy sentimentu je určit tzv. *sentiment* textu. V češtině má tento pojem poněkud jiný význam avšak v této práci je chápán jako postoj tvůrce textu k určité otázce. Jedná se tedy o jakéhosi nositele subjektivní informace zanesenou tvůrcem do textu. Obecně může být několik druhů sentimentu [1]. V této práci je však použita kategorizace do pěti tříd – *vysoce negativní, negativní, neutrální, pozitivní a vysoce pozitivní*. První a poslední jmenovanou třídu by bylo možné zanedbat a daný text označit prostě jako pozitivní nebo negativní, pro lepší shodu s poskytnutými slovníky (viz kapitola 2.3.2) byly tyto třídy ponechány. Zvláštní třídou sentimentu je třída *bipolární*, která může mít jak pozitivní tak negativní význam. Tu lze však těžko určit nebo vyjádřit na stupnici. Z těchto důvodů jsem se rozhodl ji v práci zanedbat a rozdělení sentimentu uvažovat jen do pěti výše zmíněných tříd.

Je také nutno vzít v potaz to, že analyzovaný text obecně nemusí obsahovat žádnou informaci o sentimentu [6]. Rozlišují se proto dvě kategorie textu – objektivní a subjektivní. Do objektivní kategorie lze řadit text, který je prost všech emocí a postojů tvůrce textu a obsahuje spíše jen nějaká fakta. Tato kategorie má tedy nulový sentiment a proto bude vždy patřit do neutrální třídy. Pro analýzu sentimentu je více zajímavá kategorie subjektivního textu, kterou je již možné dobře rozdělit do ostatních kategorií.

Některá slova sama o sobě nemusí mít z hlediska sentimentu vůbec žádný význam. Takovým slovům se říká *stopwords* a jedná se především o slovní druhy jako spojky (a, i, ani, ale,...), zájmena (já, ty, on, ona, ...), způsobová slovesa (musím, smíš, ...). Pokud by se udělala statistika frekvence výskytu jednotlivých slov v textu, byla by to právě tato slova, která by se objevovala

v předních pozicích. Na druhé straně této statistiky se nacházejí slova, která jsou naopak velice významná pro analýzu sentimentu. Ta nejčastěji spadají do slovních druhů jako přídavná jména (krásný, hrozné, nejlepší,...), slovesa (milovat, nesnášet,...) nebo příslovce (krásně, hrozně,...).

Jako poslední možnost kategorizace zpracovávaného textu se dá uvést ještě rozdělení podle jeho rozsahu [6]. Může se určovat sentiment u jednotlivých slov, vět a souvětí, odstavců, nebo celých publikací. V této práci je stěžejní určování sentimentu tzv. postů, které však mohou mít podobu od jednoho slova po několik odstavců. Strukturu klasického postu si můžete prohlédnout na obrázku 2.1. Častým případem publikace na Facebooku je také „text“ obsahující jen emotikon čili tzv. smajlík (více v kapitole 2.2).

Je třeba zdůraznit že analýza sentimentu není triviální problém a proto všechny metody a algoritmy pracují s určitou chybou.

2.1 Analýza sentimentu v češtině

Český jazyk má poměrně velkou ohebnost (důsledkem časování, skloňování...) a rozlišuje tak obecně daleko více slov oproti jiným jazykům. Čeština roz-lišuje celkem 42 písmen, což je o 16 více než jich rozlišuje angličtina. Z těchto důvodů je zpracování textů psaných v češtině výpočetně náročnější. Do nedávné doby také neexistovaly nástroje schopné provádět POS tagging a lematizaci – to je důležité z hlediska získávání lepší představy o informaci obsažené v textu a natrénování klasifikátorů algoritmů strojového učení. Pro tuto práci to však není tak důležité vzhledem k tomu, že používané slovníky zachycují různé morfologické tvary slov.

Pro metody strojového učení může být problém, že chybějí označkovaná data. Nejsou zde k dispozici datasety jako např. pro angličtinu *Reuters–21587*, které se využívají k natrénování klasifikátorů i pro jiné jazyky. K natrénování klasifikátorů je tedy zapotřebí použít ne úplně ideálně označkované datasety. [6]

Kromě tohoto však není analýza českého jazyka příliš odlišná od analýzy jiných jazyků.

2.2 Analýza sentimentu v sociálních sítích

Zpracovávání textu z publikací stažených ze sociálních sítí má oproti např. novinovým textům tu nevýhodu, že se zde ve velké míře vyskytují hovorové výrazy a jelikož tyto texty neprochází žádnou korekturou, je zde větší množství překlepů či pravopisných chyb. Na obrázku 2.1 si můžete názorně prohlédnout strukturu takovýchto publikací.

Obrázek 2.1: Náhled na post z Facebooku

Z hlediska analýzy sentimentu nás zajímá spíše text obsažený v těchto publikacích. Je však třeba podotknout, že se např. text komentáře může vztahovat k příloze postu nebo jinému komentáři a tím pádem má jeho analýza spíše negativní vliv na přesnost výsledku v případě, že bychom chtěli analyzovat text vztahující se k určité problematice.

Hovorové výrazy tak velký problém nepředstavují díky struktuře poskytnutých slovníků (více v kapitole 2.3.2), s překlepy a pravopisnými chybami si však navržený algoritmus poradit neumí. Dalším tvarem slov vyskytujícím se na těchto sítích bývají tvary, ve kterých jsou zdvojena (ztrojena,...) některá písmena pro vyzdvižení významu. Příkladem můžou být věty jako „To je superrrrr !“ nebo „Měl sem takovej průůůůšvih.“ atp.

Emotikony

Speciální a z hlediska informace o sentimentu také velice důležitou kapitolou jsou emotikony alias *smajlíci*. Ty někdy (hlavně v případech sarkasmu nebo ironie) vypovídají o mínění tvůrce textu více než samotný text a jsou tedy nedílnou součástí analýzy textů ze sociálních sítí. Facebook v současné době nabízí okolo třiceti základních smajlíků, ke kterým si však každý uživatel může dostáhnout celou řadu dalších.

	Smile	:)	:-)	:]	=)		Frown	:(	:-(	:[	=(
	Wink	;)	;-)				Surprised	:O(	:-O	:o	:-o
	Grin	:D	:-D	=D			Cry	:'(			
	Tongue	:P	:-P	:p	:-p		Unsure	:/	:-/	:\	:-\
	Curly lips	:3					Devil	3:)	3:-)		
	Kiss	:*	:-*				Upset	>:O	>:-O	>:o	>:-o
	Kiki	^_^					Sunglasses	8|	8-|	B|	B-|
	Squint	-_-					Glasses	8)	8-)	B)	B-)
	Confused	o.O	O.o				Pacman	:V			
	Angry	>:(	>:-(				Angel	O:)	O:-)		

Obrázek 2.2: Základní sada emotikonů Facebooku

Z obrázku 2.2 je vidět, že jen někteří smajlíci mají nějakou vypovídající informaci o postoji tvůrce textu a ostatní se dají zanedbat a považovat je tedy za určité rozšíření *stopwords*. Je patrné, že někteří smajlíci se nechají

zapsat vícero způsoby což je také nutno vzít v úvahu, protože ne všichni používají stejný zápis. U smajlíků jsou velice časté také překlepy, kdy místo „:D“ je napsáno například „:d“ nebo „xD“. Pro lepší výsledky se tyto překlepy mohou případně brát také jako korektně zapsané emotikony.

Příklad toho, jaký mají emotikony vliv na výsledné vyznění jednoduché věty jako je věta „Lepší to už být nemohlo.“ je vidět níže:

```
Lepší to už být nemohlo :-D
Lepší to už být nemohlo :-)

Lepší to už být nemohlo :-P
Lepší to už být nemohlo :-|

Lepší to už být nemohlo :-(
Lepší to už být nemohlo >:(
```

Je patrné, že zatímco první dva řádky mají jasně pozitivní vyznění, u třetího a čtvrtého řádku není sentiment zcela jednoznačný. Třetí řádek by byl nejspíše řazen do třídy bipolární. U čtvrtého řádku by se mohlo jednat o třídu neutrální nebo negativní. U pátého řádku by se dalo možná usuzovat, že je tvůrce textu třeba smutný nebo něčeho lituje a poslední je jasně míněný jako sarkasmus. Vyvstává zde tak otázka, co má mít větší váhu při určování výsledného sentimentu – emotikon či slovo?

2.3 Metody analýzy

Metod či algoritmů pro extrahování sentimentu je obecně celá řada. V této kapitole jsou popsány pouze dna hlavními směry – metody strojového učení (kapitola 2.3.1) a slovníkové metody (kapitola 2.3.2) [8].

2.3.1 Metody strojového učení

Tyto metody, označované také jako ML – *machine learning*, spadají pod problematiku umělé inteligence. Jejich princip spočívá v tom, že se trénovací množina (nějaká označkovaná množina dat) poskytne algoritmu, který si na

jejím základě vytvoří vnitřní klasifikační model. Ten je pak použit pro přiřazování zkoumaných textů do jednotlivých tříd na základě procentuální shody s jejími příznaky. Co je to příznak shrnuje následující definice.

Definice: *Příznaky jsou základní části důkazů, které spojují dokument d se třídou c, kterou se snažíme určit.*

(Vapnik, 1982)

Jak bylo řečeno v kapitole 2.1, pro metody využívající strojového učení je v češtině problém, že chybějí označkované datasety, které by byly použity pro natrénování klasifikátoru. Podle [6] je dobré pro tento účel použít např. uživatelské recenze, stažené z nějaké stránky. Je třeba mít na paměti určitá omezení, která s sebou tato data přinášejí:

- Jednotlivá slova nemusí mít nutně stejný sentiment jako výsledný sentiment celého příspěvku.
- S velkou pravděpodobností příspěvek obsahuje řadu nevýznamných slov z hlediska sentimentu.
- Řada příspěvků může být označena špatně.
- V příspěvku se může vyskytovat sarkasmus nebo ironie, což je někdy těžké z psaného textu poznat i lidskýma očima. To pak může negativně ovlivnit příznaky dané třídy a zanést tak chybu do klasifikace.
- Příspěvek může obsahovat pravopisné chyby.

Použitím všech slov z takovýchto dat jako příznaků, by se do klasifikátoru zaneslo velké množství šumu, který by pak negativně ovlivnil vlastní klasifikaci. Tento šum se dá alespoň trochu odfiltrovat zavedením minimální hranice počtů výskytu daného slova (neboli *treshold*). Tím dojde k výraznému omezení počtu použitých příznaků. Tento přístup má bohužel také své nevýhody. Může se stát, že dojde k odfiltrování např. nějakého odborného termínu a ztratí se tak velmi silný příznak (viz následující věta). [6]

Věta: *Příznak má tím větší váhu, čím méně rovnoměrněji je zastoupen v jednotlivých kategoriích.*

(Forman et al., 2003)

Pro určování vah jednotlivých příznaků se používají například iterační algoritmy:

- Improved Iterative Scaling (IIS)
- Generalized Iterative Scaling (GIS)
- Limited–Memory Variable Metric (L–BFGS)

Někdy je potřeba velikost vektoru příznaků zredukovat a nechat pokud možno jen ty příznaky, které nají největší váhu. Algoritmy, které toto řeší pracují tak, že spočtou váhu příznaku na základě jeho výskytech v jednotlivých kategoriích. Mějme následující tabulku:

Tabulka 2.1: Počty příznaků v různých kategoriích

	c_1	c_2
t_k	a	b
$\bar{t_k}$	c	d

kde c_1, c_2 představují dvě třídy klasifikace a $t_k, \bar{t_k}$ přítomnost respektive nepřítomnost příznaku v dané třídě. Nejjednodušší algoritmy pro selekci (výběr nejsilnějších) příznaků pracují s výpočtem pravděpodobnosti výskytu příznaku ve třídě. Pravděpodobnost výskytu příznaku např. ve třídě c_1 by se pak spočetla jako:

$$P(t_k, c_1) = \frac{a}{N} \tag{2.1}$$

kde N je součet příznaků všech tříd – tedy:

$$N = a \ + b + c + d \tag{2.2}$$

Naive–Bayes

Myšlenkou tohoto algoritmu je aplikace *bayesovského teorému* pro vlastní klasifikaci. Tento teorém uplatňuje tzv. naivní přístup – předpokládá, že

výskyt všech slov v textu je statisticky nezávislý na výskytu všech ostatních. Pokud bychom uvažovali rozdělení textu z hlediska sentimentu pouze na negativní a pozitivní, cílem algoritmu je vypočtení pravděpodobnosti $p(+|t)$, tedy pravděpodobnosti, že analyzovaný text t spadá do pozitivní třídy. Text bude do této třídy zařazen v případě, že $p(+|t) \geq 0,5$. Tato pravděpodobnost se tedy podle bayesovského teorému spočte jako:

$$p(+|t) = \frac{p(+) \cdot \prod_i^n p(w_i|+)}{p(+) \cdot \prod_i^n p(w_i|+) + p(-) \cdot \prod_i^n p(w_i|-)} \tag{2.3}$$

p(+) a $p(-)$ vyjadřují pravděpodobnosti výskytu jednotlivých tříd, n je počet příznaků a $p(w_i|c)$ označuje pravděpodobnost výskytu příznaku w ve třídě c. Pokud bychom tento model zobecnili pro m tříd, dostali bychom tvar:

$$p(c|t) = \frac{p(c) \cdot \prod_i^n p(w_i|c)}{\sum_{j=0}^m p(c_j) \cdot \prod_i^n p(w_i|c_j)} \tag{2.4}$$

Jsou zde ještě další metody, které tento základní model rozšiřují nebo upravují. Mezi tyto metody patří např.:[7]

- Gaussian Naive–Bayes
- Multinomial Naive–Bayes
- Bernoulli Naive Bayes

Maximum entropy

Někdy také jako MaxEnt poskytuje méně zkreslené výsledky díky používání komplexnějšího modelu, který již jako zjednodušení neuvažuje statistickou nezávislost výskytu jednotlivých slov, jak tomu bylo v předchozí metodě. Jinak je algoritmus přiřazování do jednotlivých tříd podobný. Přepis pro pravděpodobnost přiřazení textu t do třídy c je následující:

$$p(c|t) = Z(t)^{-1} \cdot e^{\sum \lambda_i w_i(t,c)} \tag{2.5}$$

Kde λ_i udává váhu příznaku w_i. Celková pravděpodobnost musí být z intervalu <0, 1> a proto je zde ještě normalizační funkce $Z(t)^{-1}$.

2.3.2 Slovníkové metody

Tyto metody nevyužívají k určování třídy, do které text spadá, automatický klasifikátor využívající příznaků jednotlivých tříd, nýbrž slovníků. Tyto slovníky obsahují předem roztříděná slova (nebo skupiny slov), tzv. *n–gramy*, na základě jejich polarity případně i intenzity sentimentu, který je reprezentován numerickou hodnotou. Studie na téma analýzy sentimentu pomocí těchto metod (zejména pak [2] a [3]) říkají, že jednotlivá slova, lze rozčlenit do několika následujících skupin [8]:

1. **Neutrální** – Slova nemající žádný význam z hlediska sentimentu. Může se jednat zejména o předložky, spojky, zájmena nebo podstatná jména...
2. **Negativní** – Slova se zápornou polaritou jako např. „nerad“, „ničivý“, „trpět“... (V případě analýzy textů ze sociálních sítí se jedná také o vulgarismy)
3. **Pozitivní** – Slova s pozitivním nábojem jako např. „milovat“, „krásně“, „radostný“...

Dále pak ale existují slova, která sice sama o sobě nenesou žádnou informaci o sentimentu, ale ve spojení s negativním nebo pozitivním slovem, mohou sentiment onoho slova velice ovlivnit. Pokud bychom se podívali na slovní druhy, jednalo by se z velké části o příslovce. Tato slova (dále jako „speciální“) se pak rozdělují do následujících skupin:

4. **Zdůrazňující** – Slova vyzdvihující význam následujícího pozitivního nebo negativního slova. Například spojení slov „velmi krásné“ nebo „stoprocentně příšerné“ ...
5. **Snižující** – Slova mající opačný efekt než předchozí a význam následujícího slova snižují. Příkladem může být „sotva nemoudré“ nebo „možná potěšující“ ...
6. **Invertující** – Slova úplně obracející smysl následujícího pozitivního nebo negativního slova. Např. „to není hezké/ošklivé“ nebo „nebude hnusně/nádherně“ ...

Nutno podotknout, že tato speciální slova nemusí ve větě vždy stát vedle svého „partnera“ a mezi nimi mohou být jiná nevýznamná slova. Někdy

dokonce mohou stát sama o sobě nebo ve spojení s neutrálními slovy a pak je problém určit, jestli pozměnit sentiment některého z následujících slov nebo nikoli. Na následujícím příkladu je vidět rozdělení slov jednoduché věty do jednotlivých kategorií. Jako příklad věty je uvedena věta „*Je velmi podobná své krásné matce, nikoli ošklivému otc*i.“

Tabulka 2.2: Rozdělení slov věty do kategorií sentimentu

Je	velmi	podobná	své	krásné	matce	nikoli	ošklivému	otci
1	**4**	1	1	**3**	1	1	**2**	1

V této větě je tedy jedno zdůrazňující, jedno pozitivní a jedno negativní slovo. Uvažujme teď, že hodnota sentimentu pozitivní a negativní třídy je v absolutní hodnotě stejná. Algoritmus, který by analyzoval větu slovo po slově, by zde mohl zvýšit význam prvního pozitivního/negativního slova, na které by po slově „velmi“ narazil. Výsledný sentiment věty by byl tudíž pozitivní. Z věty je však patrné, že slovo „velmi“ není ve spojení s žádným sentiment nesoucím slovem a tudíž je tento výsledek nesprávný. Jako možné vylepšení se zde nabízí kontrolovat při analýze čárky ve větě a pokud je mezi speciálním a normálním slovem čárka, neovlivňovat sentiment. To by však v tomto případě také moc nepomohlo. Navíc v případě textů ze sociálních sítí jako je Facebook je nutno vzít v potaz také to, že ne každý považuje interpunkční znaménka za důležitá.

Používané slovníky

V práci je použito celkem sedm slovníků. Každý slovník obsahuje slova (popř. skupinu slov) z některé se tříd zmíněných výše. Třída pozitivní a negativní byla každá navíc ještě rozdělena na dva slovníky, ve kterých jsou tak oddělena vysoce pozitivní slova od pozitivních a vysoce negativní od negativních. V následující tabulce 2.3.2 se můžete podívat na používané slovníky a intenzitu sentimentu slov, které obsahují.

Tyto slovníky byly vytvořeny za účelem poskytnutí dat pro slovníkové metody analýzy sentimentu, která by byla stejně koncipována napříč různými jazyky. Ze dvou pivotních slovníků (anglického a španělského) bylo postupně vytvořeno, pomocí automatického překladu, triangulace a manuálních úprav provedených rodilými mluvčími, dalších šest slovníků (arabský, český, fran-

Tabulka 2.3: Slovníky a jejich intenzita sentimentu

Slovník	**Intenzita sentimentu**
Vysoce pozitivní	$+4$
Pozitivní	$+2$
Vysoce negativní	-4
Negativní	-2
Zvyšující	$x + 1 \wedge y - 1$
Snižující	$x - 1 \wedge y + 1$
Inverzní	$x \cdot (-1) \wedge y \cdot (-1)$

Pozn.: *x označuje kladnou a y zápornou hodnotu sentimentu následujícího slova.*

couzský, německý, italský a ruský). Více informací o jejich tvorbě naleznete ve článku [9].

Tato práce je zaměřena na extrakci sentimentu primárně z českých textů s využitím českých slovníků, ale díky stejné struktuře všech vytvořených slovníků, je možné použít jinou sadu a analyzovat tedy text psaný v jiném jazyce. Uvažujme teď slovo „dobrý“. Ve slovníku může být uloženo několika způsoby:

- **„dobrý“** – v tomto případě má jednoznačný význam
- **„dobr_“** – toto zahrnuje slova „dobrý“, „dobré“, „dobrá“, ...

  (Lze také napsat „dobr__“. Takový výraz pak je shodný se slovy jako např. „dobrého“ nebo „dobrému“)
- **„dobr%“** - toto je nejobsáhlejší zápis. Délka ekvivalentního slova není nijak omezena takže jím může být např.„dobromyslný“ nebo „dobromyslného“...

Pro lepší představu je níže ukázáno několik položek ze dvou vybraných slovníků:

anděl%	páchnoucí%
božsk%	příšer%
brilant%	rasist%
báječn%	rozzuř%
dokonal%	shnil%
eufori%	silně+kritizov%
fantastick%	skandál%
fascin%	smrtící%
geniá%	strašn%

(Slovník vysoce pozitivních slov) *(Slovník vysoce negativních slov)*

Takto navržené slovníky mají také tu výhodu, že jdou podle nich do určité míry rozpoznat i hovorové výrazy. To hlavně díky položkám, které obsahují znak '%'. Pokud bychom tedy uvažovali např. větu „*Dnes je ale krásnej den.*“ obsahující jedno pozitivní slovo, výsledky porovnávání s různými položkami slovníku by pak vypadaly následovně:

Tabulka 2.4: Porovnávání položky slovníku s hovorovým výrazem „krásnej“

Položka	**Shoda**
krásný	false
krásn_	false
krásn__	*true*
krásn%	*true*

Díky tomu může být při analýze textů ze sociálních sítí rozpoznáno větší množství slov obsahujících informaci o sentimentu a tím pádem dosaženo i vyšší přesnosti při klasifikaci taného textu. Bohužel je zde pak ještě jeden problém a to ten, že někteří uživatelé Facebooku a jiných sociálních sítí píší často bez interpunkčních znamének. To by ovšem vyžadovalo velké rozšíření stávajících slovníků o další výrazy čímž by také velmi vzrostla jejich velikost. Co s tím? Pro účely této práce byly slovníky rozšířeny o několik nejčastějších slov na Facebooku i s jejich alternativním zápisem v podobě stejného výrazu bez interpunkce (více v kapitole 4.4).

3 Facebook platforma

Facebook platforma je dnes velice dynamicky se rozvíjející set rozhraní pro programování aplikací (neboli API - Application Programming Integface) a nástrojů poskytovaných Facebookem programátorům třetích stran. Tyto nástroje umožňují vytvářet aplikace schopné interagovat s funkcemi jádra Facebooku. Facebook nabízí vývojářům pro vývoj vlastních aplikací SDK pro různé platformy jako např. Android, iOS, JavaScript a PHP. Na stránkách pak lze také nalézt nástroje třetích stran, které sice nejsou Facebookem doporučované, ale mohou pomoci při vývoji. Jedná se zejména o SDK pro další jazyky či technologie jako např. Python, Ruby, Objective-C, HTML, Flash...

Facebook platforma je tvořena několika vysoko-úrovňovými komponentami [11]:

- **Graph API**: Nahrazuje dřívější Rest API a dovoluje vývojářům číst/zapisovat data z/do Facebooku. Poskytuje také pohled na sociální grafy a vztahy mezi jejich entitami.
- **Autentizace**: Umožňuje uživateli přihlášení k různým aplikacím na Facebooku prostřednictvím PC, mobilních zařízení nebo desktopových aplikací.
- **Sociální pluginy**: Různá rozšíření jako například tlačítko „Like“.
- **Open Graph protokol**: Dovoluje vývojářům integrovat své stránky do Facebooku.
- **Iframy**: Umožňuje přihlášení k aplikaci pomocí přihlašování Facebooku ikdyž je aplikace fyzicky umístěna na jiném serveru.
- **Mikroformáty**: Použití *hCalendar* pro události a *hCard* pro místa dovoluje uživatelům Facebooku přesunout tyto informace do svých kalendářů nebo jiných mapovacích aplikací.

V této práci se nebudeme zabývat všemi těmito položkami, ale popíši zde jen první dvě, které byly při tvorbě výsledného programu zapotřebí.

3.1 Facebook Graph API

Jak již bylo řečeno výše, Graph API je primární cesta jak číst data z Facebooku nebo je na něj naopak zapisovat. Je založené na protokolu HTTP a umožňuje dotazování na data, publikování nových, nahrávání fotek... Díky tomu, že je toto API založeno na HTTP protokolu je také možné dotazovat se na Facebook jen za pomoci svého webového prohlížeče.

Struktura

Toto API je tvořeno z *uzlů*, *hran* (např. fotky dané stránky nebo komentáře k fotce...) a *parametrů* (např. datum narození, datum publikace...). Uzly, které byly ve výsledném programu použity, můžete vidět v následující tabulce. Více informací o všech uzlech naleznete na stránkách vývojářů[1].

Tabulka 3.1: Používané uzly Graph API

Uzel	**Popis**
`/{idKomentáře}`	Publikovaný komentář
`/{idPostu}`	Publikovaný post
`/{idStránky}`	Informace o Facebookové stránce
`/{idUživatele}`	Informace o uživateli
`/{idStránky}/feed`	Veškeré publikace publikované na zeď dané stránky
`/{idStránky}/posts`	Publikace na zdi dané stránky publikované vlastníkem stránky

Facebook nabízí pro lepší porozumění celé struktuře svého API nástroj *Graph API Explorer*, který provádí jednotlivé requesty a zobrazuje získaná data v podobě JSON objektů. Nalézt jej můžete na stránkách vývojářů v záložce nástroje[2].

[1] `https://developers.facebook.com/docs/graph-api/reference`
[2] `https://developers.facebook.com/tools/explorer`

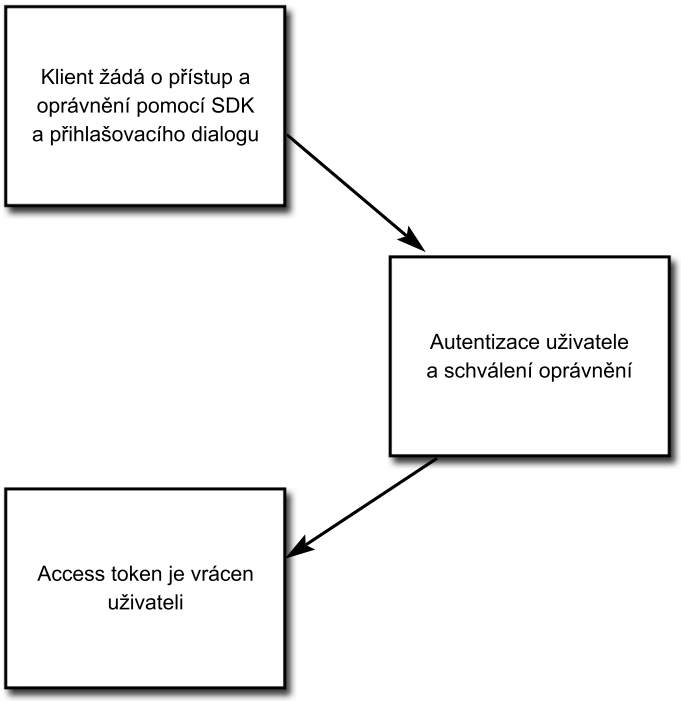
3.2 Autentizace

Každá aplikace interagující s Facebookem se u něj musí přihlásit pomocí svého klíče (APP_KEY) a tajného klíče (SECRET_KEY). Zatímco APP_KEY je volně publikovatelný, SECRET_KEY zná pouze tvůrce aplikace popř. další vývojáři. Pomocí těchto klíčů je vygenerován tzv. *Access Token*, který poskytuje dočasný zabezpečený přístup k Facebook API. Jedná se o jednoznačný string, který identifikuje uživatele, aplikaci nebo stránku a může být aplikací použit k provádějí jednotlivých volání API. Obsahuje také informaci o datu expirace, jakou aplikací byl vygenerován, oprávnění (viz níže)... Kvůli zachování soukromí uživatelů Facebooku, musí většina volaní API obsahovat tento token. Je zde několik různých typů přístupových tokenů [10]:

- **User Access Token** – Nejčastější typ, který je potřeba vždy při volání API kvůli čtení, zapisování, nebo editaci dat. K vygenerování tohoto tokenu je zapotřebí aby se uživatel přihlásil pomocí přihlašování Facebooku a povolil aplikaci.
- **App Access Token** – Tento typ je vyžadován v případě, kdy je potřeba číst nebo změnit nastavení aplikace. Je možné jej získat pouze server–to–server voláním.
- **Page Access Token** – Podobný typ jako user access token. Poskytuje práva na čtení, zápis, nebo modifikaci dat patřící určité Facebookové stránce. Pro získání tohoto tokenu je zapotřebí získat nejdříve user access token a pak požádat o *manage_pages* oprávnění
- **Client Token** – Tento typ je používán pro zpřístupnění limitované části app-úrovně API. Není potřeba provádět žádná volání API za účelem jeho získání – je zveřejněn v nastavení aplikace.

Některé z tokenů je možné získat ve dvou podobách – *short–lived token* nebo *long–lived token* (čili s krátkou nebo dlouho životností). Můžeme tak vygenerovat token s životností okolo hodiny (short–lived) až po přibližně 60 dní. Doba životnosti však nemusí být přesně dodržena. Token může např. důsledkem nějaké chyby vypršet o něco dříve. V případě tokenu získaného přihlášením uživatele do Facebooku se jedná o short-lived token. Pokud je však potřeba tokenu s delší životností, je možné takto získaný token konvertovat na long–lived token.

Přestože každá platforma generuje token trochu jinou cestou, základní strategie zůstává stejná (viz následující obrázek).



Obrázek 3.1: Kroky k získání access tokenu

Důležité k pochopení co je vlastně access token je to, že jakmile je získán nějaký token, je možné volat funkce API přes mobilní zařízení, webový prohlížeč nebo z vlastního serveru. Tokeny jsou tedy portable (přenosné). Pokud je token získán klientem, je možné jej poslat na server a na jeho straně provádět server-to-server volání. Obdobně také pokud je token získán serverem, je možné jej poslat klientovy, který pak může také volat přímo Facebook server.

Permissions

Permissions neboli oprávnění jsou řetězce, které jsou předány spolu s požadavkem na přihlášení nebo s voláním API. Jsou nedílnou součástí získaného access tokenu a uživatel je tedy musí potvrdit při přihlašování. Tato oprávnění jsou zde především z důvodu zachovávání soukromí uživatelů Facebooku a je možné je vybrat z následujících kategorií:

- **Default permissions** – Povolení k přístupu k několika základním veřejným atributům o uživateli nebo o jeho seznamu přátel.
- **Email permissions** – Přístup k primárnímu e–mailu uživatele.
- **Extended profile properties** – Všechny informace o uživateli, které mohou nebo nemusí být součástí jeho veřejného profilu.
- **Extended permissions** – Jedná se o nejcitlivější data o uživateli. Příkladem tohoto oprávněn může být např. i publikování příspěvků na něčí zeď nebo jejich mazání.
- **Page permissions** – Administrátorské oprávnění ke spravování stránky
- **Open Graph permissions**

Pokud je tedy potřeba provádět akce např. s nějakými citlivějšími daty uživatele, musí se při jeho přihlášení odeslat i seznam oprávnění, která budou potřeba. Při generování access tokenu pomocí Graph API Exploreru jsou výše zmíněné kategorie sloučeny do tří následujících:

Tabulka 3.2: Příklady oprávnění

User data permis.	**Friends data permis.**	**Extended permis.**
user_about_me	friends_about_me	create_note
user_birth_day	friends_birth_day	manage_friendlist
user_likes	frieds_likes	read_stream
user_videos	friends_videos	status_update
user_relationship	friends_relationship	create_event
user_photos	friends_photos	share_item

Poznámka: *Nejedná se ovšem o všechna oprávnění. Pro více příkladů viz:* `https://developers.facebook.com/docs/facebook-login/permissions`

4 Analyzátor sentimentu

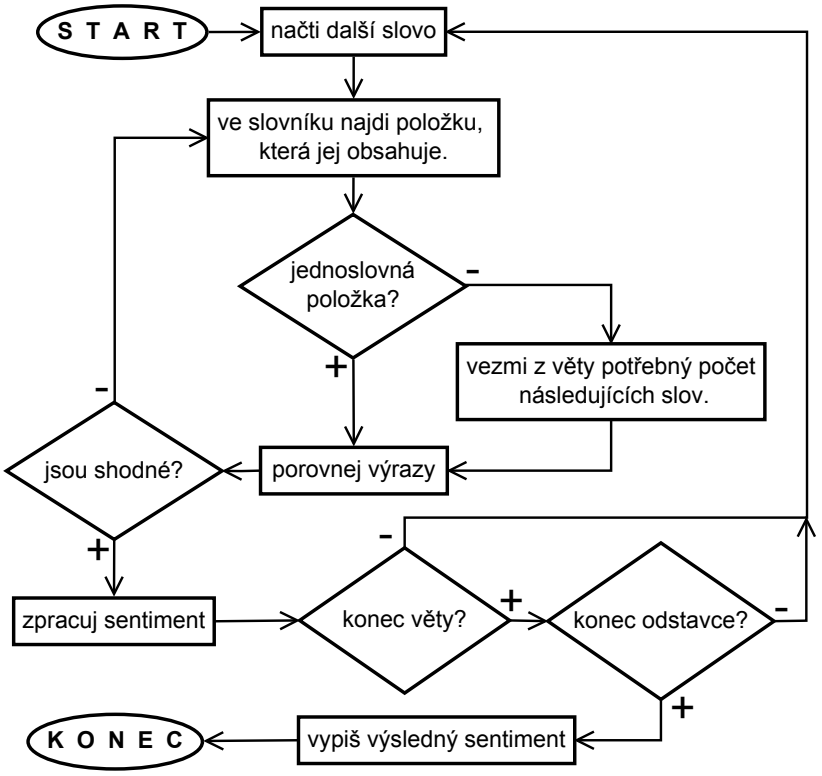
Jak bylo řečeno v úvodu práce – tato výsledná aplikace má podle zadání umožňovat uživateli zadat libovolnou Facebookovou stránku, stáhnout z ní všechny posty (s komentáři a odpověďmi) a z textů, které obsahují zjistit jejich sentiment. Výsledky by pak měli být uživateli zobrazeny v přehledné formě. Níže jsou ve zkratce popsány důležité části této aplikace.

Stahování

Aplikace dává uživateli možnost nastavení několika parametrů stahování postů. Kromě url stránky se jedná o maximální množství stahovaných postů, datum, do kterého se budou posty stahovat, nebo jméno uživatele, od kterého se budou posty stahovat. Tato nastavení lze samozřejmě i kombinovat. Jsou zde z důvodu lepšího zaměření stahování a také pro to, že pokud by aplikace měla stáhnout opravdu všechny publikace obsažené na dané stránce, běžela by velice dlouho důsledkem neustálého dotazování na Facebook server. V neposlední řadě je zde také možnost nastavení toho, jestli se mají stahovat *posts* nebo *feed* (více v kapitole 3.1).

Analýza

V aplikaci je, jak bylo řečeno v předchozích kapitolách, použita slovníková metoda pro analýzu sentimentu (viz kapitola 2.3.2). Používané slovníky byly navíc rozšířeny pro potřeby analýzy textů ze sociálních sítí (viz kapitola 4.2). Algoritmicky se nejedná o nic složitého. Pro většinu slov ve větě se jen najde položka slovníku, která toto slovo obsahuje, zjistí se, zda-li slovo odpovídá výrazu ve slovníku a podle toho se buď pozmění sentiment věty nebo se načte další položka dokud se neprojde celý slovník. Na vlastní algoritmus se může podívat na obrázku 4.1. Lze také nastavit to, jestli se při vlastní analýze bude brát ohled na emotikony obsažené v textu či nikoli. Toto „vylepšení“ může v mnoha případech velice ovlivnit výsledek ať už negativně nebo pozitivně jak je shrnuto v kapitole 5.2.

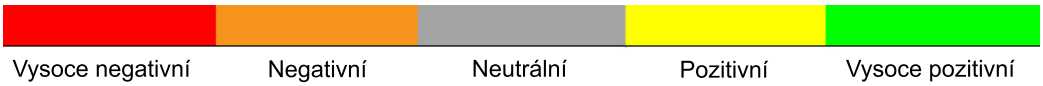


Obrázek 4.1: Algoritmus analýzy sentimentu

Vizualizace výsledků

Jako způsob vizualizace výsledků zde bylo použito vygenerování html souborů, ve kterých jsou zobrazeny všechny stažené posty zleva ohraničené barvou symbolizující jejich sentiment (viz obrázek 4.2). Tyto html stránky pak dávají možnost také takto označené posty různě filtrovat podle hodnoty sentimentu a zobrazit si např. jen pozitivní posty nebo vysoce pozitivní s pozitivními atp. Obdobně je také možné filtrovat i komentáře a odpovědi. Aplikace tak dává uživateli lepší možnost orientace ve větším množstvím postů vygenerovaných na jednu stránku.

Do výsledného html je vygenerováno také několik obrázků zobrazující základní statistiky o stažených textech jako např. jejich rozdělení podle sentimentu nebo kolik bylo celkem staženo postů, komentářů a odpovědí. Aplikace také umožňuje nastavení, kolik postů se bude generovat na jednu stránku



Obrázek 4.2: Barevné rozlišení sentimentu postů

(v rozmezí od 5 do 200) a to opět z důvodu lepší orientace. Na vlastní html reprezentaci analyzovaného postu s komentáři a odpověďmi se můžete podívat v příloze A na stránce 38.

Uživatelské rozhraní

Uživatelské rozhraní bylo vytvářeno pomocí knihovny funkcí Swing a bylo navrženo tak, aby bylo co možná nejvíce intuitivní. Pro lepší ovladatelnost bylo také nastaveno několik klávesových zkratek. Po dokončení stahování (ať už plánovaném nebo neplánovaném) se ve výchozím internetovém prohlížeči otevře první stránka vygenerovaného html s výsledky analýzy. V dolní části veprostřed je pak zobrazen čítač stažených a analyzovaných postů pro informování o stavu.

Na screenshot navrženého uživatelského rozhraní a jeho bližší popis se můžete podívat na obrázku B.1 na na stránce 42.

Více detailů o jednotlivých částech aplikace naleznete v kapitolách 4.3 a 4.4.

4.1 Nástroje použité při vývoji

Z důvodu lepší přenositelnosti mezi platformami, většího výběru knihoven třetích stran a snadnější tvorby uživatelského rozhraní byla výsledná aplikace psána v jazyce Java (verze 1.7). Pro vlastní kódování a tvorbu uživatelského rozhraní pak bylo využito vývojového prostředí NetBeans IDE 7.4.

V programu jsou použity celkem dvě knihovny třetích stran:

- **restfb** – Jednoduchý a flexibilní Facebook Graph API framework psaný v Javě, který je v aplikaci použit pro veškeré stahování dat z Facebooku. Tato knihovna podporuje také starší Facebook Rest API, které se dnes

již moc nepoužívá. Jedná se od open-source software publikovaný pod licencí MIT (více informací neleznete na domovských stránkách v sekci licence[1]).

- **JFreeChart** – Framework pro vytváření různých druhů grafů. V programu je použit pro vytvoření a následného vygenerování obrázků statistik stažených textů. Je publikovaný pod licencí GNU Lesser General Public Licence (více informací naleznete opět na domovské stránce[2]).

Obě tyto knihovny byly vybrány z širšího výběru pro jejich jednoduchost a pochopitelnost jednotlivých postupů. Pro více informací o obou knihovnách doporučuji navštívit domovské stránky, kde je k nalezení příslušná dokumentace a popřípadě také návody.

4.2 Rozšíření slovníků

Pro účely analýzy textů ze sociální sítě bylo potřeba přizpůsobit používané slovníky. Ty byly rozšířeny na základě první poloviny poskytnutých testovacích dat (viz kapitola 5.1). Nejprve byly vytvořeny seznamy slov obsažených v pozitivních a negativních větách. Tyto slova pak byla seřazena podle jejich četnosti výskytu v pořadí od nejčastějších po méně časté. Poté bylo potřeba tyto seznamy ručně projít a vybrat z těchto slov pouze ta, která mají z hlediska sentimentu nějaký význam a tato slova přiřadit do patřičného slovníku. Navíc byly také do slovníků vloženy i alternativní zápisy těchto slov (totéž slovo bez interpunkce). Níže můžete vidět několik nejčastějších slov s jejich četnostmi výskytu z obou seznamů, kde pouze vyznačená slova, pokud nebyla ve slovnících již obsažena, byla použita pro jejich rozšíření.

[1] `http://restfb.com/#licensing`
[2] `http://www.jfree.org/lgpl.php`

jsem 237	jsem 255
taky 109	**děkuji** 186
když 97	jsou 148
jste 94	**díky** 117
bych 92	**nejlepší** 111
proč 84	taky 111
není 83	bych 110
jsou 77	**super** 109
ještě 77	vůně 72
jako 74	ještě 57
(Negativní seznam)	*(Pozitivní seznam)*

K dalšímu rozšíření slovníků (tentokrát pouze vysoce negativního slovníku) byl použit seznam vulgarismů, který je volně ke stažení na webu[1]. Nejedná se o nijak zvlášť obsáhlý zdroj, který by obsahoval všechny možné vulgární výrazy, ale pro účely této práce je však dostačující. Tyto stažené vulgární výrazy pak byly také upraveny do formátu používaných slovníků (viz kapitola 2.3.2) aby bylo možné na jejich základu rozpoznávat více slov v textech z Facebooku.

4.3 Architektura

Program byl navržen tak, aby splňoval podmínky vícevrstvé architektury, konkrétně pak třívrstvé architektury, na jejímž principu je provozována spousta webových aplikací. Ta se skládá z vrstvy *prezentační, aplikační* a *datové*. Níže je popsána funkčnost jednotlivých vrstev z pohledu výsledné aplikace.

- **Prezentační vrstva** – Tato vrstva je tvořena jednak uživatelským rozhraním, kde uživatel zadává parametry stahování a jednak vygenerovaným html, kde jsou pak zobrazeny výsledky analýzy.
- **Vrstva aplikační logiky** – Tato vrstva realizuje vlastní získávání a analýzu dat. Také pak vytváří výsledné html.

[1] `http://mujweb.cz/ksulli/contents.html`

- **Datová vrstva** – Tato vrstva je reprezentována databází Facebooku a v práci tedy nebylo potřeba ji nijak vytvářet.

Z důvodu další rozšiřitelnosti programu byl program navíc rozdělen ještě do několika nezávislých modulů, z nichž každý je možno vyměnit za novější verzi. Jedná se celkem o tři moduly:

- **Modul 1** – v programu tvořen třídou `FacebookPostsDownloader` má především na starosti vlastní stahování postů z Facebooku.
- **Modul 2** – Tendo modul je v programu reprezentován třídou `PostAnalyzator` a má na starosti vlastní určování sentimentu u jednotlivých textů stažených z Facebooku.
- **Modul 3** – Tendo modul je v programu reprezentován třídou `HTMLGenerator` a má na starosti vytváření jednotlivých html souborů a vkládání html reprezentace stažených postů do jejich zdrojového kódu.

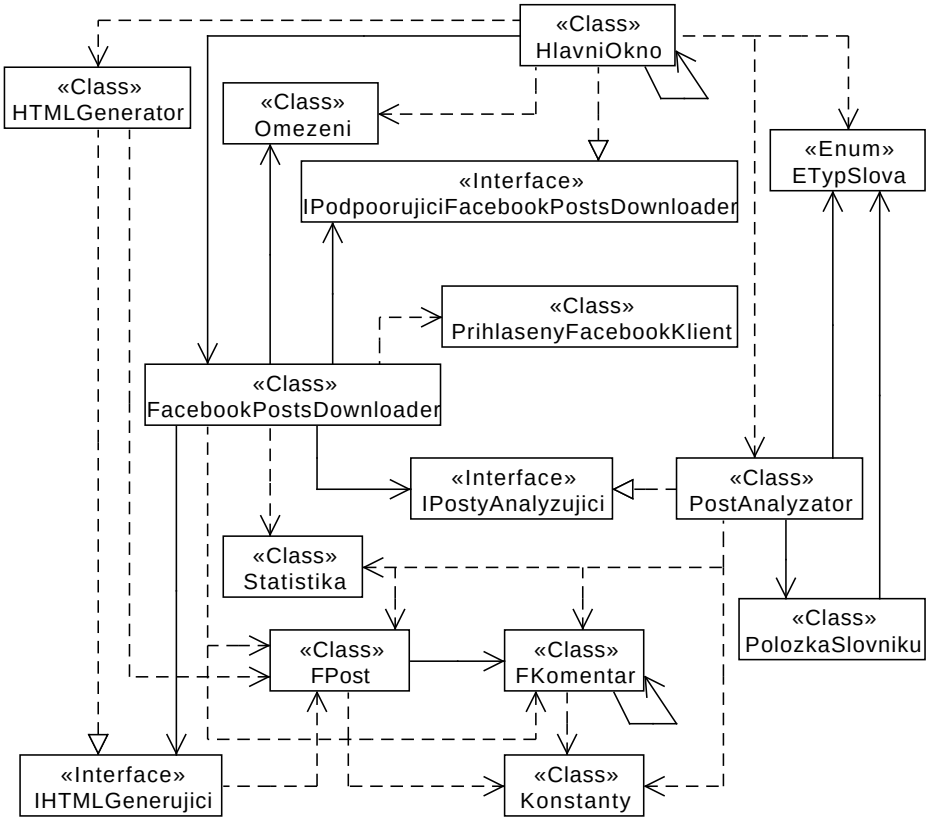
Aplikace byla také navržena jako vícevláknová, kde jedno vlákno řídí uživatelské rozhraní a druhé samotnou aplikační logiku. Je to z toho důvodu aby nedocházelo k zaseknutí GUI vždy, když započne vlastní stahování postů a bylo možné tak program zastavit.

Návrh zlepšení

Takto navržený program není z hlediska optimálního vytížení procesoru navr-žen ideálně. V tomto navržení je každý post nejprve stažen, analyzován a poté uložen v html podobě do souboru. Jako zlepšení by se zde dalo uvažovat o rozdělení stahování a analýzy postů do vlastních vláken. Tím by se docílilo toho, že vlákno, které by posty stahovalo, by nemuselo čekat než analyzující vlákno dokončí svou práci, ale uložilo by aktuální post do bufferu a mohlo by začít stahovat další post v pořadí. Program by tak byl ve výsledku o něco rychlejší.

V takovémto případě by se jednalo o klasický synchronizační problém *Producent/Konzument* (Producer/Consumer), kdy je zapotřebí obě vlákna synchronizovat pomocí synchronizačních mechanizmů (semaforů a mutexů). Tato synchronizace by měla vypadat tak, že producent (v našem případě stahující vlákno) by ukládalo posty do bufferu jen v případě, že by nebyl

zcela zaplněn. Obdobně konzument (v našem případě analyzující vlákno) by z něj posty odebíralo jen tehdy, pokud by obsahoval alespoň jednu položku. Více se tomuto problému věnovat nebudeme, protože to není stěžejní téma této práce a každý si může vyhledat jeho řešení v dostupných zdrojích.



Obrázek 4.3: UML diagram výsledné aplikace
V tomto diagramu jsou zobrazena celkem tři rozhraní, jeden výčtový typ a jedenáct normálních tříd. Popisy jednotlivých tříd a toho k čemu ve výsledné aplikaci slouží pak naleznete níže.

Třídy aplikace

V této sekci se pokusím ve zkratce vysvětlit funkcionalitu jednotlivých tříd, které je možno vidět na obrázku 4.3 zobrazujícím UML diagram aplikace. Všechny tyto třídy byly pro lepší orientaci rozděleny do vlastních souborů.

- **ETypSlova** – Jediný výčtový typ použitý v aplikaci. Jedná se o třídu uchovávající jednotlivé typy slovníků (tedy i slov v nich obsažených) a numerickou hodnotu sentimentu každého z těchto typů.
- **FKomentar** – Třída vytvořená za účelem uchování všech potřebných informacích o komentáři k postu (obdobně pak i o odpovědích na něj). Tato třída také obsahuje metodu `toHTML()`, která dané informace převede do html podoby.
- **FPost** – Třída podobná předchozí, která však reprezentuje celý post. Uchovány jsou zde tedy jednak informace o daném postu ale i o všech komentářích a odpovědích, které se k němu váží.
- **FacebookPostsDownloader** – Stěžejní třída celé aplikace. Tato třída má na starosti veškerou komunikaci s Facebookem za účelem stažení postů. Tyto posty pak předává ke zpracování a po dokončení veškerého stahování a analýzy vytvoří statistiku, ve které jsou zobrazeny základní informace o analyzovaných postech.
- **HTMLGenerator** – Třída, která má na starosti vizualizaci výsledků. Pomocí této třídy jsou tedy do uživatelem vybrané složky překopírovány soubory potřebné pro bezchybné zobrazení a běh statického html, které je pak složeno z html reprezentací jednotlivých postů.
- **HlavniOkno** – Jedná se o klientskou třídu symbolizující také grafické uživatelské rozhraní aplikace.
- **Konstanty** – Třída obsahující pouze veřejné statické proměnné, které jsou v programu přistupovány z různých míst.
- **Omezeni** – Třída navržena podle návrhového vzoru *Přepravka*, sloužící k nastavení jednotlivých omezení pro stahování postů (maximální počet, limitní datum, uživatel) – více v kapitole B.2.
- **PolozkaSlovniku** – Tato třída reprezentuje položku slovníku, ve které je uchován typ sentimentu (`ETypSlova`) a jako pole řetězců pak výraz ze slovníku, který pak slouží pro porovnávání se slovy v analyzovaném textu.
- **PostAnalyzator** – Druhá nejdůležitější třída celé aplikace mající na starosti vlastní analýzu postů. Je zde tedy obsažen algoritmus nevržené metody. Částečně se také podílí na vytváření výsledné statistiky, která je pak vložena do vygenerovaného html.

- **PrihlasenyFacebookKlient** – Jednoduchá třída oddděděná od třídy `DefaultFacebookClient` poskytnuté restfb frameworkem. Vytvoření instance této třídy je jen pohodlnější způsob vytvoření již ověřeného klienta pro volání funkcí Facebook Graph API.
- **Statistika** – Třída podle návrhového vzoru *Přepravka* obsahující počty postů náležících do jednotlivých kategorií sentimentu.

Popis všech rozhraní naleznete v kapitole 4.4.1.

4.4 Realizace

V této kapitole jsou popsány realizační kroky k zajištění v kapitole 4.3 zmíněných návrhů. Také je zde ukázáno několik řádek kódu, který byl ve výsledné aplikaci použit s vysvětlením k čemu slouží.

4.4.1 Zajištění rozšířitelnosti aplikace

Jak bylo řečeno v kapitole 4.3 bylo potřeba vytvořit tři nezávislé moduly, které se mohou snadno vyměnit za novější a sofistikovanější verzi. Z tohoto důvodu naleznete ve výsledné aplikaci tři rozhraní, která oddělují jednotlivé části programu:

- **IPodporujiciFacebookPostDownloader** – Odděluje aplikační logiku od grafického rozhraní. Je tak možné vyměnit stávající grafické rozhraní za jiné (možné je i použití TUI – text user interface).
- **IPostyAnalyzujici** – Oddělení stahování postů od vlastní analýzy textů. Zaručuje implementaci metody pro zanalyzování textů obsažených v instanci třídy `FPost` a vrácení statistických údajů pro vygenerování grafů do výsledného html. Je tak možné vyměnit používanou metodu pro analyzování sentimentu za jinou.
- **IHTMLGenerujici** – Oddělení vizualizace výsledků od stahování a analýzy postů. Zaručuje implementaci metod pro generování html podle předávaných dat.

Na UML Class–diagram celé aplikace se pak můžete podívat na obrázku 4.3.

4.4.2 Komunikace s Facebookem

Spolu s analýzou textů je stahování postů nejdůležitější součástí aplikace. Ve výsledném programu je použita open-source knihovna pro Javu – *restfb*, která je pro tento účel používána (více v kapitole 4.1). Na neštěstí pro nás dokumentace není nejlepší a některé metody třeba ani nefungují. To může být způsobeno tím, že Facebook API se neustále mění a vývojáři takovýchto knihoven nestíhají kód opravovat nebo nekompatibilitou mezi zastaralým Rest API (které knihovna také ještě podporuje) a novějším Graph API.

Další nevýhodou, na kterou jsem narazil, byla malá (přesněji řečeno žádná) odezva od komunit na různých webových diskuzních fórech zabývajících touto knihovnou.

Dále se ve dvou krocích pokusím vysvětlit jak z Facebooku stáhnout post, komentář a odpověď pomocí používané knihovny funkcí *restfb*.

Krok 1: Založení a ověření klienta

Nejprve je potřeba vytvořit klienta, který bude zprostředkovávat jednotlivá spojení s Graph API a vygenerovat access token pro jeho autentizaci. K tomu jsou v knihovně restfb předpřipravené třídy `DefaultFacebookClient` a `AccessToken`.

```
1 DefaultFacebookClient klient =
2       new DefaultFacebookClient();
3 AccessToken accessToken = klient.obtainAppAccessToken(
4       APP_ID,
5       APP_SECRET);
```

Metoda `obtainAppAccessToken()` vrátí klientský access token, který patří do řad tzv. short–lived access tokenů (více v kapitole 3.2). Bohužel způsob vygenerování jiného typu tokenu tato knihovna neumožňuje.

Ve výsledné aplikaci stačí vytvořit instanci třídy `PrihlasenyFacebookKlient`. V konstruktoru této třídy jsou obsaženy oba předchozí kroky, aby nebylo zapotřebí při vytváření dalších klientů je pořád ověřovat.

Krok 2: Navázání spojení

```
1 Connection<Post> feed = klient.fetchConnection(
2       "me/feed",
3       Post.class);
4
5 for (List<Post> posty : feed){
6       for (Post post : posty){
7             //Zpracování postu
8       }
9 }
```

Spojení lze navázat prakticky s čímkoli např. s fotkami, přáteli uživatele, událostmi atp. Jako další příklad je zde získání seznamu přátel přihlášeného uživatele (kořen `{me/friends}`):

```
1 Connection<User> pratele = klient.fetchConnection(
2       "me/friends",
3       User.class);
```

Podobně se pak mohou získat i komentáře k jednotlivým postům. V dokumentaci k restfb je sice uváděna metoda `getComments()`, ta však nefunguje a proto je tento problém potřeba řešit pomocí navázaní dalšího spojení:

```
1 Connection<Comment> con = klient.fetchConnection(
2       POST_ID + "/comments",
3       Comment.class,
4       Parameter.with("limit", 500));
```

Pozn: Analogicky pokud by se místo POST_ID uvedl identifikátor komentáře, je možné tak získat odpovědi.

Poslední parametr metody `fetchConnection()` je nepovinný a udává maximální počet komentářů, které může Facebook odeslat jako odpověď. Pro více informací o problematice komunikace s Facebookem pomocí knihovny restfb naleznete na oficiálních stránkách a popřípadě pak v dokumentaci[1].

[1] `http://restfb.com/javadoc/index.html`

4.4.3 Vytváření grafů statistiky

Pro tuto funkcionalitu je v aplikaci používána knihovna *JFreeChart*, která je pro tento účel ideální. Níže je ve dvou krocích ukázáno vytvoření klasického koláčového grafu a jeho vygenerování do .png obrázku.

Krok 1: Vytvoření grafu

```
1 DefaultPieDataset data = new DefaultPieDataset();
2 data.setValue("Hodnota_01", 30);
3 data.setValue("Hodnota_02", 150);
4
5 JFreeChart graf = ChartFactory.createPieChart(
6     "Název grafu", data, false, true, false);
```

Nejprve je zapotřebí vytvořit dataset s daty, ze kterých se později vykreslí výsledný graf. Inicializace a naplnění tohoto datasetu je ukázáno na prvních třech řádcích. Instance vlastního grafu je pak na řádku 5 a 6, kde je jako parametr konstruktoru předán akorát název grafu, dataset a hodnota true pro generování popisků grafů.

Pro zobrazení vytvořeného grafu uživateli přímo v programu je potřeba vytvořit instanci třídy `ChartFrame`, ve které bude graf zobrazen. Je samozřejmě možné vložit graf přímo do instance třídy `javax.swing.JFrame`, ale využitím možností knihovny JFreeChart si ušetříte práci s pozicováním atd. Na kód pro zobrazení grafu se můžete podívat níže.

```
1 ChartFrame frame = new ChartFrame("Test", chart);
2 frame.pack();
3 frame.setVisible(true);
```

Ve výsledné aplikaci byl však namísto tohoto postupu použit takový, ve kterém byl graf převeden do .png obrázku, který pak bylo možno zobrazit ve vygenerovaném html souboru (viz kód níže).

Krok 2: Exportování grafu do obrázku

```
1 BufferedImage img = graf.createBufferedImage(550, 380);
2 try {
3       File soubor = new File("cesta_k_souboru");
4       ImageIO.write(img, "png", soubor);
5 } catch (IOException ex) {
6     //Ošetření výjimky
7 }
```

Nejprve je zavolána metoda `createBufferedImage()`, která vytvoří obrázek o zadaných rozměrech. Poté je vytvořen nový soubor, do kterého se pomocí `ImageIO` zapíše kód vytvořeného obrázku. Více podrobností o používání této knihovny naleznete na jejích domovských stránkách a v dokumentaci[1].

[1] `http://www.jfree.org/jfreechart/api/javadoc/index.html`

5 Testování a úspěšnost

5.1 Testovací data

K testování úspěšnosti aplikace byla použita data poskytnutá zadavatelem. Tato data jsou tvořena 10 000 posty, které byly staženy z devíti nejnavštěvovanějších českých Facebookových stránek (podle statistiky zveřejněné na webových stránkách společnosti *SocialBakers*). Tyto stránky byly vybrané pro velikost svých fanouškovských základen aby byla zajištěna velká aktivita uživatelů a tím pádem bylo více textů ke stažení. Jednotlivé posty pak byly převedeny do anonymní podoby, aby tak byl zachován pouze jejich textový obsah.

Každý jednotlivý post obsažený v tomto datasetu byl pak označen typem sentimentu (n – negativní, p – pozitivní, 0 – neutrální, b – bipolární) nejprve dvěmi na sobě nezávislými anotátory. Ve 2 216 případech, kdy se tito anotátoři neshodli, byl zapojen třetí, který dané texty také označil. Po této třetí anotaci zde však stále bylo zapotřebí rozhodnout typ sentimentu pro 308 zbývajících textů, u kterých byl stále nejasný. Tyto případy pak byly vyřešeny s pomocí čtvrtého a také posledního anotátora. Bylo také zjištěno, že většina z nich byla ve výsledku označena neutrálním nebo bipolárním sentimentem.

V konečném součtu obsahuje tento dataset celkem 2 587 pozitivních, 5 174 neutrálních, 1 991 negativních a 248 bipolárních postů. Pro více informací o tvorbě těchto dat viz [12].

5.2 Testování a úspěšnost navržené metody

Protože z první poloviny poskytnutých označkovaných dat byla vybrána nejčastější slova nesoucí informaci o sentimentu, na jejichž základě pak byly rozšířeny patřičné slovníky, vlastní testování muselo probíhat na druhé polovině dat. To z důvodu aby nedošlo ke zkreslení dosažených výsledků.

Těchto 5 000 textů bylo následně rozděleno do několika souborů podle typu sentimentu. Aplikace taky byla ve výsledku testována celkem na

čtyřech souborech obsahujících 1) všechny druhy sentimentu, 2) pouze pozitivní, 3) pouze negativní a 4) pouze neutrální. Soubor, který by obsahoval pouze bipolární texty vytvořen nebyl, protože jak bylo řečeno v kapitole 2, tato třída sentimentu je velice špatně rozpoznatelná i pouhým okem a byla tedy v práci zanedbávána. Pro testovací účely byla ponechána jen v prvním případě (tedy v obecném datasetu), ve kterém bylo jako bipolární označeno celkem 141 textů.

Navržená metoda pak byla spuštěna nad daty z každého souboru a to vždy s rozšířenými nebo původními slovníky. Zároveň se metoda spustila pro každý slovník celkem dvakrát, přičemž při prvním spuštění byly brány v úvahu emotikony a při druhém nikoliv. To z důvodu získání lepší představy o tom, jak mohou emotikony v praxi ovlivnit analýzu. V tabulce 5.2 níže jsou zobrazeny počty správně rozpoznaných vět pro jednotlivé soubory/datasety pro všechny kombinace slovníků a emotikonů.

Tabulka 5.1: Počty postů v jednotlivých datasetech

	Obecný	**Pozitivní**	**Negativní**	**Neutrální**
Počet postů	5 000	1 112	1 118	2 629

Tabulka 5.2: Počty správně rozpoznaných postů v různých datasetech

	Obecný	**Pozitivní**	**Negativní**	**Neutrální**
RE	2 613 (52.3 %)	971 (87.3 %)	400 (35.7 %)	1 011 (41.9 %)
R	3 015 (60.3 %)	721 (64.8 %)	290 (25.9 %)	1 865 (70.9 %)
PE	2 584 (51.7 %)	856 (77.0 %)	337 (30.1 %)	1 251 (47.6 %)
P	2 877 (57.5%)	451 (40.6 %)	200 (17.9 %)	2 085 (79.3 %)

Legenda: *RE – rozšířené slovníky s emotikony, R – Rozšířené slovníky bez emotikonů, PE – původní slovníky s emotikony, P – původní slovníky bez emotikonů*

Z tabulky 5.2 je vidět, že nejlepší procentuální úspěšnosti bylo dosaženo při analýze pozitivního datasetu, kde bylo téměř 90 % správně zařazených textů. Je také patrné, že emotikony mohou v některých případech výsledky značně zhoršit, jak je vidět ve výsledcích pro neutrální a zejména pak pro negativní posty. To může být způsobeno právě tím, že jsou některé texty myšlené jako sarkasmus a i když by se podle slov jednalo o jasně negativní

text, několik pozitivních smajlíků snadno převrátí výsledný sentiment na opačnou stranu stupnice. Důvod nepříliš vysoké úspěšnosti metody pro negativní a neutrální posty je vidět v tabulce 5.3, kde je zobrazeno zastoupení postů obsahujících nějaký emotikon v jednotlivých datasetech.

Tabulka 5.3: Zastoupení postů obsahujících emotikony v datasetech

	S emotikonem	**S pozitivním**	**S negativním**
Obecný	2 117 (42.3 %)	1 870 (37.4 %)	279 (05.6 %)
Pozitivní	648 (58.3 %)	648 (58.3 %)	4 (00.4 %)
Negativní	383 (34.3 %)	192 (17.2 %)	198 (17.7 %)
Neutrální	1 011 (38.5 %)	977 (37.2 %)	49 (01.9 %)

Pokud teď nebudeme chvilku brát v úvahu obecný dataset zjistíme, že největší procentuální zastoupení postů obsahujících emotikon má pozitivní dataset. Jedná se však téměř bezvýhradně o pozitivní emotikony a díky tomu je dosahováno skoro 90% úspěšnosti při jejich určování. Naopak je tomu u neutrálních postů. Ty jsou ve 38.5 % nesprávně označeny jako pozitivní nebo negativní důsledkem právě toho, že obsahují smajlíky. Proto také když nejsou při analýze uvažovány emotikony dosáhne úspěšnost jejich rozpoznání k 70 až 80 %. U negativního datasetu pak smajlíci vylepší úspěšnost metody v průměru o 12 %. Jak je vidět pozitivní a negativní emotikony jsou zde rozloženy 50/50 a proto je pravděpodobnost správného rozpoznání negativního postu také poměrně malá.

Při testování aplikace bylo zjištěno, že ideální hodnota sentimentu emotikonů je $+3$ pro pozitivní a -3 pro negativní. Původně byly hodnoty nastaveny na ± 4 díky čemuž bylo dosahováno u negativního a neutrálního datasetu v průměru o 0.5 % horších výsledků. U pozitivního datasetu se výsledky nijak nezměnily. Pro hodnoty ± 2, ± 5 nebo ± 6 pak byly výsledky horší.

6 Závěr

Pro splnění zadání byla navržena metoda z oblasti slovníkových metod namísto metod machine learningu. To z toho důvodu, že byly k dispozici nově vytvořené slovníky a to celkem pro šest jazyků. Tato metoda byla pak testována na reálných datech a byla u ní zjištěna úspěšnost správného rozpoznání sentimentu textu v průměru okolo 60 % (viz 5.2). Přestože úspěšnost navržené metody není nijak vysoká, za zdůraznění stojí její jednoduché použití pro různé jazyky a nepotřebnost trénovacích dat. Pro analýzu textů psaných v jiném než českém jazyce stačí totiž vyměnit aktuálně používanou sadu slovníků za jinou a je možno v daném jazyce analyzovat.

Další využití této práce

Metoda pro analýzu sentimentu navržená v této práci (viz kapitola 4) již byla použita v kvalifikační práci *Extrakce sociálních sítí ze zpravodajských textů* (Lukáš Witz, 2014). Zde je používána pro extrahování informace o sentimentu z přímých řečí obsažených v jednotlivých článcích. Tato informace je pak využívána k určení typu vazby mezi dvěmi entitami ve výsledném grafu. Je tak možné určit např., že Jan Dvořák řekl něco negativního o Josefu Novákovi atp.

Následný vývoj

Jako další vývoj by se nabízelo rozšíření pole působnosti dosavadní aplikace o stránky osobních profilů uživatelů na Facebooku. Publikace na těchto stránkách jsou velmi často privátní z důvodu zachování soukromí a proto je výsledná aplikace, tak jak je nyní navržena, nedokáže stáhnout a její využití je tedy zatím jen na veřejné stránky firem, sociálních skupin atp. Řešením pro toto rozšíření by bylo přihlášení vlastníka onoho profilu do Facebooku. Tím by byl získán user access token a dané publikace by již bylo možné stáhnout (viz 3.2). K tomu by však bylo potřeba použití jiného Facebook Graph API frameworku než je používán nyní, protože ten stávající toto bohužel nepodporuje.

Literatura

[1] TSYTSARAU, Mikalai; PALPANAS, Themis. Survey on mining subjective data on the web. Data Mining and Knowledge Discovery, 2012, 24.3: 478-514.

[2] POLANYI, Livia; ZAENEN, Annie. Contextual valence shifters. In: Computing attitude and affect in text: Theory and applications. Springer Netherlands, 2006. p. 1-10

[3] KENNEDY, Alistair; INKPEN, Diana. Sentiment classification of movie reviews using contextual valence shifters. Computational Intelligence, 2006, 22.2: 110-125.

[4] STEINBERGER, Josef, et al. *Multilingual Entity-Centered Sentiment Analysis Evaluated by Parallel Corpora.* In: RANLP. 2011. p. 770-775.

[5] PATOČKA, Michal. *Metody strojového učení pro analýzu sentimentu. 2013.*

[6] LIDDY, Elizabeth D. *Natural language processing.* 2001.

[7] Scikit-learn. Naive Bayes [online]. [cit. 2014-04-15]. Dostupné z: http://scikit-learn.org/stable/modules/naive_bayes.html

[8] TABOADA, Maite, et al. *Lexicon-based methods for sentiment analysis.* Computational linguistics, 2011, 37.2: 267-307.

[9] STEINBERGER, Josef, et al. *Creating sentiment dictionaries via triangulation.* Decision Support Systems, 2012, 53.4: 689-694.

[10] Vývojáři společnosti Facebook. Documentation [online]. [cit. 2014-04-15]. Dostupné z: https://developers.facebook.com/docs/

[11] Techopedia. Facebook Platform [online]. [cit. 2014-04-15]. Dostupné z: http://www.techopedia.com/definition/27916/facebook-platform

[12] I. Habernal, T. Ptáček, J. Steinberger. 2013. *Supervised Sentiment Analysis in Czech Social Media*

A Reprezentace analyzovaného postu

Obrázek A.1: HTML reprezentace analyzovaného postu

B Uživatelský manuál

Ze zdrojových kódů a potřebných dalších souborů byl vytvořen .jar soubor pro snadnější spouštění. Na strukturu celého programu se můžete podívat na následujícím obrázku B.2. Jednotlivé prvky této struktur jsou pak popsány níže.

- **dict** – složka obsahující používanou sadu slovníku, kterou je tak možné vyměnit např. za sadu v jiném jazyce.
- **html** – složka se soubory potřebnými pro bezproblémové zobrazení a fungování vygenerovaných výsledků.
- **img** – složka obsahuje jeden obrázek, který byl použit v GUI.
- **lib** – složka obsahující obě používané knihovny, aby bylo možné je vyměnit za novější ale *kompatibilní* verzi.
- **src** – složka obsahující zdrojové kódy aplikace.
- **AnalyzatorSentimentu.jar** – spustitelný soubor aplikace.
- **README.TXT** – textový soubor s těmito informacemi.

B.1 Ovládání

Ovládání programu je opravdu velice jednoduché. V následujících krocích je popsán nejjednodušší postup jak stáhnout a analyzovat posty z nějaké Facebookové stránky:

1. Vložit zkopírovaný odkaz na Facebookovou stránku do pole „Stránka“. (Místo url adresy je možné použít i název stránky nebo její jednoznačný identifikátor – id)
2. Vybrat složku, do které se vygenerují výsledné html soubory s výsledky analýzy.
3. Stisknout tlačítko „Start“

Takto nastavený analyzátor začne stahovat posty ze zadané stránky bez jakéhokoliv omezení a bude tak pokračovat dokud nevyprší přístupové právo této aplikace (více v kapitole 3.2). Pokud ovšem nechcete čekat tak dlouho, je možné stahování zastavit tlačítkem „Stop“.

B.2 Nastavení

Aplikace umožňuje uživateli lépe zacílit stahování postů pomocí základních filtrů, které téměř všechny naleznete na pravé straně uživatelského rozhraní. Výjimkou je nastavení stahování uzlu Graph API `feed` nebo `posts`, které leží u pole pro vložení url adresy analyzované stránky. Více informací o Graph API uzlech naleznete v kapitole 3.1.

Filtry stahování:

- Maximální počet postů – nastavení limitního počtu pro stažení postů.
- Hraniční datum – nastaví datum, do kterého se budou posty stahovat.
- Uživatel – nastavení uživatele, od kterého se jako od jediného budou stahovat posty.

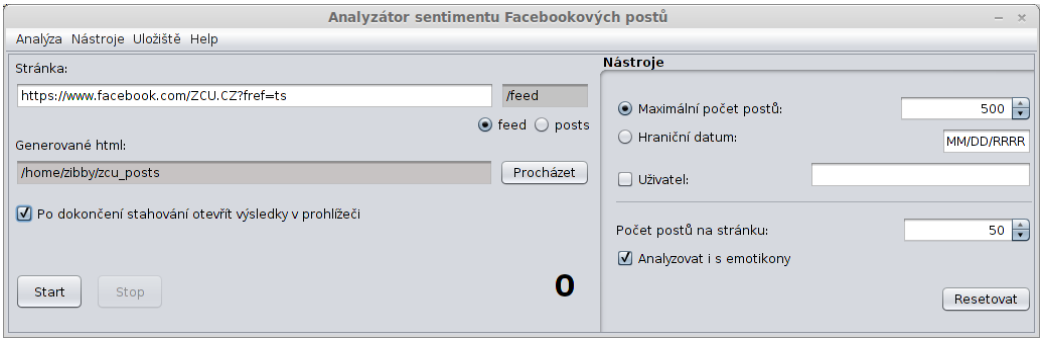
Další nastavení:

- Počet postů na stránku – nastavení počtu postů, který bude vygenerován do výsledného html na jednu stránku
- Analyzovat i s emotikony – určení, jestli se při analýze sentimentu má brát ohled na emotikony. Více informací o emotikonech a jejich vlivu na sentiment textu naleznete v kapitole 2.2.

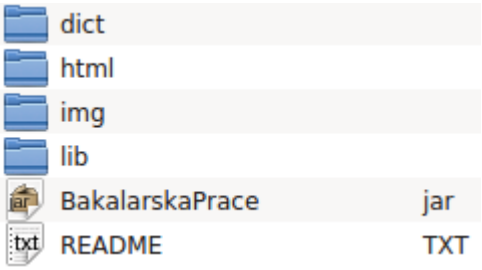
B.3 Výsledky analýzy

Jak bylo řečeno v kapitole 4 a také výše, byl jako způsob vizualizace výsledků analýzy zvolen postup, kdy je vygenerováno jednoduché html, které je poté otevřeno ve výchozím webovém prohlížeči uživatele. Na strukturu vygenerovaných souborů se můžete podívat na obrázku B.3.

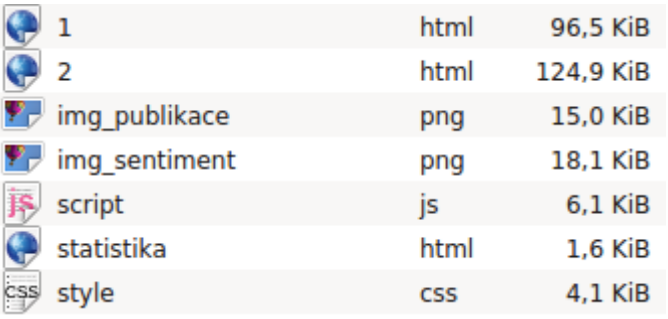
Soubory obsahující vlastní analyzované posty jsou pojmenovány jen číslem aby bylo umožněno jejich automatické generování. V každém tomto souboru je obsaženo jen tolik postů, kolik bylo nastaveno uživatelem (defaultně 50). Dále jsou ve vybrané složce vytvořeny dva png obrázky se základními údaji o stažených datech, které jsou k náhledu také v souboru `statistika.html`. Celkový vzhled html souborů je zapsán pomocí CSS3 v souboru s názvem `style.css`. Širší funkcionalitu těchto stránek pak specifikují funkce napsané v jazyku JavaScript v souboru `script.js`.



Obrázek B.1: Uživatelské rozhraní



Obrázek B.2: Struktura programu



Obrázek B.3: Struktura vygenerovaného html