VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ POSTOJŮ Z FILMOVÝCH RECENZÍ

SENTIMENT ANALYSIS FROM MOVIE REVIEWS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **DANIEL BÍLÝ**
AUTHOR

VEDOUCÍ PRÁCE **doc. RNDr. PAVEL SMRŽ, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Bílý Daniel**
Program: Informační technologie
Název: **Rozpoznávání postojů z filmových recenzí**
Sentiment Analysis from Movie Reviews
Kategorie: Algoritmy a datové struktury

Zadání:

1. Prostudujte existující zdroje dat a metody pro rozpoznávání postojů z uživatelských hodnocení filmů a televizních seriálů.
2. Navrhněte a implementujte systém, který dokáže pravidelně získávat, indexovat a analyzovat recenze filmů v češtině a angličtině.
3. Vytvořte systém pro automatickou klasifikaci shromažďovaných dat, analýzu trendů a vizualizaci výsledků.
4. Demonstrujte vytvořený systém na vhodně zvolených příkladech aktuálních úspěšných a neúspěšných filmů.
5. Vytvořte stručný plakát prezentující práci, její cíle a výsledky.

Literatura:

- dle dohody s vedoucím

Pro udělení zápočtu za první semestr je požadováno:

- funkční prototyp řešení

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Smrž Pavel, doc. RNDr., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 14. května 2020
Datum schválení: 1. listopadu 2019

Abstrakt

Tato práce je zaměřena na tvorbu systému, který je schopný pravidelně stahovat filmové recenze z webu a následně je analyzovat. Zdrojů recenzí je několik a to českých i anglických (čsfd, fdb, imdb a rotten tomatoes). Analýza sentimentu recenzí je prováděna za pomocí strojového učení. Výsledky analýz jsou zobrazovány ve webovém prohlížeči.

Abstract

This thesis is focused on creating a system which is capable of downloading movie reviews from the web and analysing them. There is several sources of movie reviews, czech and english (čsfd, fdb, imdb and rotten tomatoes). The sentiment analysis is performed using machine learning. Results of the analysis are shown in a browser.

Klíčová slova

Analýza sentimentu, stahování dat z webu, strojové učení, filmové recenze

Keywords

Sentiment analysis, web scraping, machine learning, movie reviews

Citace

BÍLÝ, Daniel. *Rozpoznávání postojů z filmových recenzí.* Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Pavel Smrž, Ph.D.

Rozpoznávání postojů z filmových recenzí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Daniel Bílý
28. července 2020

Obsah

Kapitola 1

Úvod

Rozpoznávání postojů (také známé jako analýza sentimentu nebo dolování názorů) je soubor analytických technik, jejichž cílem je extrahovat subjektivní názor z textu. Obvykle se rozlišuje mezi pozitivním, negativním a někdy i neutrálním postojem autora. Další možností je určování míry pozitivity názoru na předem definované stupnici (například často používaných jedna až pět hvězdiček). Tento úkol je samozřejmě obtížnější než prosté určování polarity. Už ze statistického hlediska je obtížnější určit správnou třídu z pěti než z pouhých dvou. Další komplikací je často nejasná hranice mezi jednotlivými třídami.

Téma analýzy sentimentu je dnes velice populární. K popularitě dopomohlo množství nestrukturovaných dat, které jsou na webu k dispozici. Těchto dat se společnosti snaží co nejvíce využít, proto do takovýchto analyzátorů investují. Je možné nalézt mnoho komerčních analyzátorů, které nabízí své služby (*Gavagai*[1], *Natural language API* od firmy Google[2] nebo *Brand24*[3].

Využívaným typem dat pro analýzu sentimentu se staly recenze a to především filmů a seriálů. Důvodem je jejich dostupnost (existují webové stránky specificky vytvořené pro recenze), množství (především filmových recenzí je nespočet) a fakt, že recenze jsou již ohodnoceny autorem (nemusí být manuálně anotovány). Právě z těchto důvodů se v práci snažím prozkoumat využitelnost klasifikátorů natrénovaných na těchto datech v jiných doménách. Dalším prozkoumaným faktorem je potřebné množství dat při trénování analyzátorů.

Tato práce má za cíl navrhnout a implementovat systém, který by byl schopný analyzovat texty filmových recenzí, a tím z nich získat názor autora k danému filmu či seriálu. Data získaná z analýzy by zpracoval a představil uživateli. Analýzu je možné provádět celou řadou metod, vybrané metody jsou tedy porovnány vzhledem k jejich přesnosti.

Výsledkem práce by měl být systém, který napomůže společnostem točící filmy i zákazníkům, kteří je sledují.

Společnosti za použití tohoto systému mohou jednoduše analyzovat názory lidí na jejich práci. Nejenže zjistí celkový názor na produkt, ale také mají možnost filtrovat názory na různé aspekty tohoto produktu. Důležitou informací získanou tímto systémem je změna názoru na daný film či seriál v čase. Společnost například může po vytvoření marketingové kampaně sledovat její dopad na názory lidí, popřípadě včas zamezit nějakému fiasku.

Zákazníkům tento systém pomůže při hledání filmu, nebo seriálu, který by se jim mohl líbit. Dále by mohl napomáhat uživatelům s doplňováním „hvězdičkového“ hodnocení při dopsání nové recenze (obsah recenze a výsledné hodnocení uživatele se občas dost liší).

[1] `https://www.gavagai.io/`
[2] `https://cloud.google.com/`
[3] `https://brand24.com/`

Text práce se skládá z následujících částí. Kapitola 2 nabízí pohled na teorii potřebnou k vytvoření tohoto systému. Prozkoumány jsou základní i pokročilé techniky analýzy sentimentu, možnosti získávání dat z webu, jejich uložení a techniky pro předzpracování dat. Kapitola 3 využívá teorii k návrhu a kapitola 4 k implementaci popisovaného systému. V kapitole 5 je vytvořený systém vyhodnocen. Aplikace běží na adrese `http://athena1.fit.vutbr.cz:8078/`.

Kapitola 2

Rozbor řešené problematiky

Tato kapitola se věnuje teoretickému popisu jednotlivých částí systému. V průběhu jsou rozebírány i případné problémy, se kterými je potřeba se vypořádat. Jejich řešení je pouze nastíněno, samotná implementace je v kapitole 4.

Prvním okruhem zájmu je způsob získávání dat z webu a jejich uložení. Dále se provede jejich analýza. Informace získané z analýzy je nutné také uložit a vhodným způsobem zobrazit uživateli (například jako tabulku nebo graf). Každému z těchto okruhů je věnována vlastní sekce.

2.1 Analýza sentimentu

V této sekci chci rozebrat základní principy a metody analýzy sentimentu. Tato analýza se snaží v textu hledat názory uživatele a určit, jestli jsou pozitivní, negativní nebo neutrální. Názory se mohou týkat produktu, služby nebo například organizace jako celku, ale také jen jednotlivých částí. Analýza sentimentu se také zabývá extrakcí emocí z textu (hněv, radost, smutek). Názory jsou obvykle emocionálně zabarvené, to napomáhá je vyhledat. Emočně zbarvený text bude pravděpodobně obsahovat slova jako „krásný“ nebo „příšerný“. Na základě tohoto faktu staví některé techniky analýzy. Objektivní komentáře typu „Film je dlouhý 120 minut“ nejsou pro tuhle analýzu zajímavé.

Výsledkem analýzy tedy může být polarita (pozitivní, negativní, popřípadě neutrální), někdy také doplněna intenzitou dané polarity (škála hodnot, například od 1 do 5).

Jak je popsáno v [29], analýza postoje se obecně dělí do třech úrovní. První analyzuje dokument jako celek (např. celý komentář), druhá úroveň je větní, která zjišťuje názor v jednotlivých větách a třetí je na úrovni aspektu. Analýza na každé z těchto úrovní obvykle vyžaduje trochu jiný přístup, některé metody analýzy jsou však využitelné na více úrovních.

Analyzování na úrovni dokumentu je vhodné provádět pokud se celý dokument vyjadřuje k jedné entitě (v případě recenzí je to daný film). Analýza se provádí za předpokladu, že daný dokument obsahuje názor.

Na úrovni větní se zjišťuje jaký sentiment mají jednotlivé věty, to dává větší detail o analyzovaném textu. Sloučením sentimentů jednotlivých vět je možné zjistit celkový sentiment dokumentu. Analýza na této úrovni souvisí s analýzou subjektivity, která rozpoznává objektivní věty od subjektivních (některé věty názor neobsahují a jak už bylo řečeno, objektivní věty nejsou pro analýzu zajímavé). Polarita názoru se tedy analyzuje až poté, co je věta označena za subjektivní. Objektivní věty se obvykle označují za neutrální.

Analýza na úrovni aspektu sleduje názory na určité části dané věci. Například ve větě „Herci byli skvělí, ale prostředí nic moc.“ se vyskytují dva aspekty filmu. Herci, ti mají hodnocení pozitivní a prostředí, které má negativní. Analýza na úrovni aspektu se dělí na dva úkoly. Prvním je nalezení zmínky o hledaném aspektu a druhým je přiřazení polarity této zmínce.

Analyzovat postoj uživatele není jednoduchý úkol. Velkou problematiku přináší lidský jazyk, který je dosti volný a kontextový. Jedním z těchto problémů je rozpoznat ironii a sarkasmus. Například z věty „Tento film je opravdu výborný!“ nelze poznat, jakým způsobem to autor myslel. Jak popisuje [11], nejzásadnější komplikací je fakt, že ironie ani sarkasmus nejsou formálně definovány a nedají se lehce poznat (například sadou pravidel). Oba tyto fenomény jsou dynamické a vyvíjí se. Ironii a sarkasmus se dá částečně poznat podle některých ustálených frází, emotikonů nebo například otazníků a vykřičníků. Dalším pomocným faktem je, že ironie a sarkasmus se častěji využívají pro záporná hodnocení a často se objevují v podobném kontextu. Ocitovaná práce se problém sarkasmu a ironie snaží řešit vytvořením párů ironických a neironických recenzí ke stejnému produktu a jejich následnou analýzou.

Zde [22] tento problém řeší využitím klasifikátorů založených na metodě pomocných vektorů a maximální entropie. Klasifikátory jsou trénovány na datech ze sociální sítě *Twitter*. V práci jsou prozkoumány různé metody předzpracování a extrakce příznaků, snaží se také zpracovat emotikony a interpunkční znaménka. Vytvořený systém detekuje ironii a sarkasmus s F1 skóre 0.92 pro angličtinu a 0.58 pro češtinu. Tento rozdíl je způsobený velikostmi trénovacích datových sad (trénovací sada pro angličtinu je mnohem větší). Při zmenšení anglické datové sady na velikost české klesne F1 skóre klasifikátoru pro angličtinu pouze na 0.73, z toho vyplývá, že v českém jazyku je obtížnější detekovat ironii a sarkasmus.

V následujících podsekcích rozebírám základní metody analýzy. Podle [8] (práce byla využita pro základní informace o metodách) jsou přístupy založené na slovníku, pravidlech nebo strojovém učení.

2.1.1 Metody využívající lexikon

Základem metod je myšlenka, že celková polarita věty nebo dokumentu se dá zjistit z polarit jednotlivých slov nebo frází. Má více možných implementací, obecně je však analýza založena na seznamu slov (frází), které jsou označeny polaritou a jejich intenzitou. Slovo „děsný“ by mělo například hodnotu -7 (polarita záporná s intenzitou 7). Naopak slovo „úžasný“ by mělo například hodnotu 6 (polarita kladná s intenzitou 6). Jakou polaritu a intenzitu by slova měly mít jsou problémy samy o sobě. Polarita je závislá na kontextu („malý pokoj v hotelu“ versus „malá čekací doba“). Určení intenzity je zase silně subjektivní, každý má k určitým slovům různé asociace.

Například v [28] je tato metoda dále rozšířena o sledování intenzifikace („docela dobrý“ versus „velice dobrý“) a negace („je dobrý“ versus „není dobrý“). Takovýchto rozšíření této metody existuje celá řada.

Přesnost metody využívající lexikon je oproti ostatním metodám více závislá na správném předzpracování dat. Normalizace, lemmatizace a stematizace velice napomáhají v přiřazení slov z textu k těm, které se vyskytují v předem vytvořeném lexikonu.

Po sestavení dostatečně velkého lexikonu nejčastěji používaných slov se vezme analyzovaný text a všem slovům se přiřadí jejich hodnota z lexikonu. Slova bez polarity (předložky, spojky apod.) se v lexikonu nenachází. Poté se jen sečtou všechny hodnoty a tím vznikne výsledné hodnocení daného textu. Výsledná hodnota se vydělí počtem slov v textu. Tímto

se celkové ohodnocení textu normalizuje a předejde se tomu, že by text byl pozitivnější jenom protože je delší.

2.1.2 Metody založené na pravidlech

Podle definice v [8], metoda využívá člověkem předem definovaných pravidel. Nejprve se celý dokument rozdělí na slova. Poté se podle daných pravidel testuje přítomnost slov, které jsou opět uložené v lexikonu. Stejně jako u metody založené na lexikonu se připočítává celkové skóre dokumentu. Pokud je celkové skóre větší nebo rovno nula je dokument pozitivní, pokud menší než nula, je dokument negativní. Výstup analyzátoru se porovná se správnou odpovědí. Pokud v lexikonu analyzátoru chybí nějaká slova z textu, která by napomohla analýze, jsou přidána do lexikonu. Analyzátor je tak schopný učit se nová slova ze vstupního textu.

2.1.3 Metody založené na strojovém učení

Těchto metod opět existuje celá řada [8] [17], každá s různými implementacemi. Základní myšlenka je vytvoření algoritmu, který je schopný ze vstupních dat vytvořit správná výstupní data bez toho, aby bylo explicitně naprogramováno, jak to má udělat.

Tohoto jsou algoritmy schopny dosáhnout za pomocí učení (trénování). Učením je zde myšleno upravování vnitřních proměnných modelu na základě vstupů tréningových dat. Vstupním datům se také říká vektor příznaků (vstupní dokument), který se skládá z jednotlivých příznaků (slov). Mimo analýzu sentimentu mohou příznaky být například hodnoty pixelů obrázku, celá nebo desetinná čísla jakýchkoliv hodnot, popřípadě pravděpodobnosti. Každá metoda strojového učení se trénuje trochu jiným způsobem, základní myšlenka úprav vnitřních proměnných modelu však zůstává.

Mezi jedny z nejpoužívanějších algoritmů strojového učení pro analýzu sentimentu patří metody podpůrných vektorů (angl. *support vector machines*), pravděpodobnostní modely a hluboké učení.

Jak popisuje [24], **metoda podpůrných vektorů** funguje na principu rozdělení N rozměrného prostoru vstupních příznaků pomocí nadroviny. V případě analýzy sentimentu je prostor obvykle rozdělen lineárně pro zamezení přeučení. Každé slovo v tomto prostoru představuje jeden rozměr. Počet slov v daném textu pak představuje hodnotu v tomto rozměru. Určitý text je tedy reprezentován bodem v prostoru, jeho souřadnice jsou určeny slovy a počtem těchto slov. Trénování při využití tohoto algoritmu spočívá v nalezení správného rozdělení na podprostory (například jeden podprostor pro pozitivní a jeden pro negativní sentiment) podle shluků pozitivních a negativních textů (bodů).

Po procesu trénování je analyzovaný text namapován do vytvořeného prostoru, podle toho v jakém podprostoru skončí je jeho sentiment prohlášen za pozitivní nebo negativní.

Pravděpodobnostní modely [12] [25] předpovídají pravděpodobnosti všem možným výstupům na základě současného vstupu a předchozí tréningové sadě. Výstup s nejvyšší pravděpodobností je prohlášen za správný.

Mají hned několik výhod. Jednou z nich je jednoduchost, to znamená, že jim stačí malá tréningová sada a rychle se trénují (oproti například neuronovým sítím). Další výhodou je poměrně velká přesnost, a to i když jsou trénovány na malém počtu dat. Neposlední výhodou je fakt, že mimo výslednou polaritu vrací i jistotu dané předpovědi. Obvykle se tedy využívají tam, kde je menší tréningová sada, není potřeba až tak vysoká přesnost nebo je potřeba mít rychle natrénovaný model. Neuronové sítě v porovnání potřebují více tréningových dat a více času na trénování, ale jsou přesnější.

Pravděpodobnostních modelů a jejich implementací je samozřejmě několik. Nejznámější z nich je naivní Bayesův klasifikátor, proto se na něj chci nyní zaměřit.

Klasifikátor se může lišit podle toho, jaká je využita implementace a s jakými daty pracuje. Například při práci s výškou lidí by se muselo vzít v potaz rozložení výšky lidské populace. To se řídí normálním rozložením, takže při výpočtu pravděpodobností se využije právě vzorec pro normální rozložení. Při práci s textem stačí pouze počítat výskyty slov.

Klasifikátor potřebuje tréningová data obsahující vstupní příznaky i hledaný výstup pro vymodelování pravděpodobností.

Již podle názvu klasifikátoru se dá zjistit, že pracuje na základě Bayesovy věty. Jak přesně ji využívá popisuji dále. Naivní je, protože je pro něj každé slovo nezávislé na ostatních. Nepracuje tedy s větami, ale se seznamem individuálních slov. Jak popisuje [12] předpoklad by byl, že přesnost kvůli naivitě nebude moc velká, ale praktické výsledky ukazují opak.

Bayesova věta:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.1}$$

Pro výpočet klasifikace se za B dosadí vstupní seznam příznaků a za A se postupně dosazují jednotlivé možné výstupy klasifikace. Výstupní třída s nejvyšší pravděpodobností je prohlášena za správný výstup klasifikátoru. Protože se pro výpočty pravděpodobností jednotlivých výstupů liší pouze čitatele zlomku, jmenovatele jsou vynechány. Na výsledku to nic nezmění (pouze porovnáváme která pravděpodobnost je vyšší).

Po dosazení by to tedy mohlo vypadat takto(pos je označení pro pozitivní sentiment, neg je označení pro negativní sentiment):

$$P(pos|Film\ byl\ super) = P(Film\ byl\ super|pos) \cdot P(pos) \tag{2.2}$$

$$P(neg|Film\ byl\ super) = P(Film\ byl\ super|neg) \cdot P(neg) \tag{2.3}$$

$P(pos)$ a $P(neg)$ jsou pouze poměry výskytů jednotlivých tříd v trénovací sadě. Teď už tedy stačí vypočítat $P(Film\ byl\ super|pos)$ a $P(Film\ byl\ super|neg)$. V podstatě by stačilo spočítat kolikrát se v tréningové sadě vyskytuje věta „Film byl super“ pod třídou pos a kolikrát pod třídou neg. Problémem je, že věta se nemusí v tréningové sadě vůbec objevit. Tento problém řeší naivita klasifikátoru. Pravděpodobnost věty se rozdělí na násobek pravděpodobností jednotlivých slov:

$$P(Film\ byl\ super) = P(Film) \cdot P(byl) \cdot P(super) \tag{2.4}$$

Pravděpodobnost jednotlivých slov se vypočítá:

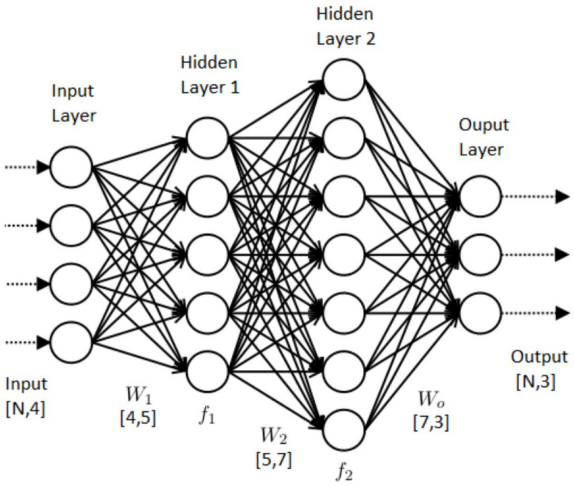
$$P(Film|pos) = \frac{x}{y} \tag{2.5}$$

Kde x je počet výskytů slova "Film"v textech třídy pos a y je počet všech možných slov ve třídě pos. Tohle samozřejmě jen přesune daný problém z vět na slova, dané slovo se nemusí objevit v trénovací sadě. To by nám dalo pravděpodobnost slova 0 a protože násobíme, pravděpodobnost celé věty by byla také 0. Problém se dá obejít využitím Laplaceova vyhlazování. Ke každému výskytu přičteme 1, aby nikdy nebylo 0 a ke jmenovateli přičteme počet všech unikátních slov přes obě třídy (aby pravděpodobnost nebyla větší než 1).

Tímto se vytvoří tabulka, kde ke každému slovu vyskytujícímu se v trénovací sadě jsou přiřazeny pravděpodobnosti každé třídy (pos a neg). Při analýze stačí jen vynásobit

pravděpodobnosti slov vyskytujících se v analyzované větě. Samozřejmě jednou se násobení provede pro třídu pos a po druhé pro třídu neg. Porovnáním výsledných pravděpodobností násobků se zjistí, která odpověď je pravděpodobnější.

Hluboké učení je dnes největší novinkou nejen při analýze sentimentu [32] [5].



Obrázek 2.1: Obrázek neuronové sítě. N značí vstupní a výstupní neurony, W značí váhy spojů mezi neurony a f jsou „výpočetní“ neurony. Převzato z `https://medium.com/coinmonks/the-artificial-neural-networks-handbook-part-1-f9ceb0e376b4`

Využívá se k řešení problémů, které nebylo možné dříve vyřešit (například klasifikace obrázků, rozpoznání slov z lidské řeči a podobně). Neustále se nachází nová využití pro tuto metodu (od překládání jazyků v reálném čase po řízení automobilů). V oblasti analýzy sentimentu je tato metoda také přelomová. Za správných podmínek dosahuje nejvyšší přesnosti ze všech algoritmů, je však doménově závislá a náročná na trénování.

Koncept je poměrně starý, počátky již v padesátých letech minulého století. Trénování neuronových sítí je výpočetně velice náročné a potřebují velké množství tréningových dat, proto je rozvoj možný až nyní díky výkonnější výpočetní technice (převážně grafickým kartám) a přístupu k obrovskému množství dat.

Jako spousta dalších algoritmů (hejno částic, mravenčí kolonie a například genetický algoritmus) je i tento inspirován přírodou. Hluboké učení se snaží napodobit funkci biologického mozku. Neuronová síť je složena z množství neuronů a spoji mezi nimi viz. obrázek 2.1. Jako v biologickém mozku, každý neuron je schopný zpracovat signál a poslat ho pomocí spojů (synapsí) dalším neuronům. Neuron je v tomto případě vlastně jen obyčejná sčítačka. Všechny vstupy do neuronu jsou vynásobeny váhou daného spoje a sečteny. K tomuto výsledku se obvykle ještě přičte práh. Po přičtení prahu je použita aktivační funkce (například *sigmoid*, *tanh* nebo *ReLu*) a výsledek je výstupními spoji poslán dalším neuronům. Hodnoty prahů a vah spojů jsou z počátku nastaveny náhodně a v průběhu trénování se upravují.

Hluboké učení je využití neuronových sítí v několikavrstvé architektuře. Architekturu je samozřejmě možné zapojit více způsoby. Pro analýzu sentimentu se nejvíce používají následující architektury neuronových sítí:

- Rekurentní - Třída neuronových sítí, kde jednotlivá propojení mezi neurony tvoří cyklus. Každý cyklus je tvořen jednou vrstvou neuronů. Oproti klasické dopředné neuronové síti má vnitřní paměť. Je tedy vhodná ke zpracování sekvenčních dat (mimo jiné

i textu). V paměti je uložen aktuální stav zpracování, tento stav je upravován každým příchozím slovem. Stav má tedy uchováno vše, co se postupně zpracovalo. Tento typ neuronových sítí trpí problémem mizejícího a explodujícího gradientu, toho se snaží vyvarovat úpravy této architektury (obousměrné anglicky *bidirectional* architektury a *LSTM*).

- *LSTM* (*long short term memory*) - Je speciálním typem rekurentní neuronové sítě, která řeší problém mizejícího gradientu. Oproti rekurentní síti má každý cyklus čtyři vrstvy neuronů, které mezi sebou komunikují. Navíc má dva vnitřní stavy místo jednoho. Jednotlivým vrstvám se říká brány a jsou následující: „zapomínající“, „pamatující“, „vstupní“ a „výstupní“. Problém mizejícího gradientu je vyřešen kombinací těchto bran a využitím jednoho vnitřního stavu navíc.

- Rekurzivní - Tento typ neuronových sítí se obvykle využívá ke zpracování dat v acyklickém grafu (stromu). Je vlastně generalizací rekurentních neuronových sítí. Neuronová síť postupně prochází stromovou strukturu od listů směrem ke kořenu a vytváří rodičovské uzly kombinací jednotlivých tokenů.

Učení neuronových sítí je možné provádět s učitelem, bez učitele a zpětnovazební. Zpětnovazební učení se při analýze sentimentu nevyužívá, proto se na něj nebudu zaměřovat.

Učení s učitelem je proces při kterém je model učen hledat k danému vstupu korektní výstup. Model tedy hledá funkci f takovou, která bude mít správný výstup $f(x)$ pro co nejvíce vstupních vzorků x. Pokud se hledá konkrétní třída z předem dané množiny hovoříme o klasifikaci, pokud se hledá hodnota v určitém rozsahu mluvíme o regresi. Korektní výstupy jsou předem definovány lidmi. Je potřeba mít dostatečně velký počet trénovacích dat, které obsahují vstup a k němu hledaný výstup. Toto může být problematické, protože vytvoření takto označených dat není jednoduché, někdy nemožné.

V případě analýzy sentimentu bude vstupem text a výstupem hledaný sentiment (pozitivní nebo negativní, popřípadě třídy hodnot). Takto označená data se předají modelu, který se na základě textu pokusí předpovědět výstup (na začátku procesu trénování pravděpodobně chybně). Předpověď je poté porovnána se správnou odpovědí. Porovnání je prováděno tak zvanými ztrátovými funkcemi. Ztrátových funkci je několik, každá se používá pro řešení jiného problému.

Podle chybovosti se modelu upraví proměnné (váhy spojů a hodnoty prahů), aby byl příště přesnější (využívá se derivací pro zjištění správné úpravy). Po dostatečně dlouhém trénování se model naučí potřebné generalizace k tomu, aby byl schopný předpovídat korektně výstup i na datech, které ještě před tím nikdy neviděl.

Učení bez učitele využívá pro trénování pouze vstupy [13]. Hledaný výstup není označen. Analyzátor se snaží ve vstupních datech nalézt strukturu dat a vztahy mezi jednotlivými příklady. Výstupem bývají shluky, kolem kterých se data pohybují. Všechna přiložená data při tréningu se tak strukturují a je možné v nich nalézt podobnosti.

Podle [31] je možné pro analýzu sentimentu využít učení bez učitele následujícím způsobem. Využívá se faktu, že negativní nebo pozitivní slova bývají vždy obklopena podobnými slovy. Například slovo „dobrý“ je vždy obklopeno obdobnými slovy jako slovo „úžasný“. Tímto způsobem se vytvoří shluk pozitivních a shluk negativních slov. Podle vzdálenosti od středu daného shluku je možné zjisti jak moc pozitivní / negativní dané slovo je. Postupně tak můžeme ohodnotit pozitivitu celého textu.

2.2 Pokročilé metody zpracování přirozeného jazyka

Tato kapitola nabízí přehled novějších technik, technologií a systémů. Základní myšlenka konceptu hlubokého učení je spolu s dalšími základními algoritmy strojového učení popsána v kapitole 2.1.3. Na základním konceptu hlubokého učení staví mnoho dalších technologií, které zde budou popsány. Samozřejmě existují nové techniky, technologie a systémy, které hlubokého učení nevyužívají. Z osobního zájmu (technologie hlubokého učení mi přijdou zajímavé) se ale soustředím právě na ně. Základní přehled existujících technologií byl nalezen v [32].

2.2.1 Vnoření slov

Anglicky známé jako *word embedding*. Vnoření slov je technika jazykového modelování a reprezentace příznaků. Slova ze slovníku jsou převedena na vektor desetinných čísel. Tento vektor je obvykle vytvořen převedením z řídkého vektoru 1 z N (anglicky *one-hot vector*). Převedený vektor již není řídký, obsahuje nenulové hodnoty ve všech dimenzích (pokud tedy určitá hodnota v dimenzi není reprezentována právě nulou). Zároveň je dimenzí mnohem méně.

Základní myšlenkou této reprezentace je, že každá dimenze nově vytvořeného vektoru představuje nějakou vlastnost daného slova, podobná slova tedy budou mít podobné číselné reprezentace. Zajímavostí takto vytvořených číselných reprezentací jsou výsledky po matematických operacích mezi slovy. Vlastnost je často vysvětlena následujícím příkladem. Když se vezme vytvořená vektorová reprezentace pro slovo „král“, odečte se od ní vektor slova „muž“ a přičte vektor slova „žena“, výsledkem by měla být vektorová reprezentace podobná slovu „královna“.

Vnoření slov je možné využitím hlubokého učení implementovat například metodami *CBOW* [18] (*continuous bag of words*), SG [19] (*Skip-Gram*) a GloVe (*Global Vector*). Metody *CBOW* a *SG* jsou dnes implementovány pod populárním systémem zvaným *Word2Vec*. Tento systém využívá vlastností obou metod pro co nejlepší výsledky. Metoda *CBOW* je přesnější pro menší datové sady, naopak *SG* je přesnější trénováním na větších datových sadách.

Obě metody trénují neuronové sítě velice podobným způsobem. Trénovací data není potřeba anotovat. Metoda *CBOW* učí neuronovou síť předpovědět slovo podle jeho kontextu (kontext je vstupem a slovo výstupem). Naopak metoda *SG* předpovídá kontext podle daného slova (slovo je vstup, kontext výstup). Jeden trénovací příklad by tedy mohl vypadat dle obrázku 2.2.

Obrázek 2.2: Ukázka trénovacího příkladu.

2.2.2 Mechanismus pozornosti pro neuronové sítě

Tato technika pomáhá řešit problém se vzdálenými závislostmi v sekvenci dat (například závislost mezi prvním a posledním slovem ve větě). Tento problém se snaží řešit oboustranné rekurentní neuronové sítě a LSTM sítě, v praxi je to však stále problematické.

Tento mechanismus je inspirován vizuální pozorností lidí. Stejně jako lidský zrak se tato metoda soustředí na určitou část ve vysokém rozlišení a okolí je v nízkém rozlišení. Část pod vysokým rozlišením se postupně posouvá v čase. Mechanismus pomáhá trénovanému modelu naučit se které části dat věnovat pozornost na základě vstupu a současného stavu zpracování dat.

Mechanismus byl například využit v systému pro překlad mezi jazyky [6]. Pozornost byla využita pro výběr slov relevantních při překladu. Model se postupně soustředí na jednotlivé části překládané věty. Samozřejmě nestačí jít slovo po slovu, protože jazyky často vyjadřují stejnou věc jiným počtem slov. Model se tedy musí naučit toto mapování. Například při překladu z angličtiny do francouzštiny se slovo *destruction* překládá jako *la destruction*, to znamená, že při překladu jednoho slova v angličtině se musí soustředit na vytvoření dvou slov ve francouzštině.

2.2.3 Paměťová síť

Anglicky známé jako *memory networks*. V této práci [30] byla síť představena jako systém schopný přijímat logická tvrzení a podle nich poté odpovídat na otázky. Systém je složen z několika komponent. Jednou z těchto komponent je paměťová část. Paměťová část je oproti například rekurentním neuronovým sítím mnohem větší a dlouhodobější. Jednotlivé části mohou být implementovány jako neuronové sítě, v práci byly prozkoumány i další metody. Paměťová část se chová jako dynamické úložiště znalostí získaných z jednotlivých tvrzení. Ostatní komponenty jsou následující:

- I (*input feature map*) - převede příchozí data do vnitřní reprezentace
- G (*generalization*) - zpracovává příchozí informace (tvrzení), upravuje podle nich paměť znalostí. Má schopnost zestručnit a zobecnit všechny získané znalosti.
- O (*output feature map*) - z již získaných znalostí a nového vstupu vytváří nový výstup. Výstup je stále ve formátu vnitřní reprezentace.
- R (*response*) - převádí výstup z vnitřní reprezentace do požadovaného formátu, například text nebo akce.

Síť nejprve dostane několik vět a poté otázku, na kterou má odpovědět. Systém je schopný nalézt ve větách potřebné informace a podle nich vytvořit odpověď. Komponenta I přečte jednotlivé věty a přeloží je do vnitřní reprezentace sítě. Poté komponenta G upraví paměť podle nově získaných informací. Po zpracování všech vstupních vět jsou věty uloženy do matice vět. Otázka je také nejprve převedena do vnitřní reprezentace, poté komponenta O nalezne potřebné informace ve vnitřní paměti a vytvoří odpověď. Nakonec je odpověď převedena komponentou R.

2.2.4 BERT

Je zkratka z anglického ***B**idirectional **E**ncoder **R**epresentations from **T**ransformers*, byl vytvořen společností Google pro lepší pochopení uživatelských vyhledávání [9]. Tento systém má za cíl naučit se modelovat (dá se říci „pochopit“) jazyk. Technická inovace spočívá ve využití oboustranného trénování transformerů (z anglického *transformer*), což jsou modely využívající koncept pozornosti. Předchozí systémy se obvykle na text dívaly jako na sekvenci slov jdoucí zleva doprava, popřípadě v kombinaci s pohledem zprava doleva. Výsledky

systému *BERT* však ukazují, že modely trénované oboustranně mají hlubší pochopení kontextu a sekvencí slov. Vstupní sekvence slov se čte jako celek, to napomáhá modelu se naučit kontext určitého slova na obou stranách. Trénování modelu probíhá následujícími metodami:

- maskováním slov - Před předáním trénovací sekvence slov modelu se nahradí 15 % slov maskovacím tokenem. Model se poté učí předpovědět maskovaná slova podle ostatních slov v sekvenci.
- předpovídáním následující věty - Model postupně dostává dvojice vět. Polovina dvojic vět na sebe navazuje, druhá polovina ne. Model se učí předpovídat jestli druhá věta patří za první či nikoliv.

Model dále využívá toho, že je předem natrénovaný na obrovské sadě dat. Při vytváření modelu pro nový úkol se obvykle musí natrénovat od úplného počátku, to může trvat i desítky hodin. Tento problém řeší právě předem natrénovaný *BERT*, který pro specifický úkol stačí doladit [27]. Autoři systému doporučují dolaďování dvěma až čtyřmi epochami, *BERT* byl v porovnání trénován stovky hodin.

BERT se dnes využívá na množství úkolů zpracování přirozeného jazyka, ve velkém počtu z nich má nejlepší výsledky. Například na těchto stránkách[1] lze vidět všechny jeho využití.

2.3 Získání dat z webových portálů

Před samotným získáním dat je dobré vědět kolik a jakých dat bude potřeba. Proto je vhodné nejprve prozkoumat metody analýzy. Z porovnání metod [8] je patrné, že největší přesnost mají analyzátory využívající strojové učení. Z tohoto důvodu bych se přikláněl k využití těchto analyzátorů. Jejich nevýhodou je nutnost trénování.

Jak je napsáno na tomto blogu [2], při vytváření analyzátoru na bázi umělé inteligence/strojového učení je potřeba obrovské množství dat pro jeho trénování. O trénování a analýze jsem již psal, nyní stačí informace, že budou potřeba data pro trénování analyzátoru. Těchto dat by mělo být co nejvíce, protože čím větší vzorek dat je použit, tím více vzorů vyjadřování se analyzátor naučí a poté je při samotné analýze přesnější.

Dalším důvodem, proč je potřeba velké množství dat je mimo trénování samotná analýza. Z velkého počtu názorů je patrnější celkový názor na produkt a je pravděpodobnější získat více pohledů na věc.

Z předchozí analýzy tedy vyplynulo, že bude potřeba co nejvíce dat, tato data je možné získat z internetových stránek. Na internetu je velké množství uživatelů a všichni nějakým způsobem přispívají k jeho obsahu [23]. Konkrétní počet uživatelů na internetu pro rok 2019 je 4,4 miliard, takové množství uživatelů je schopné vyprodukovat obrovské množství dat každou minutu, konkrétní čísla pro největší sociální média jsou na citovaném blogu.

Uživatelé přispívají na sociálních médiích, speciálních stránkách určených pro recenze nebo přímo na stránkách prodejců. Díky tomu je vytvářeno obrovské množství nestrukturovaných dat, které je možné využít pro analýzu názorů na téměř jakékoliv téma.

Mimo kvantitu dat je také důležitá jejich kvalita. Pokud je analyzátor trénován na špatných datech, nedá se předpokládat, že by měl velkou přesnost. Zpracování dat pro zvýšení jejich kvality je popsáno v kapitole 2.5. Data (obzvlášť ta stažená z internetových diskuzí)

[1] `http://nlpprogress.com/english/sentiment_analysis.html`

nejsou vždy validní. V případě filmové recenze může být napsána člověkem, který prostě nemá daný žánr rád, nebo může jít o *internetového trolla* [14]. Ve zkratce, internetový troll je někdo, kdo se snaží narušit věcnou konverzaci nad tématem. Obvykle se snaží z ostatních uživatelů dostat emocionální reakci. Validita příspěvku se však nedá jednoduše poznat, proto se musí předpokládat, že každý příspěvek je svým způsobem validní (každý má svůj názor, který se musí vzít v potaz). Při výsledné analýze nevalidní příspěvky příliš nevadí (s nimi se obvykle počítá). Všechny názory jsou důležité, protože utváří celkový pohled na daný film. Pro společnosti je vhodné sledovat všechny názory, aby věděli jakým směrem se dále ubírat, nevalidní příspěvky však při trénování analyzátoru mohou zhoršit jeho přesnost. Validita příspěvků se na některých webech dá poznat pomocí hodnocení („lajků“) od ostatních uživatelů (tato možnost však není vždy).

Pro získání dat z webových stránek je hned několik možností. Některé webové stránky mají své rozhraní pro programování aplikací (z anglického application programming interface, zkratka API), pomocí kterého lze požadovaná data získat. Jak píše [20], API je část serveru odpovědná za přijímání a odesílání požadavků a dat. K tomu jsou obvykle využívány metody GET nebo POST.

Data jsou odeslána v předem domluveném formátu jako například JSON. Na rozdíl od klasického webového serveru tedy neodesílá HTML dokument, ale pouze požadovaná data. Při práci s API většinou stačí využít jeho konkrétní knihovnu nebo url adresu a jejich předdefinované funkce. Každá taková funkce má svou dokumentaci, je tedy jasné co funkce vyžaduje a co vrátí. Vše bývá již připraveno pro pohodlí uživatele, je tedy jednoduché se v datech zorientovat, poté je zpracovat a například uložit. Většina stránek však žádné API nemá, nebo k němu dávají přístup pouze ve speciálních případech na zažádání (k přístupu bývá potřeba speciální klíč). Musí se tedy využít jiné nástroje.

Obecně se získávání dat z webu mimo použití API říká *web scraping* [20]. *Web scraping* je soubor technik, které lze využít k získání požadovaných (obvykle nestrukturovaných) dat z webových stránek. Mezi tyto techniky patří způsoby pro nalezení hledaných dat i jejich samotné stažení a následné formátování. *Web scraping* zahrnuje široký rozsah programovacích technologií mezi které patří například analýza dat, zpracování přirozeného jazyka a zabezpečení informací.

Když se provádí automatizovaný *web scraping*, webové stránky můžou být děleny do dvou typů. Stránky používající JavaScript pro vytváření (obvykle dostahování) obsahu, a stránky bez něj. Při manuálním kopírování dat ze stránky přítomnost JavaScriptu proces neovlivňuje.

Stránky bez JavaScriptu jsou po přijetí požadavku serverem celé odeslány v těle HTTP odpovědi serveru jako HTML dokument. Proto pouze stačí poslat na server HTTP dotaz a přijatý HTML dokument zpracovat podle potřeby.

Stránky používající JavaScript pro doplnění obsahu fungují jinak. Nemusí vždy přímo obsahovat hledaná data. HTML dokument obsahuje základní kostru stránky a část dat. Zbylá data jsou načítána dynamicky pomocí skriptů, které se spouští až v prohlížeči uživatele. Podle toho, jaká data uživatel potřebuje (na co klikl, jestli je přihlášený apod.) JavaScript zašle další požadavky na server. Této technologii se říká AJAX (zkratka z anglického asynchronous JavaScript and XML, neboli asynchroní JavaScript a XML). Od serveru se vrátí potřebná data obvykle ve formátu JSON a stránka se z těchto částí dat seskládá až v prohlížeči. Proto není možné na server zaslat HTTP požadavek a obdržet zpět stejný HTML dokument jako ten vykreslovaný v prohlížeči. Pro stahování z JavaScriptových stránek je možné využít *Reverse Engineering* JavaScriptu nebo jeho *Rendering*. *Reverse Engineering* se dělá za pomoci sledování požadavků JavaScriptu a následného zaslání

stejných požadavků. Přijatá data se následně seskládají do celku. Druhá metoda je vykreslení a zpracování požadované stránky za pomocí nástroje jako je Selenium. Tento nástroj využívá prohlížeč pro vykreslení stránky a vlastně simuluje návštěvu stránky uživatelem. Po vykreslení webové stránky stačí seskládaný HTML dokument zpracovat podle potřeby.

Jak je napsáno na [33] a [3] mezi konkrétní způsoby extrakce dat patří jejich manuální kopírování , *HTML parsing*, *DOM parsing*, využití *XPath*, *google sheets*, a *text pattern matching*.

Jednotlivé techniky stahování a extrakce hledaných dat jsou popsány v následujících podkapitolách.

2.3.1 Manuální kopírování dat

Anglicky se této technice říká *copy-pasting*. Technika spočívá v prostém kopírování chtěných dat ze zdroje a jejich vkládáním do datového úložiště. Při větším objemu dat tato technika zabere mnoho času a úsilí, práce je velice repetitivní. Na druhou stranu je tato metoda jedinou použitelnou pokud je webová stránka pod ochranou proti stahovacím skriptům. Efektivní je také při získávání pouze malého množství dat. Vzhledem k tomu, jak je tato technika pomalá a neefektivní se moc nevyužívá. Všechny ostatní metody popsané níže jsou automatizované.

2.3.2 *HTML parsing*

Obvykle se dělá za pomocí knihoven jako je například BeautifulSoup. Metoda spočívá ve stažení celého HTML dokumentu a jeho následném prohledání. Využívá se *tagů* a stromové struktury HTML dokumentu. Před napsáním skriptu, který extrahuje hledaná data je nutnost znát stromovou strukturu HTML dokumentu i názvy klíčových uzlů stromu. Strom se poté postupně prochází a extrahují se hledaná data. Cílí na HTML stránky, lineární i zanořené. Tato metoda je velice rychlá a jednoduchá. Využívá se pro extrakci textu, odkazů zdrojů (např. obrázků) a podobně.

2.3.3 *DOM parsing*

Document object model zkráceně DOM, definuje styl, strukturu, ale také obsah dokumentu. Tato technika je schopná získat podrobný pohled na strukturu stránky. Skripty mohou nalézt uzly dokumentu, které obsahují potřebná data a poté pomocí nástroje jako je XPath je extrahovat. Metoda funguje na podobném principu jako *HTML parsing*, místo HTML dokumentu však zpracovává DOM. DOM je oproti HTML dokumentu dynamický, takže tahle metoda je použitelná i když stránka obsahuje části generované dynamicky.

2.3.4 *google sheets*

Jednou netradičnější, ale přesto efektivní metodou je využití google tabulek. Tabulky mají funkci s názvem *IMPORTXML*, které stačí předložit odkaz na danou stránku. Pokud stránka není chráněna, funkce vrátí XML strukturu této stránky i s jejím obsahem.

2.3.5 *text pattern matching*

Využívá regulárních výrazů k extrakci hledaných dat (například UNIXový grep). Klasicky se tato metoda pojí s nějakým skriptovacím jazykem jako je Python nebo Perl. Regulární

výraz je speciální textový řetězec, který pomocí předdefinovaných značek určí hledaný vzor textu.

Mimo tyto metody existují další dostupné nástroje pro extrakci dat jako je například cURL, Wget, HTTrack, Node.js a další.

2.4 Možnosti uchování získaných dat

Data je možné ukládat v jejich nestrukturované (původní) formě, nebo se z nich extrahují pouze potřebné části. Od tohoto se bude odvíjet i způsob uložení daných dat. Pokud data nemají žádnou strukturu, ukládání do serializované podoby nebo databáze by pouze přidalo režii a nijak by to na rozdíl od prostého uložení do souboru nepomohlo. Na druhou stranu, po zpracování dat do strukturované podoby, je vhodné využít serializačních formátů nebo databáze. Takový způsob uložení napomůže v pozdějším vyhledávání a analýze těchto dat.

Zde bych se chtěl věnovat různým možnostem ukládání stažených dat. Nejčastěji stahovaná a ukládaná data budou textového typu, proto bych chtěl rozebrat právě možnosti ukládání textu (využitelné však i pro jiná data). Pro ukládání textových dat jsou v podstatě dvě možnosti. Data uložit přímo do textového souboru nebo využít nějakého databázového systému. Detaily v podkapitolách.

2.4.1 Uložení do souboru

Do souboru je možnost ukládat data ve zdrojové podobě (například ukládání celých HTML dokumentů zdrojových stránek). Data můžou zůstat v originálním formátu pro pozdější zpracování, nebo jsou nějakým způsobem předzpracovány (odstranění HTML značek a ponechání čistého textu) a poté uloženy. Tato metoda je však většinou nepraktická, protože se v takovýchto datech nedá jednoduše vyhledávat a data nejsou tak přehledná.

Pro ukládání do souboru se proto obvykle využije serializace. Serializace je proces převádění strukturovaných dat do formátu, který umožňuje uložení nebo přenos těchto dat a jejich následné znovupoužití. Serializovaná data mohou být v souboru uložena v textové podobě (čitelné i pro člověka) nebo v binární podobě (bez deserializace čitelné pouze pro počítač). Serializovaná data se ukládají persistentně, to znamená že se data neztratí po ukončení programu nebo například výpadku proudu. Takto uložená data je díky jejich jednotnému formátu možné použít i na jiné počítačové platformě (jiné součástky, jiný operační systém). V serializovaných datech se vyhledává mnohem lépe než v těch nestrukturovaných díky ukládání částí dat pod názvy atributů.

Mezi nejpoužívanější serializační formáty patří JSON, XML, YAML, CSV a TSV.

2.4.2 Uložení do databáze

Databáze je organizovaná kolekce dat obecně uložená v elektronické podobě na počítačovém systému (obvykle serveru) [1]. Implementačně je databáze vlastně kolekce souborů, do kterých se ukládají data. Rozdíl oproti klasickému uložení do souboru je takový, že zde se o tyto soubory stará systém řízení báze dat (SŘBD). SŘBD slouží jako rozhraní mezi databází (kolekcí souborů) a jejími koncovými uživateli nebo programy, které s databází komunikují. SŘBD Umožňuje uživateli pohodlnou práci s daty. Dále je díky tomuto systému možné určovat jakým způsobem budou data organizována a optimalizována (například indexací). SŘBD napomáhá se správou databáze sledováním transakcí v databázi, monitorováním výkonu, zálohou a obnovením databáze. Tento systém dále umožňuje autentizaci a autori-

zaci pro ochranu uložených dat. Databáze může být vytvořena lokálně nebo na serveru, se kterým se komunikuje přes internet.

Typů databází je několik, nejběžněji se používá relační a objektová. Data jsou v relační databázi organizována jako série tabulek, kde řádky tabulky jsou jednotlivé záznamy (například záznam o osobě) a sloupce tabulky jsou jednotlivé atributy sledované u tohoto záznamu (jméno, věk a pohlaví osoby). Toto umožňuje jednoduše vyhledávat, upravovat nebo přidávat data. Mimo tyto jednoduché akce existují takzvané agregační funkce (například dotaz na průměrný věk osoby). Databáze nejsou stěžejním tématem této práce, proto jsem popsal pouze úplné základy. Databází je mnohem více typů a používají různé jazyky (SQL, NOSQL apod.), ale dále to tu nechci rozvádět.

2.5 Předzpracování textu pro následnou analýzu

Předzpracování textu (anglicky preprocessing) je množina kroků, které se provádí s daty před tím, než jsou analyzována.

Při použití analyzátoru založeném na hlubokém učení je povinný pouze krok převedení textu do číselné podoby. Důvod je takový, že tyto modely jsou schopny pracovat pouze s čísly [15]. Konkrétní metody převedení textu na číselnou reprezentaci budou popsány dále. Většina ostatních metod převod na číselnou podobu nepotřebuje.

Další kroky předzpracování nejsou pro funkčnost analyzátorů klíčové, avšak tyto kroky napomůžou s přesností analýzy (přesností je zde myšlen poměr mezi počtem správných předpovědí analyzátoru a celkovým počtem předpovědí).

Dalším z cílů předzpracování textu je převést vstupní text do predikovatelné, analyzovatelné podoby a odstranit šum. Šumem se v tomto kontextu myslí všechen nesmyslný text nebo znaky. Šum je pro analýzu irelevantní popřípadě škodlivý. Pokud se do analyzátoru vloží nekvalitní data, nedají se očekávat kvalitní výsledky. Odstraněním šumu a normalizací slov dosáhneme dvou cílů.

Za prvé, analyzátory mají obvykle limitovanou velikost slovníku. Odstraněním irelevantních slov nebo znaků a sloučením více tvarů slova do jednoho necháme ve slovníku prostor pro užitečnější slova. Za druhé, při výsledné analýze (analyzovaný text se také předzpracuje) bude analyzátor považovat různé tvary jednoho slova za stejné. Nestane se tedy, že by ve slovníku měl slovo „dobrý“ a slovo „dobry“ by nerozpoznal.

Předzpracování textu tedy může být rozděleno na normalizaci (úpravu textu do normální formy), tokenizaci (rozdělení textu na atomické části, těmi může být slovo nebo věta) a převedení do číselné podoby. Text převedený do číselné podoby je již možné použít v analyzátoru založeném na hlubokém učení.

2.5.1 Normalizace

Normalizací je v kontextu NLP myšleno převedení textu do normální (kanonické) formy [15]. Cílem tohoto kroku je z textu odstranit všechny nechtěné znaky a slova převést do správné formy. Text je převeden do čisté formy odstraněním všech nepotřebných řetězců a znaků. Tento krok je velice důležitý obzvlášť pokud je použit text z internetových diskuzí. Tento text je velice často psán nespisovně, obsahuje různé speciální znaky, gramatické chyby a slangové výrazy. Normalizací se všechna tato slova převádí do jednoho (obvykle jejich spisovného) tvaru. Důvodem je to, že po převedení slov do číselné reprezentace by se slova „film“,„fiilm“ a „Film“ brala jako úplně odlišná. Zvláštní nepotřebné znaky se obvykle úplně odstraňují. Odstraňování je prováděno například regulárními výrazy.

Při normalizaci se musí dbát na to, co je cílem následné analýzy. Pokud cílem analýzy by bylo předpovědět emoce autora textu, bylo by velice nevhodné převádět všechna písmena na malá a odstranit speciální znaky, ze kterých mohou být seskládány emotikony. Velikost písmen a emotikony totiž nesou emocionální informaci, kterou je vhodné zpracovat. Tímto se snažím naznačit, že je potřeba určit správnou úroveň normalizace a odstraňovat pouze to, co není užitečné. Na [15] se dá zjistit, kolik normalizace je potřeba v jaké situaci. Čím více je trénovacích dat, tím méně normalizace je potřeba, protože model je schopný se z nich naučit všechny potřebné detaily (že např. slova film a Film jsou vlastně totožná). Dále se také musí myslet na to, jak šumivý daný text je, popřípadě jaký je jeho obsah. Čím šumivější text je, tím je normalizace více potřeba. Pokud obsah textu není dostatečně obecný (všechna data si jsou dosti podobná) je lepší použít více normalizace.

Nejjednodušším, ale přesto efektivním způsobem normalizace textu pro analýzu sentimentu je **převedení všech písmen na malá**. Tímto se předejde problému v předchozím příkladě s malým nebo velkým písmenem na začátku slova („film“ vs. „Film“) a odstraní se redundantnost ve slovníku. Pokud je tento způsob normalizace použit při tréningu, musí se poté použít i při samotné analýze dat. Analyzátor by všechna slova s velkým písmenem neměl ve slovníku (neznal by je) a snížila by se přesnost. Při analýze sentimentu velikost písmen nehraje roli. Spíše při ní jde o celkové vyznění dané recenze, proto je vhodné tento způsob normalizace využít.

Po již popsaném odstranění nechtěných znaků, převedení slov do gramaticky správné formy (například „supeeer“ na „super“) a jejich úpravy na pouze malá písmena, můžeme text upravovat dále. Mezi takové úpravy patří odstranění stop slov, lemmatizace a stematizace.

Odstranění stop slov je proces při kterém se odstraňují všechna slova s velice malou informační hodnotou pro analýzu. Opět záleží na cíli analýzy, ale obvykle tato slova jsou zájmena, předložky a spojky. Často se odstraňují i čísla, většinou nenesou velkou informační hodnotu. Myšlenkou je odstranit všechny nedůležitá slova a soustředit se na ta důležitá, tato metoda však při použití hlubokého učení přesnost příliš nezvýší.

Stematizace a **Lemmatizace** jsou velice podobné metody. Obě se snaží najít kořenový tvar všech slov v textu. Cílem je opět snížit počet slov ve slovníku, protože se z několika slov stane jedno (například z „natáčejí a „natočil“ se stane „točit““. Lemmatizace se snaží nalézt opravdový spisovný kořen slova. Oproti tomu stematizace používá heuristik pro vytvoření nějakého kořenu, který často ani není opravdovým slovem.

2.5.2 Tokenizace

Jak už název napovídá, cílem tohoto kroku je rozdělit analyzovaný text na tokeny. Tokenem může být jakákoliv pro analýzu významná část textu (slovo, věta, odstavec) [10]. Podle způsobu tokenizace jsou při procesu odstraňovány oddělující znaky (mezery, tečky, čárky a podobně). Podle [15] je tokenizace velice komplexní a důležitý krok předzpracování textu, přesto mu není věnována dostatečná pozornost. Přestože se tento krok zdá triviální, jeho správné provedení je důležité pro zvýšení přesnosti a odstranění chyb v systému.

Určení správného způsobu rozdělení textu je závislé na cíli tohoto kroku a jazyku (popřípadě typu textu) na kterém je tokenizace prováděna.

Pro analýzu sentimentu se obvykle rozděluje text na věty nebo slova. Při tokenizaci textu na slova máme několik možností. První z nich jsou **naivní tokenizační algoritmy**, které slepě rozdělují text podle mezer. Nastává problém s interpunkcí ve větě. Interpunkční

znaménka se „přilepí“ k nějakému slovu, tím vznikne slovo, které analyzátor nezná a vznikne přesně ten problém, který je předzpracováním snaha odstranit.

O něco lepším přístupem je oddělovat slova podle mezer a interpunkčních znamének. Tímto se odstraní předchozí problém, ale vzniknou další. Algoritmus selhává například se zkratkami („U.S.A“) nebo u slov obsahující uvozovky a apostrofy. Dalším problémem jsou různé speciální znaky, které by mohly být pro určité typy analýzy potřebné (např. emotikony a odkazy).

Kvůli těmto a dalším problémům je nutné využít komplexnější **tokenizátory založené na pravidlech**. Možnost mít pro každý případ pravidlo, jak se má s textem nakládat pomáhá řešit všechny předem zmíněné problémy. Tato metoda se dá implementovat například regulárními výrazy, ale není to příliš doporučováno. Při komplexnějších pravidlech se začnou objevovat chyby, které se těžko opravují. Lepším přístupem je využít již existující implementace jako jsou například Spacy, Moses nebo NLTK tokenizátor. Implementace mají již předem vytvořená pravidla pro nejčastější případy v různých jazycích. Nejprve vše rozdělí klasicky podle mezer a poté projdou jednotlivé tokeny a pročistí je. Oddělí se interpunkce a podobně. Uživatel těchto knihoven si samozřejmě může specifikovat svá vlastní pravidla, kdy tokenizovat a kdy ne. Implementované knihovny jsou schopny provádět jak tokenizaci na slova tak na věty.

Speciálním případem tokenizátoru, který se snaží ještě dále zlepšit přesnost systému je **„subword tokenizer“**. Název napovídá jeho funkci. Metoda má mnoho konkrétních implementací, každá využívá trochu jiný algoritmus. Detaily o každém z nich jsou na [15]. Všechny algoritmy sdílí hlavní myšlenku toho, že všechny často se objevující slova by měly mít svá vlastní id (označení po převedení do číselné reprezentace). Méně častá slova se skládají z jejich částí. Tento způsob pokrývá velké množství slov za využití co nejmenšího počtu id, tedy slov, které si tokenizátor musí uložit do slovníku. Tento tokenizátor se musí natrénovat na datech, aby se naučil nejčastější slova a části těch méně častějších. Je tedy potřeba mít trénovací data. Po natrénování tokenizátoru jsou využita naučená slova, popřípadě části slov k rozdělení analyzovaného textu.

2.5.3 Převedení textu do číselné podoby

Jak již bylo řečeno dříve, analyzátory založené na hlubokém učení nepracují přímo s textem, ale s čísly. Z tohoto důvodu se vstupní text (ať už při tréningu nebo při samotné analýze) musí převést na číselnou reprezentaci.

Nejjednodušším způsobem, jak převést slova na čísla by samozřejmě bylo vzít seznam všech nalezených slov a každému z nich přiřadit nějaké číslo [16]. Takže z věty „Film se mi líbil.“ by byl vytvořen slovník, kde slovo „Film“ by bylo reprezentováno číslem 1, slovo „se“ číslem 2 a podobně.

Tenhle způsob by v určitých situacích byl použitelný, ale musí se počítat s tím, že analyzátor s čísly provádí výpočty. Slovo reprezentované vyšší hodnotu by se chovalo, jako by mělo větší váhu, takové chování není vždy chtěné.

Řešením je převést danou číselnou hodnotu na kód 1 z N. Tento kód reprezentuje každé slovo (jeho číselnou hodnotu) pomocí vektoru. Vektor má velikost takovou, jak dlouhý je celý slovník (pokud je ve slovníku 10 000 slov, vektor bude mít 10 000 prvků). Ve vektoru se na indexu, který se rovná číselné hodnotě slova přičte jednička, zbytek budou nuly. Pro předchozí příklad by tedy slovo „Film“ mělo vektor $[1, 0, 0, 0]$, slovo „se“ by mělo vektor $[0, 1, 0, 0]$ a tak dále. Využitím této metody je možné reprezentovat jakkoliv dlouhý text. Této metodě se anglicky říká *bag of words*. Je schopná reprezentovat slova v textu

i jejich počet, ale není schopná zachytit jejich kontext, tedy jaká slova se vyskytují kde. Reprezentaci je možné vylepšit využitím n-gramů nebo tf-idf [26].

Kapitola 3

Návrh systému

Cílem této kapitoly je podívat se na vytvářený systém z abstraktnějšího pohledu a prozkoumat jednotlivé akce, které bude systém provádět. Akcí bude několik, proto systém bude rozdělen do logických celků, které obstarávají určitou část procesu analýzy. Na každou z těchto částí jsou kladeny určité požadavky, které musí být splněny.

Systém je navrhován postupně, podle jednotlivých kroků analýzy. Prvním krokem je obstarání dat, druhým je jejich uložení, dále pak jejich zpracování a analýza. Informace získané analýzou poté budou prezentovány.

3.1 Získání dat

Jak již bylo popsáno v teoretickém rozboru (kapitola 2.3), data jsou pro tento systém velice důležitá. Dále bylo rozebráno, že budou potřeba data pro trénování i samotnou analýzu. Trénovací datová sada by měla ideálně obsahovat anotace hledaných odpovědí klasifikátoru. Recenzí bude potřeba co nejvíce a při jejich větším počtu nepřipadá manuální anotace (označování) v úvahu. Z tohoto důvodu by bylo ideální najít zdroj recenzí, kde jsou jednotlivé recenze již anotovány. Možností jejich získání je hned několik, záleží na konkrétním zvoleném zdroji. Nejpohodlnějším způsobem získávání recenzí by bylo za využití *API* dané webové stránky, další možností je využít jednu z popsaných technik extrakce dat v kapitole 2.3.

Možných zdrojů dat je velké množství, lidé dnes využívají pro sdíleních svých názorů sociální média, blogy, fóra, specializované stránky pro recenze a další.

Získávání dat ze sociálních médií je pravděpodobně nejjednodušší, protože obvykle nabízí *API*. Na druhou stranu data na sociálních médiích neobsahují požadované anotace, poměrně často bývají irelevantní a kvůli zvyšujícímu se soukromí na těchto platformách se informace stávají nepřístupnými.

Dále by se daly využít již vytvořené datové sady. Komunita věnující se zpracování přirozeného jazyka je velice aktivní po celém světě, takže se dá nalézt již anotovaná datová sada téměř v jakémkoli jazyku. Tento přístup by se dal využít k počátečnímu natrénování klasifikátorů, systém by však měl být schopný automaticky a pravidelně stahovat nové recenze, aby zůstal aktuální.

Nejlepším zdrojem dat tedy nejspíš bude některá z webových stránek specializujících se na filmové recenze. Tyto stránky splňují všechna dříve zmíněná kritéria. Navíc velké platformy obvykle mají i vlastní *API*, které by zjednodušilo získávání recenzí.

3.1.1 Extrahované informace

Nezávisle na zdroji recenzí, ze získávaných příspěvků bude pro úspěšnou analýzu nutnost získávat určité informace. Nejdůležitější je samozřejmě samotný text recenze, dále je však potřeba číselné hodnocení autora pro následné porovnání hodnot při trénovaní klasifikátorů. Další důležitou informací je cíl dané recenze (tedy film, kterého se recenze týká). Uživatelské jméno autora se dá využít k případné identifikaci této recenze, popřípadě k přiřazení váhy k recenzi na základě toho, kdo danou recenzi napsal. Datum vytvoření recenze bude užitečné pro sledování postupné změny názorů v čase. Poslední získávanou informací by měla být webová stránka, ze které byla recenze stažena. Tato informace je důležitá pro následné nalezení recenze a pro identifikaci jazyka recenze (systém by měl pracovat pro český i anglický jazyk).

3.1.2 Ukládání dat

Zmíněno v teorii (kapitola 2.4), k uchování získaných dat je více možností. Tento systém by však nejlépe fungoval s využitím relačního databázového systému. Důvodem je strukturovanost extrahovaných dat (bude stačit ukládání do tabulek, není tedy potřeba NoSQL databáze). Na druhou stranu ukládání do souborů (v serializačním formátu) by nebylo velice moudré, protože je předpokládané velké množství recenzí a k uloženým datům bude pravděpodobně přistupovat více částí systému najednou. Velké soubory se pomalu zpracovávají a není možné s nimi pracovat z několika míst najednou.

3.2 Analýza dat

Po získání a uložení dat v systému navazuje jejich analýza. Data by měla být ukládána ve své původní formě, tedy neupravené, pro zachování co nejvíce informací. Před samotnou analýzou by se však měla předzpracovat využitím technik popsaných v kapitole 2.5. Předzpracování pomáhá systému zvýšit jeho přesnost.

Nejvyšších přesností dosahují metody klasifikace založené na strojovém učení, z tohoto důvodu bude tato část systému rozdělena na část pro trénování klasifikátoru a část využívající natrénovaný klasifikátor.

Recenze bych analyzoval na úrovni dokumentu a úrovni aspektu. Na úrovni dokumentu by byla prováděna polaritní analýza (pozitivní / negativní polarita), dále pak analýza míry této pozitivity (jedna až pět, nebo jedna až deset hvězd). Dále by mohla být prováděna analýza na úrovni aspektu pro získání detailnějších informací z recenze. Ke zjištění jaké aspekty budou analyzovány bude potřeba prozkoumat uživatelské recenze a zjistit o čem uživatelé nejčastěji píší. Pro aspektovou analýzu bude oproti ostatním nutné ručně anotovat recenze, nalézt již vytvořenou datovou sadu se mi asi nepodaří.

Analýzu je možné provádět množstvím metod, chtěl bych jich tedy vyzkoušet několik a porovnat jejich výsledky.

Výsledky analýzy by se měli opět ukládat do relační databáze pro jejich následné zpracování a zobrazení.

3.3 Zpracování a vizualizace výsledků

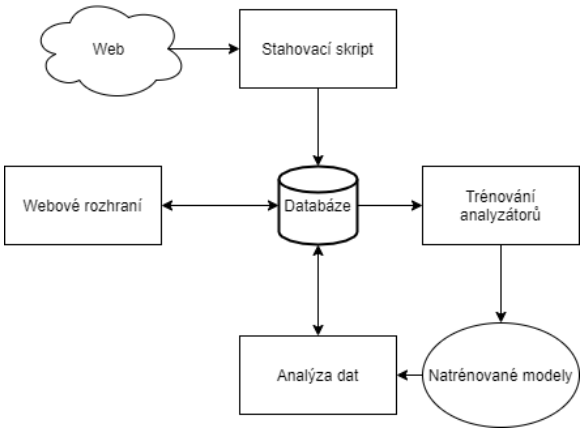
Pouhý soupis výsledků jednotlivých analýz by toho moc neprozradil. Je tedy vhodné výsledky agregovat a poté je zobrazit například grafem. K tomuto bych využil webové rozhraní

z důvodu pohodlí uživatele. Nemusí nic stahovat ani instalovat, může se na stránky podívat téměř na jakémkoliv zařízení a uživatelé by si například mohli vytvořit vlastní účty.

Jednotlivé recenze a jejich výsledky bude samozřejmě nutné seskupit podle filmu, kterého se týkají. Uživatel poté může vyhledávat daný film, nebo procházet žebříček těchto filmů podle výsledných hodnot analýz. Dále by mohly být sledovány různé trendy titulů (popularita žánrů, jestli jsou populárnější filmy či seriály a podobně).

3.4 Výsledný systém

Jednotlivé části systému by spolu měly komunikovat dle obrázku 3.1.



Obrázek 3.1: Architektura navrhovaného systému. Jednotlivé části budou ovládány nástrojem pro plánování úloh.

Kapitola 4

Implementace systému

V této kapitole popisuji implementaci dříve definovaného systému. Systém je rozdělen do následujících částí:

- Stahování potřebných dat.
- Databáze.
- Trénování analyzátorů.
- Analýza sentimentu.
- Rozhraní pro zobrazení výsledků analýzy.

Pro každou z těchto částí je věnována samotná podkapitola. Při popisu každé z částí jsou nejprve prozkoumány možnosti řešení, které byly brány při v potaz a následuje popis výsledné implementace.

Převážná většina systému je vytvořena v jazyku Python verze 3.6.9. Je využito několik pomocných skriptů pro bash. Výsledky analýzy jsou uživateli zobrazovány v prohlížeči za použití webového aplikačního rámce Django. Systém běží na školním serveru *Athena1* a je implementován pro češtinu i angličtinu.

Protože je na systém požadavek pravidelné aktualizace dat, skripty pro stahování, zpracování dat a sémantickou analýzu je možné spouštět manuálně i automaticky. Postupně se tedy zvětšuje počet uložených recenzí v databázi. Tyto recenze jsou poté (opět automaticky) analyzovány natrénovanými klasifikátory. Natrénování klasifikátorů se dělá manuálně.

Automatické spouštění všech skriptů je prováděno pomocí plánovače úloh *Cron*. Úlohy plánovače (takzvané *Cronjobs*) jsou zapisovány do souboru zvaném *Crontab* a mají následující formát:

minuta[0-59] hodina[0-23] den měsíce[1-31] měsíc[1-12] den týdne[0-6] příkaz ke spuštění

Namísto konkrétních čísel je možné využít znak hvězdičky, který zastupuje každou instanci (každou minutu, každou hodinu a podobně). Z toho vyplývá že *Cron* umožňuje spouštění jednou v konkrétní čas, ale i pravidelně.

Cron nespouští přímo požadovaný python skript, ale jeden z dříve zmiňovaných bash skriptů. Tento bash skript nejprve zapne vývojové *Conda* prostředí obsahující všechny systémem používané knihovny a potom až daný python skript.

Většina skriptů tvořících tento systém nějakým způsobem pracuje s daty, potřebují tedy přístup k nějakému úložišti dat.

Pro ukládání dat jsou podle potřeby využity:

- Lokální soubory, formátu JSON nebo prostého textu.
- PostgreSQL databáze.
- SQLite databáze.

4.1 Stahování dat

Původně měli být zdroje dat Facebook a Twitter. To se ovšem ukázalo jako ne tím nejlepším řešením, protože data ze sociálních sítí nebývají příliš kvalitní. Obě platformy mají navíc limitovaný přístup k datům a chybí zde číselné ohodnocení od autora.

Rozhodl jsem se tedy využít weby pro recenze filmů, kde jsou příspěvky relevantnější (uživatelé více vyjadřují svůj názor) a jsou dostupná číselná hodnocení od uživatelů. Díky číselnému hodnocení nemusí být texty recenzí manuálně anotována a hodnocení může být použito jako referenční výsledek pro trénování analyzátorů. Toto řešení není úplně ideální, protože uživatelé občas dají pozitivní hodnocení (celkově se jim film líbil) a do textu recenze napíší všechny chyby filmu (určité části nebo detaily, které se jim nelíbily). Tohle by mohlo analyzátor při trénování zmást. Ideální by asi bylo data překontrolovat a hodnocení podle obsahu textu změnit. Dat je však tolik, že tohle není možné provést.

Zdroji dat se tedy staly weby *csfd.cz*, *fdb.cz*, *rottentomatoes.com* a *imdb.com*. Pro získání dat z webových portálů jsem se pokusil získat přístup k jejich API, žádný z webů mi však nevyhověl (portál rottentomatoes.com má pro tento účel dotazník). Ke stažení recenzí z daných webových stránek tedy byly vytvořeny skripty využívající *web scraping*. Každý portál má svůj vlastní skript.

Všechny skripty pro stahování recenzí (kromě toho pro čsfd) mají společnou třídu *Database_manager* v souboru *database_manager.py*. Tato třída má na starost práci s PostgreSQL databází. Databáze běží na školním serveru *Athena1*. Třída zajišťuje připojení a přihlášení k databázi a implementuje několik funkcí pro dotazy a ukládání dat.

Stahování dat se spouští třídou *Data_downloader* v souboru *data_downloader.py*. Tato třída pouze podle nastavení v souboru *config.py* spouští jeden stahovací skript za druhým.

Detailně jsou jednotlivé způsoby stahování dat popsány v následujících podkapitolách.

4.1.1 Stahování z ČSFD

Stahování z této webové stránky je implementováno třídou *Web_scraper_csfd* v souboru *web_scraper_csfd.py*. Webové stránky *csfd.cz* jsou celé implementovány staticky (nevyužívají JavaScript pro doplňování obsahu stránky). Z tohoto důvodu je stahování prováděno za využití knihovny *urllib*, která umožňuje zasílat HTTP požadavky. Tato knihovna je implicitně součástí Python distribuce. Po získání HTTP odpovědi serveru jsou data zpracována pomocí knihovny *BeautifulSoup* verze 4.8.2. Knihovna umožňuje procházet stromovou strukturou HTML dokumentu. Pro extrakci hledaných dat je nutné si nejprve projít strukturu stránky. Nejpohodlnější je využít funkci „Prozkoumat“, kterou má každý webový prohlížeč a zjistit kde se ve stromové struktuře data nachází. Podle toho pak stačí využít vyhledávací funkce knihovny *BeautifulSoup*. Možné je vyhledávat například podle názvu HTML značky, jejího id nebo třídy (class). Možností je samozřejmě více, pro detail má knihovna dokumentaci[1].

[1] `https://www.crummy.com/software/BeautifulSoup/bs4/doc/`

K vytvoření datové sady pro trénování analyzátorů i následnou analýzu jsou ke každé recenzi získávány následující informace:

- Název titulu, ke kterému se recenze vztahuje.
- Text recenze.
- Hodnocení uživatele, na čsfd může uživatel hodnotit nula (odpad!) až pěti hvězdičkami, hodnota je před uložením normalizována na hodnotu mezi nulou a jedničkou.
- Identifikační číslo uživatele.
- Datum zveřejnění recenze.

Stahování nebylo možné provádět přímo na školním serveru *Athena1*, protože po odeslání požadavku ze školního serveru na webový server se vždy vrátilo chybové hlášení. Z tohoto důvodu byl skript spouštěn na mém osobním počítači.

K databázi na školním serveru se z venkovní sítě nedá připojit. Z tohoto důvodu byla na osobním počítači použita databáze SQLite. Databáze byla použita pro zaznamenávání již stažených příspěvků, aby skript nestáhl jeden příspěvek několikrát. Tabulka těchto záznamů má dva sloupce. Jeden pro název titulu a jeden pro identifikační číslo autora recenze. Předpokladem je, že každý uživatel napíše maximálně jednu recenzi ke každému titulu.

Princip algoritmu pro stahování dat je následující. Nejprve se skript připojí k SQLite databázi pomocí třídy *Database_manager_csfd*. Poté je zavolána funkce *download_all*, která započne stahování. Algoritmus využívá postupně vzrůstajícího identifikačního čísla uživatelů. Identifikační číslo uživatele od kterého má skript začít stahovat je nastavováno manuálně pro lepší kontrolu nad skriptem (alternativně by se mohlo číslo nastavit podle posledního záznamu v databázi). Postupně se tedy přičítá číslo aktuálního uživatele a prochází se odkazy následujícího formátu:

https://www.csfd.cz/uzivatel/<identifikační číslo uživatele>

Přijetím odpovědi serveru se zjistí, jestli účet uživatele s daným identifikačním číslem existuje. Pokud byl účet smazán, vrátí se chybová odpověď. Algoritmus si zaznamenává počet po sobě jdoucích chybových odpovědí. Pokud tato hodnota překročí určitou mez, stahování končí, protože algoritmus došel na konec seznamu identifikačních čísel uživatelů (identifikační čísla ještě nikdo nepoužívá).

Pokud se od serveru vrátí pozitivní odpověď, přejde se do komentářové sekce uživatele. Postupně se prochází všechny komentáře daného uživatele a stahují se dříve zmiňované informace. Identifikační číslo uživatele a název komentovaného titulu se porovná se záznamy v SQLite databázi. Pokud takový záznam ještě neexistuje, tak se v databázi vytvoří a informace se uloží. V opačném případě se informace zahodí. Stažená data jsou ukládána do lokálního souboru ve formátu JSON a později vložena do databáze na školním serveru. K uložení dat do školní databáze se používá pomocný skript *upload_csfd.py*. Na školním serveru jsou recenze ukládány do tabulky *reviews*.

4.1.2 Stahování z FDb

Implementováno třídou *Web_scraper_fdb* v souboru *Web_scraper_fdb.py*. Stejně jako webové stránky čsfd jsou i tyto implementovány plně staticky, využívají se tedy stejné nástroje (*urllib* a *BeaurifulSoup*). Stránky samozřejmě mají jinou strukturu než čsfd, proto

je nutné je také projít (využitím nástroje *Prozkoumat*) a vyhledat všechny důležité elementy HTML dokumentu.

Seznam získávaných informací se oproti tomu z čsfd liší pouze v použití uživatelských jmen namísto identifikačních čísel.

Na rozdíl od skriptu pro čsfd, tento byl spouštěn na školním serveru. Byla tedy využita školní PostgreSQL databáze. K připojení a provádění akcí s databází je využita již zmiňovaná třída *Database_manager*. Pro záznamy již stažených recenzí (aby se jedna recenze nestáhla několikrát) je využita tabulka s názvem *users_fdb*, která funguje stejným způsobem jako SQLite databáze pro čsfd. Výsledné informace o recenzích jsou přímo ukládány do tabulky *reviews*.

Algoritmus funguje velice podobným způsobem jako při stahování z čsfd. Hlavním rozdílem je způsob procházení stránky (důvodem je samozřejmě jiná struktura stránek).

Dalším rozdílem je postupné procházení uživatelů, které v tomto případě nevyužívá identifikačního čísla. Uživatelé i zde identifikační čísla mají a odkazy na uživatelské účty by se daly procházet stejným způsobem jako u čsfd. Stránky však při neexistenci uživatelského účtu s daným identifikačním číslem neodpoví chybovým hlášením. Namísto toho přesměrovávají na hlavní stránku (fdb.cz). Z tohoto důvodu jsou uživatelé nejprve vyhledáni v žebříčku uživatelů a až poté se stahují jejich příspěvky.

4.1.3 Stahování z Rotten Tomatoes

Provádí třída *Web_scraper_rottentomatoes* v souboru *Web_scraper_rottentomatoes.py*.

Stránky jsou pouze částečně statické. Stahování dynamických stránek je mnohem pomalejší, protože se musí využít prohlížeče pro jejich vykreslení. Z tohoto důvodu jsem se rozhodl stahovat pouze statickou část stránek. Využité knihovny a základní postup procházení stránek je tedy pořád stejný. Stahování dynamických stránek jsem využil až pro stránky imdb, kde je u každého titulu mnohem více recenzí a tak se čekaní na načtení stránek více vyplatí. Například pro film Joker je na rotten tomatoes 137 000 hodnoceni, na imdb je to 821 000. Do těchto počtů jsou započítána i hodnocení bez napsání textu recenze, ale dá se předpokládat že na obou platformách komentuje stejné procento lidí. Stránky rotten tomatoes nezobrazují počet napsaných recenzí, proto bylo porovnání provedeno takto.

Statická část rotten tomatoes obsahuje top 100 filmů pro každý ze 17ti žánrů. Na této platformě jsou uživatelé rozdělení do dvou kategorií. Jedna kategorie jsou *Audience* (tedy obecenstvo), což jsou obyčejní uživatelé této platformy. Druhou kategorií jsou *critics* (tedy kritici), toto jsou uživatelé schválení platformou jako důvěryhodní. Stahovaná (statická) část obsahuje pouze kritiky, část s obecenstvem je implementována dynamicky. Počet recenzí se tedy dále zmenší, protože kritiků není tolik co obecenstva. Současně je v databázi 130 000 recenzí z rottentomatoes, což je téměř stejně jako z fdb.

Při stahování jsou ke každé recenzi získávány tyto informace:

- Název titulu, ke kterému se recenze vztahuje.
- Text recenze.
- Uživatelské jméno
- Hodnocení uživatele, na rozdíl od ostatních stránek, zde uživatel nevyužívá předem danou stupnici, ale může hodnocení psát ručně.
- „Čerstvost“, uživatelé ještě navíc mohou titulu přiřadit „čerstvé“ nebo „shnilé“. Hodnoty jsou stahovány, ale k ničemu nebyly využity.

- Datum zveřejnění recenze.

V PostgreSQL databázi jsou skriptem využívány dvě tabulky.

Tabulka *users_rottentomatoes* je pro záznamy stažených recenzí, stažená data se opět ukládají do *reviews*.

Algoritmus oproti ostatním prochází namísto uživatelů seznam filmů v žánrech. Jinak je princip téměř stejný. Jediný větší rozdíl potřebný zmínit je zpracování číselného hodnocení uživatelů. Jak už je napsáno dříve, uživatelé mohou číselné hodnocení psát ručně. Prozkoumáním několika recenzí lze zjistit, že uživatelé číselné hodnocení většinou píší v následujících dvou tvarech:

- <číslo>/<číslo>
- Hodnocení od A po F, někdy také s + či −

Pokud algoritmus rozpozná, že hodnocení je psáno prvním způsobem, čísla se vydělí a tím je hodnocení normalizováno mezi nulu a jedničku. V druhém případě je hodnocení namapováno na hodnotu mezi nulu a jedničku.

4.1.4 Stahování recenzí z imdb

Implementováno třídou *Web_scraper_imdb* v souboru *Web_scraper_imdb.py*. Stránky mají všechny důležité části implementované dynamicky. Pro projití stránek a stažení všech potřebných dat se tedy musí využít jedna z technik pro dynamické stránky z kapitoly 2.3.

Pro zpracování dynamicky generovaného obsahu stránky jsem se rozhodl využít *Selenium*. Selenium nabízí množství nástrojů původně vyvinutých pro automatické testování webových aplikací, dá se však využít i pro procházení webových stránek, vyhledání a případnou extrakci dat. Balíček se skládá z několika komponent, k mému účelu však využívám jen *Selenium WebDriver*. Nástroj pro svou práci využívá webový prohlížeč, který je schopný ovládat. Nabízí funkce jako vyhledávání HTML elementů, simulaci psaní klávesnicí nebo simulaci klikání[2].

Podporovány jsou následující webové prohlížeče:

- Google Chrome
- Internet Explorer
- Safari
- Opera
- Firefox

Nástroj je nejprve potřeba nainstalovat (v mém případě nainstalováno ve vývojovém *Conda* prostředí) a poté jej importovat ve skriptu který jej bude používat. Po výběru jednoho z podporovaných prohlížečů je dále nutné stáhnout webový ovladač daného prohlížeče.

Data získávaná o každé recenzi jsou stejná jako u fdb. Stažená data jsou opět ukládaná do školní PostgreSQL databáze, přesněji do tabulky *reviews*. Pro záznamy již stažených recenzí je využita tabulka *users_imdb*.

Při inicializaci objektu vytvořeném z třídy *Web_scraper_imdb* se nejprve vytvoří nastavení pro prohlížeč a nastaví se „headless“ na „True“. To má za následek to, že se prohlížeč

[2] `https://www.selenium.dev/`

nespouští v okně a běží na pozadí (tedy není vidět). Vzhledem k tomu, že většinou tento skript poběží automatizovaně by otevřené okno nedávalo smysl. Otevřené okno se dá využít pro řešení případných chyb, protože se dá sledovat všechno co se v prohlížeči děje. Dále se při inicializaci vytvoří profil pro prohlížeč *Firefox* a nastaví se preference pro angličtinu. Jinak by se jako výchozí nastavení využila čeština, to by udělalo nepořádek v datech, protože se stahují anglická data. Stránka imdb totiž některá slova automaticky překládá do nastaveného jazyka.

Po vytvoření všech nastavení se načte soubor s ovladačem pro prohlížeč *Firefox* a daná nastavení se mu předají. Na konci inicializace se ještě vytvoří objekt pro připojení k databázi (za pomocí již několikrát zmiňované třídy *Database_manager*).

Jakmile se dokončí inicializace, zavolá se funkce *download*, ve které začíná stahovací algoritmus. Skript se připojí k databázi a otevře prohlížeč na stránce:

https://www.imdb.com/feature/genre/?ref_=nv_ch_gr

Na této stránce je seznam žánrů filmů a seriálů. Skript si uloží odkazy na všechny žánry a začne je procházet. Každý žánr obsahuje obrovské množství titulů seřazených podle popularity.

Algoritmus opět funguje velice podobně předchozím. Tituly se stránku po stránce prochází a stahují se recenze uživatelů. Jednou z komplikací, kterou použití nástroje *Selenium* přináší je nutnost synchronizace stahovacího skriptu s prohlížečem. Před tím než skript může začít extrahovat hledaná data ze stránky musí počkat, než se dokončí dříve zadaná instrukce (nejčastěji se čeká než se načte stránka). K tomu se dají využít funkce obstarávající čekání.

4.1.5 Stahování dodatečných informací z imdb

Implementováno třídou *Add_info_imdb* v souboru *add_info_imdb.py*. Cílem tohoto skriptu je získat ke každému filmu dodatečné informace, které nebyly staženy při stahování recenzí. Tyto informace jsou důležité pro analýzu trendů.

Tento skript byl ke stahovacím skriptům přidán až jako poslední. V době implementace ostatních stahovacích skriptů ještě nebylo patrné jaké trendy bude systém sledovat. Proto jsou data stahována takto dodatečně. K vyhledávání a extrakci dat je i zde využitý *Selenium WebDriver* využívající webový prohlížeč *Firefox*.

K titulům jsou doplňovány následující informace:

- Typ titulu (film nebo seriál).
- Originální celkové hodnocení (jedna až deset hvězd), normalizováno mezi nulu a jedničku.
- Seznam žánrů titulu.
- Seznam zemí podílejících se na tvorbě titulu.

Skript pro přístup do školní PostgreSQL databáze má vlastní třídu *Add_info_database*, která je přímo v souboru *add_info_imdb.py*. Tento skript provádí nad databází úplně jiné dotazy než ostatní stahovací skripty, proto byla vytvořena jiná třída. Typ titulu a originální hodnocení jsou ukládány do tabulky *movies*. Seznam žánrů má svou vlastní tabulku *genres*. Seznam zemí má *countries*.

Inicializací objektu vytvořeného z třídy *Add_info_imdb* se nastaví proměnné dále používané algoritmem. Webový prohlížeč je nastaven stejným způsobem jako při stahování recenzí z imdb. Je vytvořen objekt pro připojení k databázi využitím již zmiňované třídy a kompiluje se regulární výraz použitý dále v algoritmu.

Následně je zavolána funkce *download*, kde se skript připojí k databázi, vytvoří se databázový dotaz na názvy titulů z tabulky *movies* a otevře se prohlížeč na hlavní stránce *imdb.com*. Algoritmus poté pomocí dříve vytvořeného databázového dotazu a posouvajícího se databázového kurzoru postupně získává názvy titulů.

Názvy titulů někdy obsahují závorku s doplňujícími informacemi. Název titulu bez závorky s doplňujícími informacemi je prostě vložen do vyhledávacího okna a vybírá se první shodná nabídka.

Název titulu obsahující závorku je pomocí regulárních výrazů rozdělen na část před závorkou a informace v závorce. Pokud by název titulu nebyl takto rozdělen, imdb vyhledávač by titul vůbec nenašel. Po rozdělení je část názvu před závorkou vložena do vyhledávacího okna. Imdb vyhledávač obvykle poskytne více možností. Správná možnost je zvolena právě podle doplňujících informací ze závorky. Pokud se žádná správná možnost nenajde, klikne se na tlačítko *More title matches* a dojde k dalšímu pokusu o vyhledání správné možnosti. Když se ani nyní nenajde správná možnost, vyhledávání pokračuje dalším titulem.

Po nalezení správného výsledku vyhledávání se z detailů o titulu stáhnou všechny dříve zmiňované informace.

4.2 Databáze

Je typu *PostgreSQL*, vytvořena na školním serveru *Athena1*.

Tato sekce slouží jako přehled všech používaných databázových tabulek pro lepší orientaci. U každé tabulky je vysvětleno k čemu slouží a jaké hodnoty jsou ukládány do jednotlivých sloupců. Většinou jsem pro práci s databází používal *phpPgAdmin*, což je webové rozhraní pro zjednodušení ovládání.

4.2.1 aspects

Tabulka slouží k ukládání výsledků aspektové analýzy.

- review_id - cizí klíč do tabulky reviews, udává recenzi, které se výsledek analýzy týká.
- stat_used - boolovská hodnota, říká jestli výsledek analýzy byl již agregován do celkového výsledku v tabulce movies.
- <aspekt>_pos a <aspekt>_neg - je pět aspektů, takže celkově deset sloupců. Obsahují počty pozitivních / negativních výskytů aspektů. Za každý pozitivní / negativní výskyt daného aspektu v recenzi je přičtena jednička.

4.2.2 classes_5

Tabulka slouží k ukládání výsledků analýzy polarity s mírou intenzity (jedna až pět hvězd).

- review_id - cizí klíč do tabulky reviews, udává recenzi, které se výsledek analýzy týká.
- <třída>_star - Intenzita je dělena do pěti tříd, je tedy pět sloupců. Jedná se o výstupy analyzátoru. Každý sloupec představuje předpověď analyzátoru pro danou třídu. Hodnota je indikována desetinným číslem.

- final_rating - Výsledná předpověď analyzátoru, je vybrána třída s nejvyšší hodnotou předpovědi. Indikováno názvem třídy.
- stat_used - boolovská hodnota, říká jestli výsledek analýzy byl již agregován do celkového výsledku v tabulce movies.

4.2.3 countries, genres

Tyto dvě tabulky slouží k uložení dodatečných informací o titulu, využívané při analýze trendů. Název titulu se v tabulkách obvykle opakuje, titul může mít více zemí podílejících se na tvorbě i více žánrů.

- title - název titulu.
- country - země podílející se na titulu.
- title - název titulu.
- genre - žánr titulu.

4.2.4 language_mapping

Pomocná tabulka k namapování zdroje recenzí a jazyka.

- source - zdroj recenzí (čsfd, fdb, imdb, rottentomatoes)
- language - jazyk (cz / en)

4.2.5 movies

Obsahuje všechny názvy titulů pro každý jazyk. Název titulu může být stejný pro češtinu i angličtinu, proto je primárním klíčem kombinace názvu titulu a jazyka titulu. Seznam titulů byl vytvořen podle tabulky reviews.

- title - název titulu.
- avg_polarity - průměrná polarita titulu, vypočítaná z hodnot v tabulce pos_neg, hodnota nula až jedna.
- avg_stars - průměrný počet hvězd titulu, vypočítaný z hodnot v tabulce classes_5. Desetinná hodnota jedna až pět.
- <aspekt>_score - průměrná hodnota jednotlivých aspektů (pět sloupců), vypočítaná z tabulky aspects, hodnota nula až jedna.
- language - jazyk zdroje recenzí titulu (cz / en).
- popularity - popularita daného titulu, vypočítaná jako celkový počet recenzí týkajících se tohoto titulu.
- movie_series - boolovská hodnota, značí jestli je titul film nebo seriál.
- original_rating - originální celkové hodnocení titulu získané skriptem pro stahování dodatečných informací.

4.2.6 pos_neg

Tabulka slouží k ukládání výsledků analýzy polarity (pozitivní / negativní).

- review_id - cizí klíč do tabulky reviews, udává recenzi, které se výsledek analýzy týká.
- predicted_value - desetinná hodnota předpovědi (nula až jedna).
- predicted_class - název předpovězené třídy (pos / neg). Hodnoty predicted_value menší nebo rovno 0,5 jsou zde uloženy jako negativní, nad 0,5 jsou pozitivní.
- stat_used - boolovská hodnota, říká jestli výsledek analýzy byl již agregován do celkového výsledku v tabulce movies.

4.2.7 reviews

Tabulka obsahuje všechny stažené recenze.

- id - globálně unikátní identifikační číslo recenze.
- title - název titulu, kterého se recenze týká.
- text - text recenze.
- rating - číselné hodnocení přiřazené uživatelem, normalizované mezi nulu a jedničku.
- critic - jméno autora recenze.
- date - datum publikace recenze.
- freshness - hodnota „čerstvosti“, pouze u recenzí z rotten tomatoes. Hodnota se nakonec k ničemu nevyužívá.
- source - zdroj odkud byla recenze stažena (čsfd, fdb, imdb, rotten tomatoes)
- analyzed - boolovská hodnota, určuje zda byla recenze analyzována.

4.2.8 users_csfd, users_fdb,users_imdb,users_rottentomateos

Tabulky zaznamenávající stažené recenze. Předpokladem je, že každý uživatel napíše ke každému titulu maximálně jednu recenzi.

- movie - název titulu, kterého se recenze týká.
- critic - jméno autora recenze.

4.3 Analýza textu recenzí

Systém provádí tři typy analýz. Analýzu polarity (pozitivní / negativní), analýzu míry pozitivity (od jedné do pěti hvězd) a aspektovou analýzu. Všechny tři typy analýzy jsou implementovány pro český i anglický jazyk a to za pomocí následujících metod:

- Neuronová (*bidirectional* LSTM) síť, implementováno pomocí knihovny *PyTorch*. Architektura sítě v souboru *rnn_model.py*

- Naivní Bayesův klasifikátor, využita hotová implementace z knihovny *nltk*.
- Metoda podpůrných vektorů (SVM, support vector machine), využita také knihovna *nltk*. Knihovna *nltk* nemá přímo svou implementaci, ale přebírá ji z knihovny *sklearn*.

Všechny tyto metody jsou založené na strojovém učení, musí se tedy nejprve natrénovat modely, které jsou ukládány pro pozdější použití. Pro natrénování jsou potřeba trénovací data, detailní popisy použitých dat jsou v podsekcích jednotlivých analyzátorů. Data byla z databáze (tabulka reviews) vložena do lokálních souborů ve formátu JSON pro urychlení procesu trénování. Trénování je poměrně dlouhý proces, proto trénovací skripty byly spouštěny plánovačem úloh a to obvykle přes noc.

Webové stránky čsfd a fdb jsou navštěvovány i slovenskými uživateli. Část recenzí byla tedy psána slovensky. Pro odstranění slovensky psaných recenzí byl použit nástroj *cld3*.

Analyzátory jsou implementovány v několika souborech. Každému souboru je věnována podsekce.

4.3.1 polarity_analyzer.py

Obsahuje třídu *Polarity_analyzer* zodpovědnou za trénování polaritního analyzátoru implementovaného neuronovou sítí. Vstupem analyzátoru je celý dokument (recenze) a výstupem je hodnota mezi nulou a jedničkou (namapována do pozitivní nebo negativní třídy). Soubor také obsahuje pomocné funkce pro předzpracování trénovacích dat a měření času trénování.

Pro implementaci neuronové sítě je využita knihovna *PyTorch*, která umožňuje tvorbu neuronových sítí a zjednodušuje práci s daty (pro práci s daty jsou využity třídy *Field* a *Iterator*).

Pro trénování tohoto analyzátoru existují dvě datové sady. Jedna pro češtinu a druhá pro angličtinu. Ze všech informací stahovaných k recenzím jsou při trénování využity pouze text recenze a číselné hodnocení uživatele. Číselné hodnocení uživatele je při trénování použito jako správný (hledaný) výstup klasifikátoru, data tedy nemusela být ručně anotována. Číselné hodnocení je v rozsahu od nuly po jedničku, klasifikátor však určuje pouze dvě polarity (pozitivní nebo negativní). Číselná hodnota je tedy namapována do dvou kategorií. Pokud je hodnota menší nebo rovna 0.5 tak je polarita označena jako negativní, v opačném případě je prohlášena za pozitivní.

Obvykle mají polaritní analyzátory ještě neutrální výstup, ten jsem se rozhodl vynechat. Důvodem je fakt, že recenze téměř nikdy nejsou neutrální. Oproti například příspěvkům ze sociálních sítí, recenze téměř vždy nesou nějaký názor. Recenze jsou přeci jen psány za účelem projevení názoru.

V inicializační funkci třídy *Polarity_analyzer* je možnost nastavit jazyk trénovaného klasifikátoru. V závislosti na nastaveném jazyku se mění způsob tokenizace, způsob předzpracování dat a použitá trénovací sada. Podle nastaveného jazyku je natrénovaný model uložen do jiné složky.

Tokenizaci recenzí na slova zajišťují knihovny *spacy* (pro anglické recenze) a *nltk* (pro české recenze).

Předzpracování dat je pouze minimální. Trénovací sady obsahují velké množství příkladů, takže složitější metody předzpracování (lemmatizace, stematizace, odstraňování stop slov, POS tagging a podobně) by trénování zpomalily a na přesnosti by přidaly pouhé jednotky procent. Využívá se tedy pouze převedení všech písmen na malá (u angličtiny i češtiny) a odstranění diakritiky (pouze u češtiny).

Po nastavení jazyka následuje zvolení velikosti slovníku. Slovník nesmí být příliš malý (při nedostatku známých slov bude klasifikátor nepřesný), ale ani příliš velký (klasifikátor by se nenaučil pracovat se situacemi, kdy nějaké slovo nezná, na tuto situace však jistě při následné analýze narazí). Pro vytvoření slovníku je v současnosti využito 25 000 nejčastějších slov. O vytvoření slovníku se stará třída *Field* z knihovny *PyTorch*.

Dále je vybrána grafická karta k provádění výpočtů. Jak bylo zmíněno v teorii, trénování neuronových sítí je výpočetně náročné. Na školním serveru *pcknot1* jsem dostal přístup k výkoné grafické kartě, která trénování urychlila.

Neuronová síť je definována třídou *RNN* v souboru *rnn_model.py*. Síť je oboustranná (anglicky bidirectional) a využívá LSTM (long short term memory).

Popsaný skript trénuje model podle nastaveného počtu epoch. Jedna epocha je jedno projití celou trénovací sadou. Na konci každé epochy se zjistí ztráta (anglicky loss) modelu na modelem neznámých datech. Model s nejnižší ztrátou je uložen k budoucímu použití.

4.3.2 class_analyzer.py

Třída *Class_analyzer* se využívá k natrénování analyzátoru míry pozitivity. Implementace je i zde provedena neuronovou sítí. Vstupem analyzátoru je opět celý dokument (recenze), ovšem výstupem je pět desetinných hodnot. Každá hodnota odpovídá jedné z následujících tříd:

- one_star - třída indikující recenzi s jednou hvězdičkou. Rozsah $[0 - 0.2]$
- two_star - recenze se dvěma hvězdičkami. Rozsah $(0.2 - 0.4]$
- three_star - třemi. Rozsah $(0.4 - 0.6]$
- four_star - čtyřmi. Rozsah $(0.6 - 0.8]$
- five_star - a pěti. Rozsah $(0.8 - 1.0]$

Poté co klasifikátor vytvoří predikci jsou hodnoty porovnány, třída s nejvyšší hodnotou je považována za odpověď.

Tento analyzátor používá stejné dvě datové sady pro trénování jako analyzátor polarity. Rozdílem zde je, že hodnocení uživatelů jsou rozděleny do pěti dříve zmíněných tříd místo dvou. Hodnoty jsou rozdělovány po 0.2 inkrementech dle rozsahů výše.

Implementace tohoto klasifikátoru je velice podobná polaritnímu. Liší se v rozdělení dat do jiných tříd. Neuronová síť má pět výstupních neuronů (polaritní klasifikátor má pouze jeden). A je využita jiná ztrátová funkce. Zde je využita *cross entropy loss*, kdežto polaritní klasifikátor používá *BCE with logits loss*.

4.3.3 aspect_analyzer.py

Obsahuje třídu *Aspect_analyzer*, která trénuje několik binárních klasifikátorů pracujících dohromady za cílem provedení aspektové analýzy. Jednotlivé klasifikátory jsou opět implementovány užitím neuronových sítí. Analýza je prováděna vzhledem k pěti předem definovaných aspektů:

- Herec
- Příběh

- Postava
- Audio-vizuální efekt
- Zkušenost

Každý z těchto aspektů využívá jeden klasifikátor pro zjištění přítomnosti aspektu a jeden pro určení jeho polarity. Analýza je prováděna na úrovni věty. První klasifikátor od každého aspektu zjistí, jestli daná věta obsahuje daný aspekt. Pokud ano, je využit druhý klasifikátor daného aspektu pro zjištění jeho polarity. Analýza funguje za předpokladu, že každá analyzovaná věta obsahuje maximálně jednu polaritu od každého aspektu. Analyzátor by tedy nefungoval správně kdyby v jedné větě byly dva názory na jeden aspekt.

Trénovací sady dat byly anotovány ručně využitím nástroje *brat*[3] (opět jedna sada pro češtinu a jedna pro angličtinu).

Implementace jednotlivých klasifikátorů je v podstatě stejná polaritnímu analyzátoru. K předzpracování dat se zde přidává lematizace kvůli menším trénovacím sadám.

4.3.4 review_analyze.py

Skript využívá tři dříve popsané analyzátory k analýze recenzí. Jsou využity dříve natrénované analyzátory založené na neuronových sítí (mají nejvyšší přesnost). Text recenzí je získán z databáze, provedou se na něm analýzy podle nastavení (žádná až všechny tři) a získané výsledky jsou uloženy zpět do databáze.

Recenze jsou získávány z tabulky *reviews* a jsou předzpracovány a tokenizovány stejně jako při trénování daného analyzátoru. Výsledky se ukládají do tabulek *pos_neg*, *classes_5* a *aspects*.

4.3.5 naive_bayes_svm.py

V souboru jsou implementovány všechny tři dříve zmiňované typy analýzy (trénování i samotná analýza) pomocí naivního Bayesovského klasifikátoru i metodou pomocných vektorů (opět pro oba jazyky). Pro reprezentaci slov je využita metoda *BOW* popsána v kapitole 2.5. Předzpracování i tokenizace zůstávají stejné jako u klasifikátorů založených na neuronové síti. Natrénované modely jsou opět uložené (knihovnou *pickle*), k vytváření výsledků analýzy se však nepoužívají (mají horší přesnost než neuronové sítě).

Třída *Polarity_analyzer* nabízí nastavit jazyk, množství použitých dat pro trénování, metodu klasifikace („svm“ nebo „bayes“) a typ analýzy (polaritní nebo míru intenzity). Po nastavení je možné zavolat funkci *train_analyzer* pro trénování, nebo *analyze* pro využití dříve natrénovaného analyzátoru se stejnými parametry jako jsou ty nastavené k analýze textu.

Třída *Aspect_analyzer* má k nastavení jazyk a použitou metodu („svm“ nebo „bayes“). Poté je možnost zavolat funkci *train_all_analyzers* pro natrénování sítě klasifikátorů nebo *analyze* k provedení analýzy.

4.4 Vizualizace výsledků

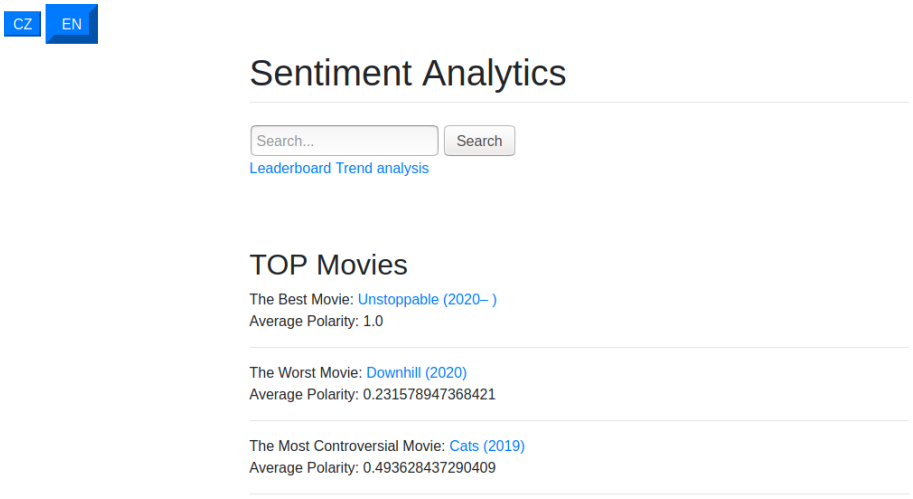
Výsledky jsem se rozhodl uživateli zobrazit ve webovém prohlížeči. Důvodem je především dostupnost, uživatel nemusí nic instalovat a k informacím má přístup kdekoliv kde má internetové připojení.

[3] `https://brat.nlplab.org/`

Pro implementaci webového rozhraní jsem využil aplikační rámec *Django*, který zjednodušuje proces návrhu i implementace webových stránek. Pro vykreslení grafů byl využit rámec *Highcharts*, pro zlepšení celkového vzhledu stránek *Bootstrap* a mnou definované kaskádové styly.

Data jsou do webových stránek dotazována z dříve popsané databáze.

4.4.1 Domovská stránka



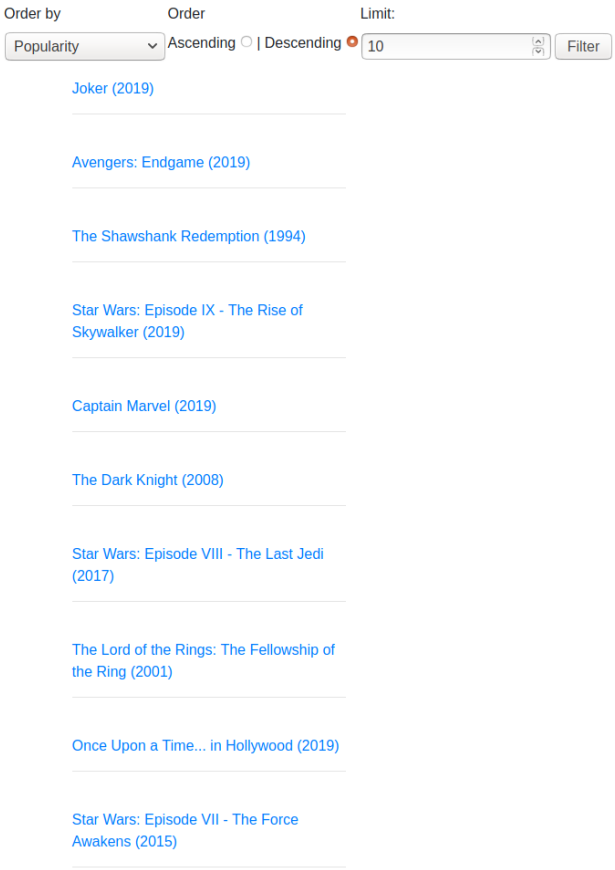
Obrázek 4.1: Domovská stránka webového rozhraní.

Obrázek 4.1 ukazuje vzhled domovské stránky. V levém horním rohu jsou tlačítka pro přepnutí jazyka stránek. Celý systém je implementován pro češtinu i angličtinu, proto je i zde možnost přepínat jazyky. Přepnutí jazyka změní text na stránkách i data dotazovaná z databáze. Stránky přepnuté do češtiny využívají pouze data získaná z českých zdrojů.

Domovská stránka nabízí vyhledávací okno, vyhledávat se dají tituly podle jejich názvu. Výsledkem vyhledávání je seznam všech titulů, jejichž název obsahuje hledaný výraz. Ke každému z vyhledaných filmů je připsána jeho popularita (celkový počet recenzí v databázi).

Dále je na stránce tlačítko pro přejití na žebříček titulů, pod ním je seznam top filmů. Tlačítko pro přejití na stránku analýzy trendů je přítomno pouze pokud je stránka přepnuta do angličtiny. Analýza trendů se pro češtinu neprovádí.

4.4.2 Žebříček



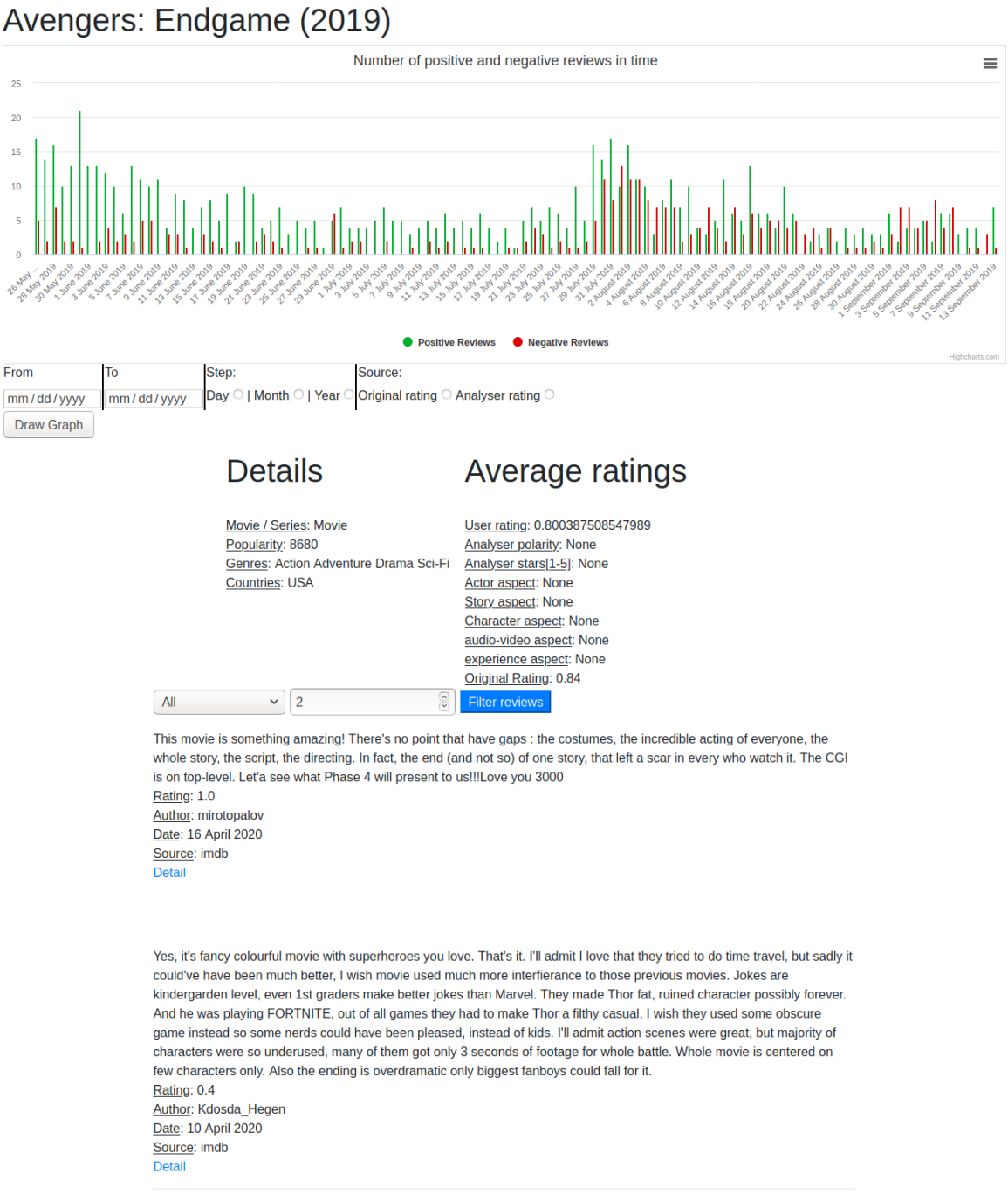
Obrázek 4.2: Žebříček nejlepších / nejhorších titulů.

Jak je vidět na obrázku 4.2, žebříček nabízí limitovaný seznam titulů seřazených vzestupně nebo sestupně podle jednoho z následujících parametrů (hodnoty z tabulky movies):

- Průměrná polarita.
- Průměrný počet hvězd.
- Popularita.
- Průměrné hodnoty jednotlivých aspektů.

Celkově je tedy možnost řadit tituly podle jednoho z osmi parametrů. Limit počtu zobrazených titulů je samozřejmě nastavitelný.

4.4.3 Detail filmu



Obrázek 4.3: Detailní informace o filmu.

Stránka zobrazující detaily o filmu je na obrázku 4.3. V horní sekci stránky je graf zobrazující názor na titul v čase. Nastavením grafu lze měnit časový interval i jeho krok. Dále lze nastavit zdroj názorů na daný film, první možností je originální hodnocení uživatelů, druhou možností jsou výsledky polaritní analýzy.

Ve spodní sekci stránky jsou jednotivé recenze k danému filmu. Ty je možné filtrovat podle aspektu o kterém recenze hovoří. Opět je možnost limitovat maximální počet zobrazených recenzí.

4.4.4 Detail recenze

ID: 79459
This show is honestly great. It is easily the best animation show I have seen in a while on any network. The second the show starts you can tell that the characters fit perfectly with each other, and involved with the craziness that happens in the universe with Rick and Morty, they are immediately hilarious and extremely creative at the same time. A couple episodes in and I could tell that this was a unique show, and that people need to be watching it because it has huge potential. And with Dan Harmon writing and directing at the helm of it all, the magic that this show brings is a rare gem in television that people need to watch.. So yeah I highly suggest this show to anyone who wants a true genuine laugh at least once a week.
Rating: 1.0
Author: tabraham44
Date: 22 January 2014
Source: imdb

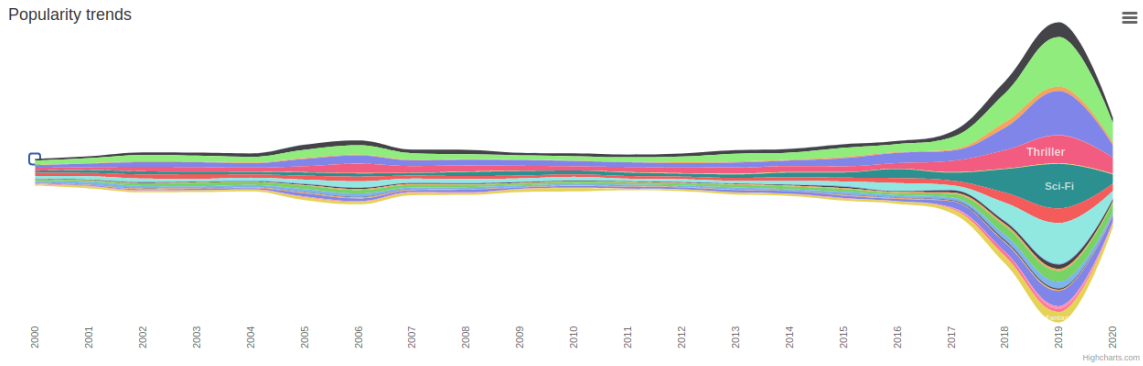
This review was used to calculate the overall score of the movie.

Polarity analysis:
Predicted Value: 0.00359405833296478
Final Polarity: pos
Stars analysis:
One star prediction: -5.45775842666626
Two star prediction: -4.40656185150146
Three star prediction: -2.44675469398499
Four star prediction: -0.287770211696625
Five star prediction: 1.18071663379669
Final result: five_star
Aspect analysis[numbers of positive/negative mentions of aspects]:
Actor positive: 1
Actor negative: 1
Audio-video positive: 0
Audio-video negative: 0
Character positive: 4
Character negative: 0
Experience positive: 0
Experience negative: 1
Story positive: 2
Story negative: 2

Obrázek 4.4: Detailní informace o recenzi.

Stránka detailů recenze je na obrázku 4.4. Obsahuje všechny detaily týkající se dané recenze. Pokud byla recenze již analyzována, obsahuje také výsledky analýz.

4.4.5 Analýza trendů



Obrázek 4.5: Graf trendů popularity.

Stránka obsahuje graf trendu popularity v čase (obrázek 4.5). Popularita je zde definována jako celkový počet recenzí v databázi. Vykreslování grafu je opět nastavitelné. Krok grafu je vždy jeden rok, trendy nemá moc smysl sledovat v menším měřítku. Popularitu je možné sledovat podle typu titulu (film / seriál), podle žánru a podle země původu.

Kapitola 5

Vyhodnocení systému

Tato kapitola popisuje způsob vyhodnocení vytvořeného systému a jeho výsledky. Vyhodnocování přesnosti analyzátorů proběhlo experimentálně. Součástí této kapitoly jsou experimenty se systémem, kterými se snažím zjistit další informace, jako například schopnost systému analyzovat data z jiných domén. Pro tento systém je velice důležitý dostatek dat, proto je zde jejich množství sepsané a porovnané. Poslední část demonstruje použití vytvořeného systému.

Výsledky jsou porovnány s dalšími dostupnými publikacemi.

5.1 Datová sada

Celkově se podařilo nasbírat poměrně velké množství recenzí. Pro porovnání s dalšími publikacemi, tato česká práce [7] používá datovou sadu o velikosti 21 tisíc recenzí, anglická [4] používá 50 tisíc recenzí. Datová sada použita v anglické práci je tvořena recenzemi z *imdb.com*, tato datová sada je pro analýzu sentimentu využívána velice často, knihovna *Pytorch* má vytvořenou třídu pro stažení a práci s touto datovou sadou. Mými zdroji dat byly weby *čsfd.cz* a *fdb.cz* pro české recenze. Zdroji anglických recenzí se staly weby *imdb.com* a *rottentomatoes.com*. Celkové počty nasbíraných recenzí z každého portálu jsou sepsány v tabulce 5.1.

Data byla po jejich získání ukládána do databáze (tabulka *reviews*). Pro trénování analyzátorů byla data uložena do lokálního souboru ve formátu JSON. Z databáze byly použity pouze záznamy, které měly neprázdný text recenze a obsahovaly číselné hodnocení autora.

Recenze byly ze všech webových portálů stahovány s číselným hodnocením autora recenze. Tyto hodnocení byly nejprve normalizovány na desetinné číslo od nuly po jedničku. Recenze byly dále rozděleny podle tohoto hodnocení na pozitivní a negativní. Negativní recenze byly všechny ty, které měly hodnocení menší nebo rovno 0.5. Pozitivní byly ty, které měly hodnocení větší než 0.5. Tímto byla vytvořena datová sada pro analyzátor polarity (pozitivní / negativní).

Stejná data byla využita i pro vytvoření datové sady pro klasifikátor míry pozitivity (jedna až pět hvězdiček). Jednotlivé recenze byly rozděleny do pěti kategorií podle uživatelského hodnocení s krokem 0.2.

Recenze byly dále rozděleny na trénovací (60%), validační (15%) a testovací (25%). Validační data jsou využita již při trénování ke zjištění přesnosti (*accuracy*) a ztráty (*loss*). Tyto hodnoty jsou využity k vybrání epochy ve které byl model nejúspěšnější, tento model je uložen. Testovací data jsou použita až na konci trénování k detailnějšímu přehledu o úspěšnosti modelu.

Všechny recenze jsou uloženy v původním stavu, předzpracovány jsou až v průběhu trénování / analýzy.

Tabulka 5.1: Počet recenzí pro polaritní analyzátor a analyzátor míry polarity

Zdroj	Pozitivní	Negativní	Celkem
IMDb	1 811 198	855 672	2 666 870
Rotten Tomatoes	5 098	74 270	79 368
ČSFD	239 290	758 021	997 311
FDb	37 055	108 562	145 617
Celkem	2 092 641	1 796 525	3 889 166

Data pro aspektové analyzátory byly anotovány ručně, proto jich je také mnohem méně, konkrétní počty v tabulkách 5.2 pro češtinu a 5.3 pro angličtinu. Opět pro srovnání, tato práce [21] má pro angličtinu čtyři datové sady, každá okolo 2 000 vět.

Datová sada byla vytvořena následovně, nejprve byly vybrány náhodné recenze, poté byly rozděleny na věty pomocí nástrojů *spacy* a *nltk*. Jednotlivé věty byly ručně označeny pomocí nástroje *brat*, který běží v prohlížeči. Server pro tento nástroj byl spouštěn lokálně na mém počítači. Každá věta byla anotována žádným až pěti aspekty a polaritou daného aspektu. Takto anotovaná data byla použita pro trénování analyzátorů polarity aspektů. Pro vytvoření datové sady k trénování analyzátorů detekujících aspekty ještě byly doplněny náhodné věty, datová sada pro analyzátory detekující aspekt byla tedy oproti hodnotám v tabulkách dvakrát větší.

Tabulka 5.2: Počet vět pro klasifikátory polarity aspektů v češtině

Aspekt	Pozitivní	Negativní	Celkem
Herec	127	107	234
Postava	69	69	138
Audio vizuální efekty	97	78	175
Příběh	81	81	162
Zkušenost	79	83	162
Celkem	453	418	871

Tabulka 5.3: Počet vět pro klasifikátory polarity aspektů v angličtině

Aspekt	Pozitivní	Negativní	Celkem
Herec	125	127	252
Postava	89	92	181
Audio vizuální efekty	107	99	206
Příběh	164	168	332
Zkušenost	104	107	211
Celkem	589	593	1 182

5.2 Maximální dosažená přesnost

V této sekci jsou porovnány maximální dosažené přesnosti všech prováděných analýz vzhledem k použité metodě. Analyzátory byly trénovány na dříve popsaných datových sadách. Pro trénování analyzátorů založených na neuronové síti se data vybalancovala a rozdělila na trénovací, testovací a validační sady. Využita byla tedy většina dat. Oproti tomu při trénování analyzátorů založených na metodě podpůrných vektorů a naivní Bayesovské metodě, jsem byl z důvodu paměťové náročnosti způsobu trénování nucen vždy použít jen část trénovacích dat.

Sledována je pouze přesnost (*accuracy*) analyzátorů, která je definována jako poměr mezi počtem správných předpovědí (*true positive+true negative*) a celkovým počtem testovacích příkladů. Recenze byly před trénováním i testováním předzpracovány stejně jako bylo popsáno v kapitole 4.

Snažím se zde odpovědět na otázku, zda-li je vůbec při analýze recenzí nutné používat relativně komplikovanější a náročnější metody jako jsou neuronové sítě, nebo stačí jednodušší metody jako podpůrné vektory nebo naivní Bayesova.

5.2.1 Analýza polarity

Prvním testovaným analyzátorem je analyzátor polarity. Analyzuje na úrovni celého dokumentu (recenze) a přiřazuje mu pozitivní nebo negativní třídu.

Tabulka 5.4: Maximální přesnost použitých metod

Metoda	**Přesnost (*Accuracy*)**	
	CZ	EN
Neuronová síť (oboustranné LSTM)	0.81	0.88
Metoda podpůrných vektorů	0.80	0.83
Naivní Bayesovská metoda	0.69	0.69

Z tabulky 5.4 je patrné, že nejvyšší přesnost má dle očekávání systém založený na hlubokém učení. Z tohoto důvodu byl zvolen pro automatizovanou analýzu recenzí. Tento klasifikátor (obzvlášť pro angličtinu) dosáhl poměrně vysoké přesnosti, za nejmodernějšími systémy však ještě zaostává. Nejlepších výsledků by se pravděpodobně dosáhlo využitím systému jako je BERT a jeho doladěním na doménu filmových recenzí. Chtěl jsem si však klasifikátor z osobních důvodů implementovat sám, proto jsem se vydal touhle cestou.

Porovnáním přesností lze vidět, že metoda podpůrných vektorů moc nezaostává a její přesnost by byla přijatelná. V tomto případě by tedy asi nebyla potřeba využívat komplikovanějších metod.

5.2.2 Analýza míry pozitivity

Druhým typem analyzátoru je analyzátor míry pozitivity. Také analyzuje na úrovní dokumentu, přiřazuje jednu z pěti tříd (jedna až pět hvězd).

Tabulka 5.5: Maximální přesnost použitých metod

Metoda	**Přesnost (*Accuracy*)**	
	CZ	EN
Neuronová síť (oboustranné LSTM)	0.62	0.55
Metoda podpůrných vektorů	0.46	0.53
Naivní Bayesovská metoda	0.45	0.52

Pro automatizovanou analýzu je opět používán analyzátor založený na hlubokém učení kvůli jeho přesnosti. Ostatní metody však moc nezaostávají, takže by byly použitelné také. Hned na první pohled lze v tabulce 5.5 vidět, že oproti analýze polarity mají všechny metody přesnost nižší. To se dá očekávat, protože tento typ klasifikace je obtížnější. Z důvodu poměrně velké změny v přesnosti jsem se však rozhodl analyzovat chyby hlouběji. Analýza chyb byla provedena na anglické datové sadě s využitím klasifikátoru založeném na hlubokém učení. Celkový počet testovacích příkladů byl 146 664, z toho bylo 60 565 chyb (to znamená přesnost 0.58). U každé chyby bylo zaznamenáno, jestli se klasifikátor dopustil chyby pouze o jednu hvězdu nebo více. Výsledky jsou následující (nalevo od pomlčky je předpokládaná odpověď analyzátoru).

Počet chyb o jednu hvězdu:

- jedna hvězda - 792
- dvě hvězdy - 4 132
- tři hvězdy - 10 404
- čtyři hvězdy - 4 921
- pět hvězd - 9 361

Počet chyb o více než jednu hvězdu:

- jedna hvězda - 4 434
- dvě hvězdy - 9 890
- tři hvězdy - 6 324
- čtyři hvězdy - 7 960
- pět hvězd - 2 347

Celkově bylo 29 610 chyb o jednu hvězdu, to je 48 % všech chyb. Předpokládám, že tohle je z důvodu malého rozdílu v obsahu textu recenzí mezi těmi blízko u sebe v hodnocení.

5.2.3 Analýza aspektů

Aspektová analýza se skládá ze dvou úkolů, detekci aspektu (výsledky v tabulce 5.6) a určení polarity daného aspektu (tabulka 5.7).

Tabulka 5.6: Detekce aspektu

Metoda	**Přesnost (*Accuracy*)**	
	CZ	EN
Neuronová síť (oboustranné LSTM)	0.72	0.71
Metoda podpůrných vektorů	0.74	0.73
Naivní Bayesovská metoda	0.72	0.66

Tabulka 5.7: Určení polarity aspektu

Metoda	**Přesnost (*Accuracy*)**	
	CZ	EN
Neuronová síť (oboustranné LSTM)	0.69	0.69
Metoda podpůrných vektorů	0.65	0.68
Naivní Bayesovská metoda	0.64	0.67

Na české datové sadě byla přesnost zjištěna pro aspekt „herec“, na anglické pro aspekt „příběh“. Tyto aspekty byly vybrány, protože mají nejvíce anotovaných vět. Důvodem je nutnost rozdělení datových sad na trénovací a testovací části. Klasifikátory ostatních aspektů mají velice podobnou přesnost.

Z výsledků je patrné, že všechny metody mají velice podobné výsledky. Tohle je způsobeno menším počtem dat. Pro automatickou analýzu je opět zvolena neuronová síť.

5.3 Využití analyzátorů v jiných doménách

Těmito experimenty je testována schopnost polaritních analyzátorů pracovat na datech z jiných domén. Filmových recenzí je obrovské množství, navíc jsou již anotovány autorem. Kdyby se tedy ukázalo, že analyzátory natrénované na těchto datech jsou úspěšné i v jiných doménách, mohlo by to ulehčit práci s hledáním trénovacích dat. Testy byly provedeny na následujících datových sadách (vyvážená byla pouze sada Mallcz):

- Facebook - česká datová sada, necelých 5 000 příspěvků.
- Mallcz - česká datová sada, použito 10 000 recenzí.
- Hudební nástroje - anglická datová sada, jedná se o recenze z webového portálu *Amazon*. Přes 10 000 recenzí.
- Automobilový průmysl - anglická datová sada, jedná se o recenze z webového portálu *Amazon*. Přes 20 000 recenzí.

Tabulka 5.8: Polaritní analyzátory v jiných doménách

Metoda	**Přesnost (*Accuracy*)**			
	Facebook	Mallcz	Hudební nástroje	Automobilový průmysl
Neuronová síť	0.57	0.51	0.83	0.75
Podpůr. vekt.	0.52	0.47	0.77	0.59
Naivní Bayes.	0.43	0.45	0.14	0.12

Ve výsledcích jde vidět (tabulka 5.8), že obecně jsou úspěšnější anglické datové sady. To bude pravděpodobně tím, že čeština je bohatější jazyk a tím těžší pro klasifikátor na naučení.

Přesnost naivní bayesovské metody v anglických datových sadách je opravdu zvláštní. Všechno bylo zkontrolováno, test byl proveden několikrát a vždy vyšel stejně.

Z výsledků vyplývá, že analyzátory kromě třech případů nejsou velice úspěšné. Filmové recenze jsou nejspíše příliš konkrétní doménou a pro trénování analyzátorů do jiných domén se moc nehodí. Klasifikátory by se daly doladit (dotrénovat na konkrétní doméně) a tím zvýšit jejich přesnost. Toto už však testováno nebylo.

5.4 Přesnost v závislosti na velikosti trénovacích dat

Zde chci porovnat přesnosti metod polaritní analýzy v závislosti na počtu trénovacích příkladů. Cílem je zjistit potřebné množství dat pro natrénovaní analýzátoru s přijatelnou přesností. Využity jsou části již popsaných datových sad. Testování probíhalo na náhodně vybrané sadě o velikosti 100 000 recenzí. Trénovací sada (také vytvořená náhodně) se postupně zvětšovala.

Tabulka 5.9: Přesnost v závislosti na velikosti trénovacích dat

Počet recenzí	**Neuronová síť**		**SVM**		**Bayes**	
	CZ	EN	CZ	EN	CZ	EN
50	0.65	0.54	0.66	0.69	0.54	0.52
100	0.66	0.46	0.67	0.71	0.62	0.66
200	0.70	0.62	0.77	0.70	0.71	0.64
500	0.70	0.67	0.78	0.76	0.72	0.67
1000	0.72	0.65	0.77	0.77	0.74	0.68
2000	0.72	0.66	0.80	0.78	0.72	0.68
5000	0.74	0.68	0.79	0.80	0.73	0.69
10000	0.78	0.75	0.81	0.80	0.75	0.71

V tabulce 5.9 jde vidět, že po trénování na datové sadě o velikosti 5 000 recenzí již za svým maximem zaostává pouze klasifikátor založený na neuronové síti. Z toho vyplývá, že pro menší datové sady je lepší využít jiné metody.

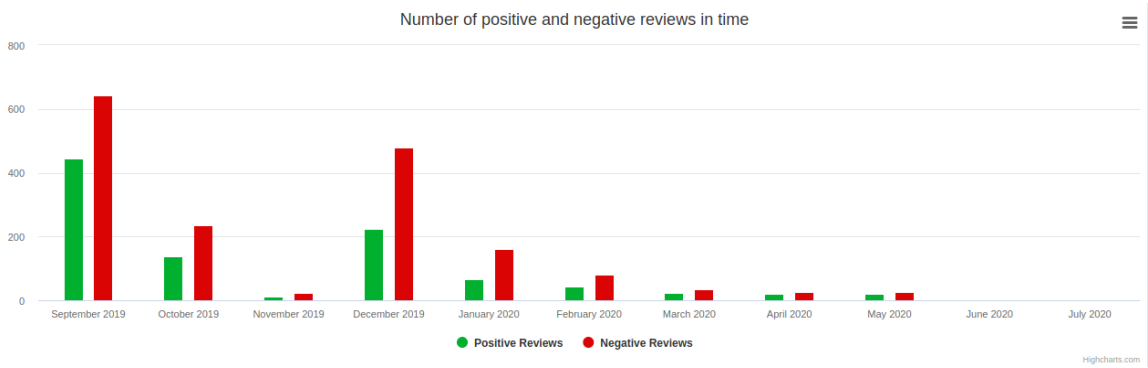
Cekově z výsledků vyplývá, že pro natrénování poměrně přijatelně přesného klasifikátoru není potřeba příliš mnoho dat (v poměru k počtu dostupných recenzí).

5.5 Demonstrace systému

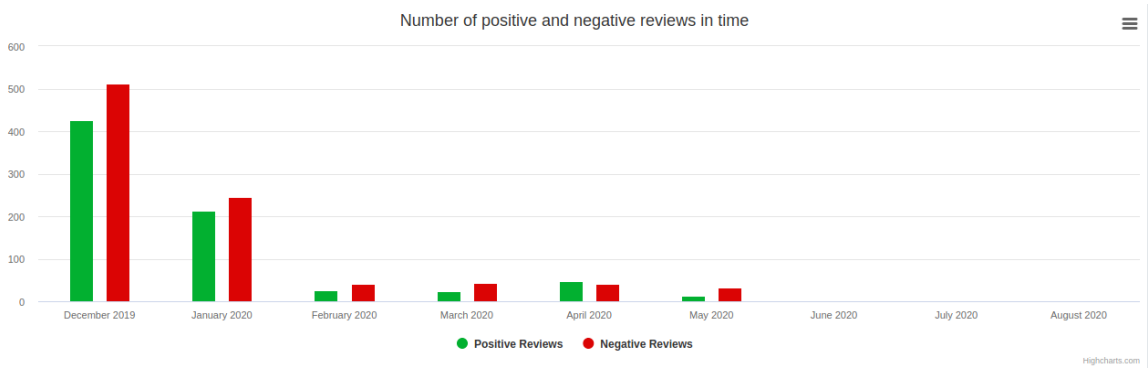
V této sekci chci ukázat několik příkladů jak se dá vytvořený systém využít a jaké informace se dají za jeho pomoci získat.

5.5.1 Správné vydání filmu

Pro analýzu dat byl použit dříve popsaný graf polarity (kapitola 4.4.3).



Obrázek 5.1: Počet pozitivních a negativních hodnocení v čase pro film *Ad Astra*

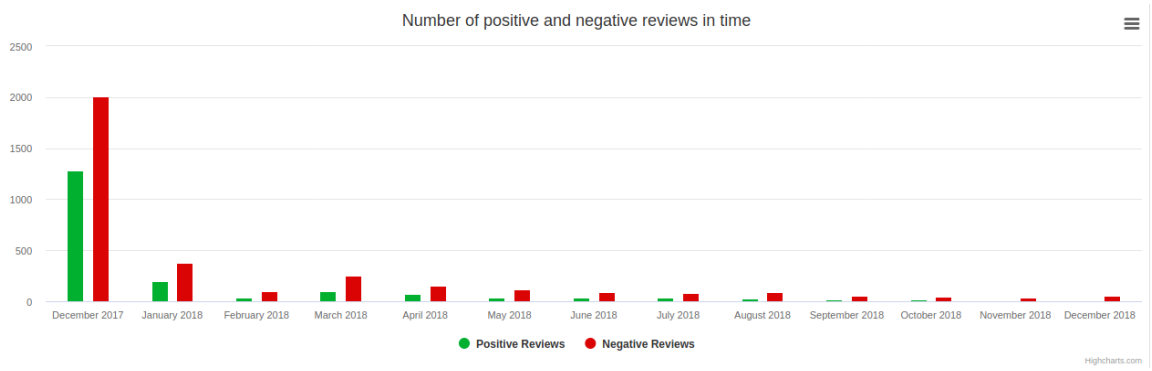


Obrázek 5.2: Počet pozitivních a negativních hodnocení v čase pro film *Cats*

Oba filmy jsou poměrně neúspěšné, celkové hodnocení mají okolo 50 %. Pohledem na oba grafy 5.1 a 5.2 lze zjisti, že jejich průběhy jsou velice podobné. Po vydání poměrně vysoká popularita (velký počet pozitivních i negativních recenzí). Po čase však tato popularita rapidně klesá. Pohledem na graf 5.1 lze vidět v prosinci 2019 velký nárůst popularity. Po krátkém zkoumání lze zjistit, že v tomto měsíci byl film vydán na DVD, to je zhruba 3 měsíce po vydání. Tento fakt velice pozvedl popularitu. Oproti tomu film *Cats* (graf 5.2) žádný takový nárůst nemá. DVD tohoto filmu bylo totiž vydáno až téměř po půl roce od vydání, podle grafů tohle byla chyba. Po tak dlouho době již vyprší nadšení z nového filmu a jeho marketingové kampaně.

5.5.2 Pravdivost celkových hodnocení

U převážné většiny titulů je originální celkové hodnocení a celkové hodnocení vytvořené systémem velice podobné. Zde prezentuji jeden příklad kdy tohle není pravdou.



Obrázek 5.3: Počet pozitivních a negativních hodnocení v čase pro film *Star Wars: Episode VIII - The Last Jedi (2017)*

Již na první pohled lze v grafu 5.3 vidět, že se nejedná o velice úspěšný film. Celkové hodnocení systému(vytvořené průměrem hodnocení uživatelů *imdb*) je 41 %. Při návštěvě webových stránek *imdb* však vidíme hodnocení 7/10. Velice podobný případ lze nalézt i u 2 roky staršího předchůdce *Star Wars: Episode VII - The Force Awakens*. Tam systém tvrdí 57 %, na *imdb* je však napsáno 7,9/10.

Kapitola 6

Závěr

Cílem této práce bylo navrhnout a implementovat systém schopný pravidelně získávat, indexovat a analyzovat recenze filmů v češtině a angličtině. Cíl práce byl splněn.

V práci je prostudována teorie základních i pokročilých metod pro rozpoznávání postojů a zpracování přirozeného jazyka. Dále byly určeny vhodné zdroje recenzí filmů a seriálů pro oba jazyky. Těmito zdroji se staly webové portály *csfd.cz*, *fdb.cz*, *imdb.com* a *rottentomatoes.com*. Recenze z těchto portálů jsou pravidelně stahovány a indexovány v relační databázi. Současně jsou v databázi téměř čtyři miliony recenzí filmů a seriálů.

Využitím těchto recenzí byly natrénovány analyzátory třech typů. Prvním z nich je analyzátor polarity na úrovni dokumentu určující celkový postoj recenze (pozitivní / negativní). Tento analyzátor dosahuje přesnosti až 88 % pro angličtinu a 81 % pro češtinu. Druhým typem je analyzátor míry pozitivity pracující také na úrovni dokumentu, tento analyzátor rozděluje recenze do pěti tříd (jedna až pět hvězd). Tato analýza dává detailnější informace o recenzi, ale má nižší přesnost, a to 55 % pro angličtinu a 62 % pro češtinu. Po detailnější analýze chyb toho klasifikátoru se ukázalo, že má problém s rozlišením recenzí s podobným hodnocením (například recenze s pěti a čtyřmi hvězdičkami). Posledním typem je analýza aspektů, která dává nejdetailnější informace. Předem bylo definováno pět aspektů, které jsou u recenzí sledovány (herec, postava, příběh, audio vizuální efekty a zkušenost diváka). Analýza je rozdělena do dvou kroků, nejprve jsou jednotlivé aspekty nalezeny a poté ohodnoceny polaritou (pozitivní / negativní).

Výsledky z automaticky prováděných analýz jsou ukládány do relační databáze a zobrazovány ve webovém rozhraní (`http://athena1.fit.vutbr.cz:8078/`). Rozhraní uživateli nabízí vyhledávat a filtrovat filmy a seriály. Každý film má vytvořenou časovou osu názorů na daný film s detailní analýzou jednotlivých recenzí týkajících se filmu. Dále jsou sledovány trendy titulů podle žánru, země původu a jestli je daný titul film či seriál.

V práci bych pokračoval anotací dalších dat pro aspektovou analýzu, tím by se zvýšila její přesnost, popřípadě bych otestoval další metody implementace analyzátorů. Dále bych rád udělal hezčí a dynamičtější webové rozhraní. Pro přilákání uživatelů na platformu by bylo dobré jim dát možnost přímo interagovat s klasifikátory, dnes je téma strojového učení velice populární.

Literatura

[1] *Databáze* [online]. [cit. 2020-03-19]. Dostupné z: https://www.oracle.com/cz/database/what-is-database.html.

[2] *How Much Data is Enough for Analysis into Various Fields?* [online]. 2019. 2019-06-20 [cit. 2020-03-17]. Dostupné z: https://medium.com/@cogitotech/how-much-data-is-enough-for-analysis-into-various-fields-b4f06c2ce99b.

[3] Agency, J. *The Most Effective Web Scraping Methods* [online]. 2018. 2019-02-27 [cit. 2020-03-19]. Dostupné z: https://expertise.jetruby.com/the-most-effective-web-scraping-methods-62e7e34ada69.

[4] Ali, N., Hamid, M. a Youssif, A. SENTIMENT ANALYSIS FOR MOVIES REVIEWS DATASET USING DEEP LEARNING MODELS. *International Journal of Data Mining Knowledge Management Process*. Květen 2019, roč. 09, s. 19–27.

[5] Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Hasan, M. et al. The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches. *CoRR*. 2018, abs/1803.01164. Dostupné z: http://arxiv.org/abs/1803.01164.

[6] Bahdanau, D., Cho, K. a Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*. 2015, abs/1409.0473.

[7] Çano, E. a Bojar, O. Sentiment Analysis of Czech Texts: An Algorithmic Survey. *CoRR*. 2019, abs/1901.02780. Dostupné z: http://arxiv.org/abs/1901.02780.

[8] Devika, M., Sunitha, C. a Ganesh, A. Sentiment Analysis: A Comparative Study on Different Approaches. *Procedia Computer Science*. 2016, roč. 87, s. 44 – 49. Fourth International Conference on Recent Trends in Computer Science Engineering (ICRTCSE 2016). Dostupné z: http://www.sciencedirect.com/science/article/pii/S187705091630463X. ISSN 1877-0509.

[9] Devlin, J., Chang, M., Lee, K. a Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*. 2018, abs/1810.04805, s. 4171—-4186. Dostupné z: http://arxiv.org/abs/1810.04805.

[10] Drakos, G. *All you need to know about NLP Text Preprocessing* [online]. 2019. 2019-06-13 [cit. 2020-03-28]. Dostupné z: https://gdcoder.com/all-you-need-to-know-about-nlp-text-preprocessing/.

[11] Filatova, E. Irony and Sarcasm: Corpus Generation and Analysis Using Crowdsourcing. In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Language Resources Association (ELRA), Květen 2012, s. 392–398. Dostupné z: http://www.lrec-conf.org/proceedings/lrec2012/pdf/661_Paper.pdf. ISBN 978-2-9517408-7-7.

[12] Garg, A. a Roth, D. Understanding Probabilistic Classifiers. In: De Raedt, L. a Flach, P., ed. *Machine Learning: ECML 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, s. 179–191. ISBN 978-3-540-44795-5.

[13] Garreta, R. *A Gentle Guide to Machine Learning* [online]. 2015. 2015-08-27 [cit. 2020-04-10]. Dostupné z: https://monkeylearn.com/blog/gentle-guide-to-machine-learning/.

[14] kar. *Co spojuje internetové trolly? Psychologové vypracovali jejich nelichotivý profil* [online]. 2017. 2017-07-11 [cit. 2020-03-18]. Dostupné z: https://ct24.ceskatelevize.cz/veda/2178160-co-spojuje-internetove-trolly-psychologove-vypracovali-jejich-nelichotivy-profil.

[15] keitakurita. *A Deep Dive into the Wonderful World of Preprocessing in NLP* [online]. 2019. 2019-11-06 [cit. 2020-03-25]. Dostupné z: https://mlexplained.com/2019/11/06/a-deep-dive-into-the-wonderful-world-of-preprocessing-in-nlp/.

[16] Malik, F. *NLP: Text Data To Numbers* [online]. 2019. 2019-06-25 [cit. 2020-03-31]. Dostupné z: https://medium.com/fintechexplained/nlp-text-data-to-numbers-d28d32294d2e.

[17] Medhat, W., Hassan, A. a Korashy, H. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*. 2014, roč. 5, č. 4, s. 1093 – 1113. Dostupné z: http://www.sciencedirect.com/science/article/pii/S2090447914000550. ISSN 2090-4479.

[18] Mikolov, T., Corrado, G., Chen, K. a Dean, J. Efficient Estimation of Word Representations in Vector Space. In:. Leden 2013, s. 1–12.

[19] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. a Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In: Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z. a Weinberger, K. Q., ed. *Advances in Neural Information Processing Systems 26*. Red Hook, NY, USA: Curran Associates, Inc., 2013, s. 3111–3119. Dostupné z: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[20] Mitchell, R. *Web scraping with Python: Collecting more data from the modern web*. 2nd. "O'Reilly Media, Inc.", 2018. ISBN 978-1491985571.

[21] Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S. et al. SemEval-2016 Task 5: Aspect Based Sentiment Analysis. In: *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: Association for Computational Linguistics,

červen 2016, s. 19–30. Dostupné z: https://www.aclweb.org/anthology/S16-1002. ISBN 978-1-941643-95-2.

[22] Ptáček, T., Habernal, I. a Hong, J. Sarcasm Detection on Czech and English Twitter. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, Srpen 2014, s. 213–223. Dostupné z: https://www.aclweb.org/anthology/C14-1022.

[23] Schultz, J. *How Much Data is Created on the Internet Each Day?* [online]. 2014. 2019-06-08 [cit. 17. března 2020]. Dostupné z: https://blog.microfocus.com/how-much-data-is-created-on-the-internet-each-day/#.

[24] Stecanella, B. *An Introduction to Support Vector Machines (SVM)* [online]. 2017. 2017-06-22 [cit. 2020-04-13]. Dostupné z: https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/.

[25] Stecanella, B. *A practical explanation of a Naive Bayes classifier* [online]. 2017. Aktualizováno 25. 5. 2017 [cit. 20. června 2020]. Dostupné z: https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/.

[26] Stencanella, R. *The Beginner's Guide to Text Vectorization* [online]. 2017. 2017-09-21 [cit. 2020-03-31]. Dostupné z: https://monkeylearn.com/blog/beginners-guide-text-vectorization/.

[27] Sun, C., Qiu, X., Xu, Y. a Huang, X. How to Fine-Tune BERT for Text Classification? *CoRR*. 2019, abs/1905.05583, s. 194–206. Dostupné z: http://arxiv.org/abs/1905.05583.

[28] Taboada, M., Brooke, J., Tofiloski, M., Voll, K. a Stede, M. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*. 2011, roč. 37, č. 2, s. 267–307. Dostupné z: https://doi.org/10.1162/COLI_a_00049.

[29] Veselovská, K. *Sentiment analysis in Czech*. Praha, Czechia: ÚFAL, 2017. Studies in Computational and Theoretical Linguistics, sv. 16. ISBN 978-80-88132-03-5.

[30] Weston, J., Chopra, S. a Bordes, A. Memory Networks. *CoRR*. 2015, abs/1410.3916.

[31] Wójcik, R. *Unsupervised Sentiment Analysis* [online]. 2019. 2019-11-26 [cit. 2020-04-10]. Dostupné z: https://towardsdatascience.com/unsupervised-sentiment-analysis-a38bf1906483.

[32] Zhang, L., Wang, S. a Liu, B. Deep learning for sentiment analysis: A survey. *WIREs Data Mining and Knowledge Discovery*. 2018, roč. 8, č. 4, s. e1253. Dostupné z: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1253.

[33] ShieldSquare. *Types of Scraping Techniques: How to Prevent Scraping* [online]. 2019. 2019-12-03 [cit. 2020-03-19]. Dostupné z: https://www.shieldsquare.com/what-are-the-different-scraping-techniques/.

Příloha A

Obsah přiloženého paměťového média

- doc - Složka obsahující text technické zprávy a jeho zdrojový kód.
- src - složka zdrojových kódů. Obsahuje podsložky:
  - analyzer - Složka obsahující skripty pro trénování analyzátorů a následnou analýzu.
  - bash_scripts - Skripty ke spouštění.
  - downloader - Složka se skripty pro stahování recenzí ze zdrojových webů.
  - helper_scripts - Pomocné skripty.
  - sentiment_analytics - Webové rozhraní implementované v aplikačním rámci *Django*.
- plakat.pdf - Plakát.
- readme.txt - Textový soubor popisující postup instalace systému.