Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra teoretické informatiky

Diplomová práce

# Nástroje a postupy pro žurnalistiku nové generace

***Bc. Jakub Bartel***

Vedoucí práce: Ing. Pavel Kordík, Ph.D.

4. května 2015

# Poděkování

Děkuji Ing. Pavlu Kordíkovi, Ph.D. za vedení této práce. Děkuji celé mé rodině za podporu a vytvoření báječných podmínek pro celé mé studium.

# Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 6. května 2015 .....

## Odkaz na tuto práci

Bartel, Jakub. *Nástroje a postupy pro žurnalistiku nové generace.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

# Abstrakt

Tato práce popisuje praktické nasazení přístupů z oblasti strojového učení v oboru online žurnalistiky. Zabývá se sběrem, zpracováním a dalším použitím textových dat, sociálních signálů a dat z analytických systémů pro sledování návštěvnosti webů. Systém je následně schopen personalizovat sledování světového dění – aktuálně populární témata a události – a navíc predikovat budoucí popularitu jednotlivých témat. K tomu používá nástroj pro obsahové doporučování, který netriviálním způsobem kombinuje s nástrojem prediktivního modelování.

**Klíčová slova** žurnalistika, strojové učení, zpracování dat, zpracování textu, textmining doporučování, predikce, shlukování, vizualizace

# Abstract

This thesis deals with practical implementation of the methods used in the field of machine learning into the area of online journalism. It explains collection, processing and further usage of text data, social signals and data received from analytic systems for web traffic monitoring. The system described in the thesis is able to personalize the process of following the development of up-to-date world events – topics and events that are currently popular – as well as to predict potential popularity of individual topics in the future. For this purpose it uses a tool for content recommendation, which is combined with a tool for predictive modelling in a nontrivial manner.

**Keywords** journalism, machine learning, textmining, data preprocessing, recommendation, prediction, clustering, vizualization

# Obsah

# Seznam obrázků

# Seznam tabulek

# Úvod

Metody *strojového učení* nacházejí své uplatnění v mnoha oblastech lidských činností. Spojení nárůstu dostupného výpočetního výkonu a objemu a přístupnosti dat přináší stále nové a nové příležitosti v oborech, kde bylo využití strojového zpracování informací doposud nemožné či těžko představitelné. Jednou z takových oblastí je právě žurnalistika. Množství zpráv, reportáží a aktualit, které jsou dennodenně publikovány, vytváří prostředí, v němž je velmi obtížné třídit dostupné informace, vyhledávat potenciálně zajímavé události a sledovat vývoj aktuálního dění. Zároveň roste potřeba o aktuálních událostech informovat v co nejkratším možném čase, efektivně využívat dostatek ověřených zdrojů a sledovat vývoj témat v čase.

Své uplatnění tak na trhu nacházejí služby, které zpracovávají obsah dostupný na internetu, dokáží ho třídit, kategorizovat a následně přehledně prezentovat uživateli. Tyto nástroje však nabízejí omezené funkcionality v oblasti pokročilé datové analytiky a automatické personalizace. Použití vhodných postupů a metod pro zpracování velkého objemu dat může vést k podstatnému zefektivnění práce žurnalistů a zároveň zkvalitnění výstupů jejich činnosti např. ve smyslu přesného tématického cílení na zájmy jejich cílové skupiny – čtenářů.

Nedílnou součást novodobé online publikační činnosti tvoří neustálá práce s vydávaným obsahem. Ta nekončí v momentě zveřejněn článků, ale díky možnostem spojeným např. s *realtime statistikou* aktuální návštěvnosti webu je možné pružně analyzovat potenciální příležitosti a následně je využívat pro přivedení nových návštěvníků. V tomto kontextu se nesmírně důležitým stává používání sociálních sítí jako kanálu k oslovení široké komunity. Sociální sítě umožňují mimo přirozeného (organického) zobrazení a šíření příspěvků také znásobení jejich dosahu skrze integrované reklamní systémy. Z druhého pohledu jsou *sociální signály* výborným indikátorem

skutečného zájmu o daný článek a v širším kontextu také celkového zájmu o téma, kterého se článek týká.

Propojení a strukturalizace veškerých dostupných dat umožňuje sledovat a analyzovat současné dění a následně vyhledávat aktuálně důležitá či populární témata. Díky algoritmům strojového učení je však možné se posunout o další krok vpřed a naučit stroj samostatně na základě zkušenosti budoucí popularitu s předstihem *predikovat*. Takový druh informace znamená pro redaktory neocenitelnou pomůcku při rozhodování jakému tématu věnovat svůj čas a kromě jiného jim přináší i velkou konkurenční výhodu. Vstupem do *predikční analýzy* zároveň nemusejí být pouze data veřejně dostupná, ale také data z vlastních analytických systémů. Predikce se následně přizpůsobí přímo charakteru konkrétního vydavatelství a dokáže pracovat s mnohem vyšší přesností. Pro zohlednění preferencí jednotlivých redaktorů nachází své využití tzv. *rekomendační systémy*, čili algoritmy pro *personalizaci* obsahu.

Vývoj a provozování aplikací, které zpracovávají a analyzují velké množství dat, vyžaduje specifický přístup z hlediska návrhu a implementace jednotlivých součástí systému a především jejich následného propojení. Je třeba dbát na to, aby systém byl v dostatečné míře škálovatelný a splňoval dnešní vysoké nároky na rychlost odezvy, spolehlivost a dostupnost.

Tato práce se zabývá právě budováním takového systému, detailním popisem propojení jednotlivých přístupů strojového učení pro řešení skutečného problému a popisuje implementaci reálně použitelné aplikace.

Kapitola 1

# Teoretická část

## 1.1 Online žurnalistika

Nesmírná konkurence napříč online zpravodajskými portály nutí vytvářet nové postupy pro sledování událostí a (nejen) světového dění tak, aby se důležitá informace předala vlastním čtenářům v co nejkratším čase a zároveň v co nejvyšší kvalitě a rozsahu. Běžnou praxí je vedle vytváření vlastního originálního obsahu také sledování velké množiny tématicky relevantních zdrojů, které pokrývají oblast zájmu daného vydavatelství. Novinář kontroluje své zdroje, provádí selekci zajímavých či důležitých událostí a následně publikuje souhrn a nebo vlastní názory v novém článku. Mezi možné zdroje lze zařadit např. články vydané na jiných portálech, tiskové zprávy nebo příspěvky na sociálních sítích.

## 1.2 Strojové učení

Strojové učení neboli machine learning se během posledních let stalo jednou ze základních částí informačních technologií a současně, ač často skrytou, součástí běžného života. S rostoucím množstvím generovaných a dále dostupných dat lze očekávat, že inteligentní datová analýza bude důležitým prvkem pro další technologický pokrok.

Využití strojového učení v různých oblastech lidských činností umožňuje zjednodušení, zrychlení a zpřesnění prováděných prací, zároveň přináší částečně automatickou kontrolu kvality výstupů [14]. Část strojového učení zabývající se zpracováním textů lze využít právě v oboru žurnalistiky, se kterou se velmi úzce pojí. Dále je možné zpracování textu obohacovat např. o procesy typu shlukování, klasifikace, predikce, kolaborativní filtrování.

## 1.3 Rekomendační systémy

Rekomendační systémy patří mezi softwarové nástroje a techniky poskytující návrhy a doporučení objektů (*items*) konkrétním uživatelům (*users*). [1] Tyto návrhy následně ovlivňují uživatele v procesu rozhodování o provedení určité akce, např. výběr položky zboží k nákupu, přehrání videa v online videoportálu, nebo přečtení online zpráv a novinek.

Primárně jsou rekomendační nástroje vhodné na místa, kde potenciálně obrovské množství informací a objektů neumožňuje uživateli se fundovaně rozhodovat. Informace z jeho profilu však mohou být dostatečně obsáhlé na to, aby systém automaticky rozhodování zjednodušil omezením nebo vhodným seřazením možností prezentovaných uživateli.

Dále je možné rekomendační systém považovat za určitý druh filtru, který odstraňuje nerelevantní informace přesně na základě chování a preferencí uživatele. Zároveň tak odhaduje kvalitu a relevanci dat vzhledem k osobním charakteristikám každého uživatelského profilu. Přirozeně tak dochází k personalizaci nabízeného obsahu a tím obecně k větší spokojenosti uživatelů.

Nejčastějším výstupem rekomendačního systému je vhodným způsobem seřazený seznam objektů. Toto seřazení může být generováno na základě modelu chování uživatele a jeho osobních preferencí, nebo na základě chování skupiny uživatelů jemu podobných. Rozlišují se dvě základní skupiny rekomendačních systémů:

- filtrování na základě obsahu (*content-based filtering*)
- kolaborativní filtrování (*collaborative filtering*)

a dále systémy založené na:

- demografických údajích
- komunitě uživatelů
- hybridní rekomendační systémy

### 1.3.1 Filtrování na základě obsahu

Systém sestavuje doporučení na základě objektů, se kterými již uživatel v minulosti interagoval a snaží se najít takové nové objekty, které jsou v určité míře podobnosti nejblíže historii uživatele. Míra podobnosti je vyčíslena na základě porovnání atributů (*features*) a kategorií objektů. Doporučení tak

vychází z předpokladu, že uživatele s interakcemi v určité oblasti, např. akční filmy, budou dále zajímat akční a jim podobné filmy.

Nevýhoda doporučování na základě obsahu spočívá v malé diverzifikaci výstupní množiny objektů a přílišné specializaci [1] – systém hledá pouze nejvíce podobné objekty, což mnohdy potenciál doporučení značně omezuje, např. při koupi telefonu je záhodno nabídnout pouzdro na telefon namísto dalších, podobných telefonů. Systém je také velmi citlivý na kvalitu a množství informace obsažené v příznakových vektorech objektů.

### 1.3.2 Kolaborativní filtrování

Na rozdíl od filtrování na základě obsahu využívá kolaborativní filtrování informace o ostatních uživatelích systému. Základní myšlenka spočívá v předpokladu, že uživatele, který určitým způsobem ohodnotil objekt $u$, bude zajímat objekt $v$, pokud jiný uživatel ohodnotil objekt $u$ podobným způsobem a zároveň kladně hodnotil objekt $v$. Doporučení tedy vychází z reálného hodnocení a interakcí uživatelů s objekty, namísto spoléhání se na popis objektů, který buď nemusí být vůbec k dispozici nebo není dostatečně kvalitní. Systém dále může nabídnout objekty s velmi rozdílným obsahem, pokud uživatelé o takové objekty mají zájem.

Nutnou součástí systému je historie interakcí uživatelů. Ta během nasazení systému nemusí být k dispozici a systém tak zvyšuje svoji přesnost až s přibývajícími daty o uživatelích.

### 1.3.3 Demografické filtrování

Rekomendační systémy založené na demografickém profilu uživatele předpokládají, že uživatelům z různých demografických skupin je třeba generovat různý obsah. Roli může hrát země původu uživatele, věk apod. Tento přístup je populární především v oblasti marketingu.

### 1.3.4 Komunitní filtrování

Při komunitním filtrování probíhá doporučení na základě preferencí přátel uživatele. Ukazuje se, že lidé více spoléhají na doporučení svých přátel, než jiných anonymních osob. Ve spojení s rostoucí oblibou sociálních sítí se tato oblast často nazývá *social recommender systems*. Systém automaticky analyzuje spojení v sociální síti a mapuje interakce přátel na profil uživatele, což vede k přirozenému filtrování obsahu.

### 1.3.5 Hybridní systém

Hybridní rekomendační systém kombinuje přístupy předchozích systémů. Je nasazován z důvodu potlačení nevýhod a omezení, jakými jsou např. malá diverzifikace výstupní množiny, *cold-start problem* – kolaborativní filtr nemůže doporučovat objekty, pokud nemají žádné zaznamenané interkace, což ale nijak neomezuje filtrování na základě obsahu, kombinací přístupů je tak možné dosáhnout požadovaných výsledků –, či umožňují např. implementovat zadání specifických požadavků na výstup (uživatelské filtrování) nebo upravovat seřazení objektů na základě dalších atributů (tzv. *boosting*).

### 1.3.6 Algoritmus nejbližších sousedů

Algoritmus nejbližších sousedů (kNN) pracuje s množinou objektů, mezi kterými vyhledává objekt „nejbližší“ objektu na vstupu. Algoritmus předpokládá, že každá instance odpovídá vektoru v $n$-dimenzionálním prostoru [13]. Vzdálenost objektů je určena metrikou vzdálenosti, např. standardní Euklidovskou vzdáleností

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2},$$

kde $x$ a $y$ jsou vektory v $n$-dimenzionálním prostoru. Algoritmus kNN následně iterativně projde množinu všech instancí a na výstupu vrátí $n$ instancí s nejmenší hodnotou vypočtené vzdálenosti vzhledem k instanci na vstupu.

## 1.4 Klasifikace a prediktivní modelování

V oblasti strojového učení se klasifikací rozumí úloha, kde na základě množiny pozorování zařazaných do určitých tříd je potřeba identifikovat, do které ze tříd patří nové pozorování na vstupu [14]. Jednotlivá pozorování mají určenou množinu vlastností (*features*), které nabývají určitých hodnot a na jejichž základě samotná klasifikace probíhá. Příkladem může být rozhodování, zda nově příchozí e-mail patří mezi spam či nikoliv.

V kontextu zpracování textových dokumentů se klasifikací rozumí zařazení konkrétního textu do příslušné kategorie dané jeho obsahem a dalšími vlastnostmi [15].

Klasifikace se skládá ze dvou hlavních částí: *učení* neboli tvorba klasifikačního modelu, který je schopný klasifikovat data za pomocí testovacích

dat; *klasifikace* neboli proces vybavování modelu, kdy je provedeno určení tříd nových dat.

Predikce je pokročilý proces, který má za cíl projekci vlastností známých pozorování na pozorování nově příchozí. Jedná se především o doplnění neznámých vlastností analyzovaných pozorování. (Klasifikace je tedy predikce s jedním výstupním atributem.) Program tak prochází dokumenty, hledá co nejvíce obecný tvar pravidel, která budou platit i pro nové objekty.

Pro určení úspěšnosti naučených pravidel se využívá způsobu testování, kdy je stranou ponechána část *učících dat* jako *testovací vzorek*, na kterém se provede ověření platnosti navržených pravidel [15].

Dalším druhem predikce je *regrese*, kdy jsou pomocí naučeného modelu dopočítány spojité číselné hodnoty některého atributu [8].

## 1.5 Shlukování a tématická podobnost

V úloze klasifikace dokumentů byly objekty zařazovány do množiny předem známých tříd [15]. Tyto třídy byly vytvořeny na základě předešlé znalosti struktury daného problému. Pokud tato struktura není známá, je třeba zavést postup, jak dokumenty třídit do skupin vzájemně sobě podobných objektů. K této úloze slouží právě shlukování (clustering).

Jelikož třídy nejsou předem známé, tudíž nejsou známé ani jejich vlastnosti, je tyto třídy následně popsat na základě vlastností instancí do nich zařazených.

V oblasti zpracování textů se vstupní texty třídí do kategorií tématicky podobných dokumentů. K tomuto účelu lze využít specializované algoritmy, které modelují tématické celky napříč texty nalezené, např. algorimtus LDA [4], nebo použít standadní techniky pro shlukování objektů.

## 1.6 Strojové zpracování textu

Pro umožnění algoritmického zpracování textů pomocí standardních přístupů strojového učení je potřeba nestrukturovanou textovou reprezentaci převést do podoby strukturovaných, nejlépe číselných, vektorů. K tomuto účelu slouží postupné kroky zpracování textů, ke kterým lze zařadit následující [15]:

- *tokenizace*: proces identfikace tokenů – slov, příp. dalších textových objektů jako datum apod. – ve vstupním textu
- *lemmatizace*: převod slov do jejich základního morfologického tvaru

– *stemming*: vytvoření základního tvaru slova odstraněním předpon, přípon, koncovek

– *vygenerování vektoru*: převod množiny tokenů do vektorové podoby např. vytvořením *bag-of-words* vektorů (každé slovo ze *slovníku* je reprezentováno jednou dimenzí vektoru a hodnota určená např. jednoduchým počtem výskytů slova v textu)

– *transformace vektorů*: úprava hodnot ve vektorech textů, např. vypočtením frekvence místo počtu výskytů, vypočtením tf-idf statistiky jednotlivých slov apod.

  Pro úpravu hodnot vektorů je velice vhodná tf-idf statistika, kterou lze vypočíst podle vzorce [15]

$$tfidf(j) = tf(j) * idf(j)$$
$$idf(j) = \log \frac{N}{df(j)}$$

Dále zle provávět pokročilejší metody hledání důležitých vlastností textu, mezi něž patří

– identifikace *jmenných entit*, neboli objektů v textu, které patří do určitých tříd, např. vlastní jména, jména organizací, názvy míst apod.

– *multiword features*: pracuje s jednotlivými tokeny a hledá takové spojení tokenů, které vytváří určitou zajímavou vazbu

Kapitola 2

# Datové zdroje a akvizice dat

Internet je zdrojem velkého množství nestrukturovaných textových dat. Pro účely novináře je potřeba tato data stahovat, zpracovávat a obohacovat o další dostupné informace tak, aby nově vytvořená struktura umožňovala na data nahlížet z různých nových úhlů pohledu. Vhodné spojení se světem sociálních sítí umožňuje novináři např. sledovat reálnou odezvu publika a tvořit obsah přesně podle zájmů jeho cílové skupiny. Nutnou součástí sledování výsledků vytvořeného obsahu jsou data o čtenosti – zobrazení stránek, návštěvy, čas strávený na stránce apod. Pro sběr těchto dat se využívají analytické nástroje třetích stran, např. nejrozšířenější z nich Google Analytics.

## 2.1 Textová data

Zpravodajské portály, magazíny, blogy a další stránky produkující obsah nabízejí pro strojové čtení obsahu tzv. rss feedy. Soubor ve formátu XML je pravidelně aktualizován a jeho standardizovaný formát umožňuje aplikacím třetích stran zpracovávat (jinak strojově obtížně čitelný) obsah daného portálu. Nejrozšířenější služby postavené nad standardem rss se nazývají rss čtečky. Uživatelé s jejich pomocí sledují vybrané portály a jejich nově vydávané články bez nutnosti ručního procházení jednotlivých webů. Zároveň většina moderních rss čteček umožňuje obsah třídit a filtrovat. Pro uživatele se tak jedná o velké zjednodušení procesu explorace nového obsahu. Rss feedy obsahují především nadpis článku a odkaz na samotný web, dále ve většině případů krátký úryvek nebo perex článku. Jen malé procento rss feedů obsahuje přímo kompletní text článku.

Odkazů jednoduše získaných ze strukturovaných rss feedů lze následně využít k jednoduchému stažení html stránky z cílové url. Výhoda tohoto pří-

stupu spočívá v tom, že odkazy vedou vždy na články, nikoliv obecné sekce webu, nebo stránky se statickým obsahem, úvodní stranu apod. Zároveň není třeba procházet celý web tzv. crawlerem (jak to musí provádět např. vyhledávač Google pro indexaci celého webu), řešit duplicitní, nedostupné, nebo poškozené url.

### 2.1.1 Zpracování HTML souborů

Po stažení cílové html stránky je potřeba vyfiltrovat požadovaný text daného článku. Každý web má velmi odlišnou strukturu zdrojového html kódu, proto není možné data získat pomocí jednoduchých pravidel, nebo zadané cesty ke konkrétnímu elementu apod. K účelu automatické extrakce textu článku slouží postup nazvaný *boiler plate removal* [11].

Nejznámějšími volně dostupnými nástroji pro tento účel jsou *boilerpipe* [12] a *readability* [2]. Jedná se o jednoduše použitelné java, resp. python knihovny.

## 2.2 Příspěvky na sociálních sítích

Sociální sítě umožňují skrze svá API programově přistupovat ke vkládaným příspěvkům. Jedná se o další zdroj (nejen) textových dat. Příspěvky mohou obsahovat odkazy, které lze stahovat a v případě, že se jedná o článek, uložit do databáze článků.

Sociální síť Twitter nabízí dva typy API: REST API a Streaming API. Příspěvky z konkrétního profilu lze pomocí REST API získat snadno pomocí HTTP požadavku:

```
GET https://api.twitter.com/1.1/statuses/user_timeline.json
?screen_name={userId}
```

Výsledkem je json řetězec se strukturou tweetů, ve kterém se již nacházejí vyparsované url adresy jako specální atributy.

Jednoduše tedy lze v pravidelných časových intervalech stahovat příspěvky a aktualizovat lokální databázi.

## 2.3 Sociální signály

Pro sledování sociálních signálů pro jednotlivé články se používá speciálních typů requestů. Url adresa článku slouží jako identifikátor. Pro sociální sítě Facebook, Twitter, Reddit jsou url adresy následující:

```
GET https://api.facebook.com/method/links.getStats?urls={url}
&format=json

GET http://urls.api.twitter.com/1/urls/count.json?url={url}

GET https://buttons.reddit.com/button_info.json?url={url}
```

Každý endpoint vrací různý formát jednoduchého jsonu, v němž se nacházejí hodnoty např. *shares*, *likes*, *comments* pro Facebook.

Pomocí těchto volání lze pravidelně stahovat a ukládat časový průběh hodnot jednotlivých metrik.

## 2.4 Analytické nástroje

Sběr statistik o návštěvnosti stránek téměř bez výjimky zajišťují analytické nástroje třetích stran. Nejrozšířenějším nástrojem je v současnosti zdarma dostupný Google Analytics.

### 2.4.1 Google Analytics

Google Analytics nabízí všem účtům přístup ke statistikám srze API. Dokumentace je dostupná na adrese `https://developers.google.com/analytics/devguides/reporting/core/v3/`. Podstatné je, že vše potřebné lze v API requestu definovat – pro dimenzi url a čas po hodinách přidat metriky *pageviews*, *visits*, *time on site*. API vrací snadno pochopitelný json.

Přístup je autorizován pomocí protokolu oAuth.

### 2.4.2 DAX Comscore

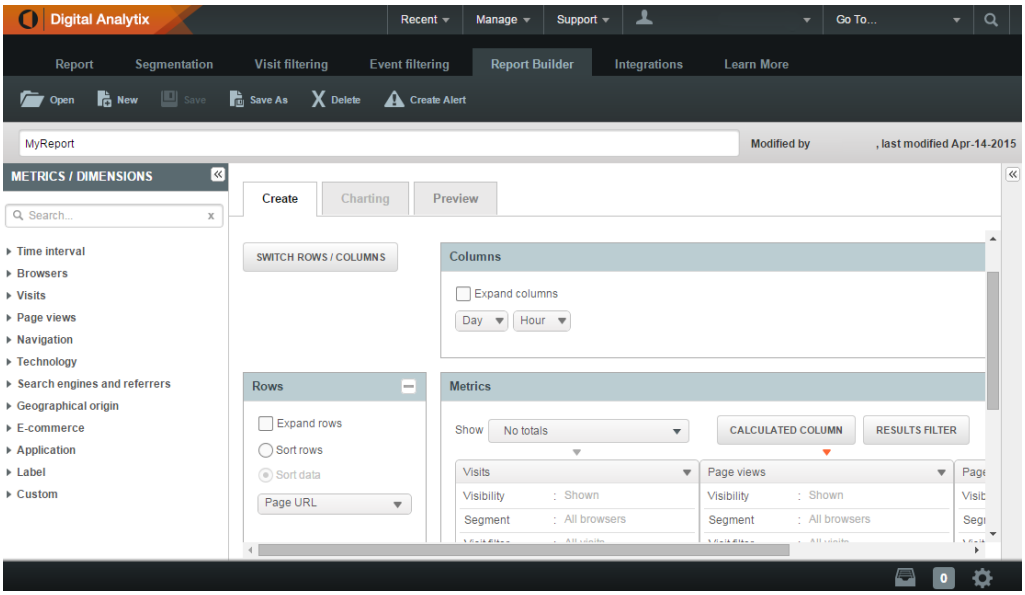
Nástroj od společnosti Comscore je využíván v mnohem menší míře než Google Analytics. Z pohledu datového exportu vyžaduje o řád složitější nastavení.

Nejprve je potřeba ve webovém rozhraní založit nový report. Pro tento report nastavit dimenze a metriky, viz obrázek 2.1. Každý report má vlastní unikátní *reportId*, které se použije v API requestu. Ten má následující formát:

```
GET https://dax-rest.comscore.com/v1/reportitems.xml?itemid=
{reportId}&startdate={from}&enddate={to}&site={siteType}
&eventfilterid={eventFilterId}&format=csv&client={client}
&user={user}&password={password}
```

*eventFilterId* je id uživatelsky definovaného filteru, *siteType* segment, pod kterým jsou data zaznamenána, *client*, *user*, *password* přihlašovací údaje. Výstupem volání je textový soubor se záznamy na řádcích a sloupci oddělenými „|“. Po rozparsování souboru je k dipozici dataset s metrikami namapovanými na url a čas.

API není nijak zapezpečeno, pouze přihlašovacími údaji v url.

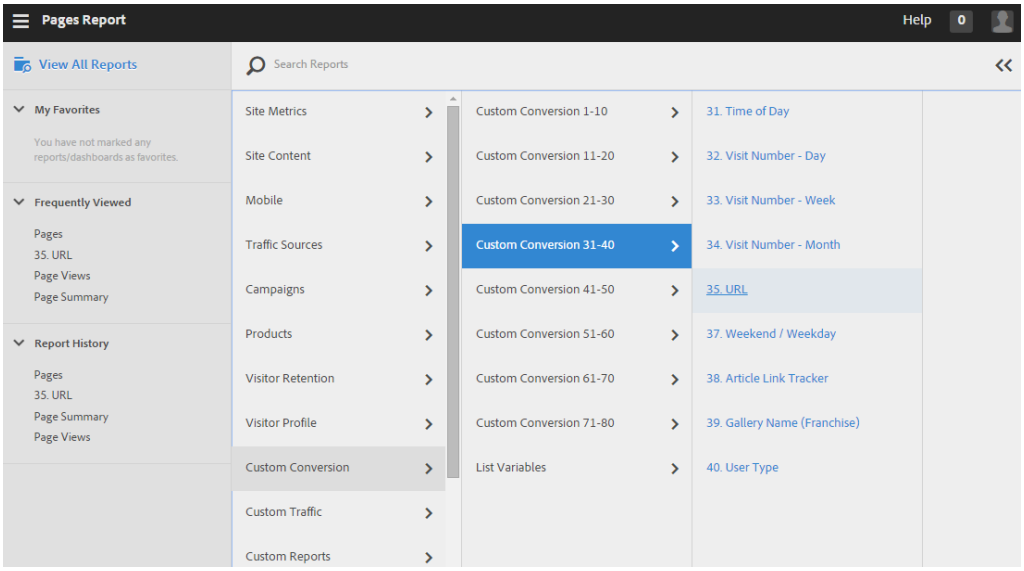



Obrázek 2.1: *Ukázka nastavení DAX Comscore reportu pro export statistik jednotlivých url.*

### 2.4.3 Adobe Omniture

Omniture je analytický systém od společnosti Adobe. Opět je používán ve výrazně menším měřítku, než nejrozšířenější Google Analytics.

Omniture nabízí poměrně dobře zdokumentované API – `https://marketing.adobe.com/developer/documentation`. Pro vytvoření exportu se definuje *reportDescription* s popisem úlohy. Statistiky k jednotlivým url adresám nelze genrovat přímo, Omniture používá uživatelsky definovaná id pro každý sledovaný objekt (např. článek). Report s nastavením *elements* na hodnotu *page* vrátí strukturu, ve které je obsaženo *id* a *url*. Bohužel atribut *url* nemusí být vždy vyplněný. Výhodné je, pokud má Omniture profil nastavené vlastní metriky, které automaticky sleduje, a mezi nimi je i kompletní url, viz obrázek 2.2.




Obrázek 2.2: *Ukázka seznamu vlastních atributů nastavených v rozhraní Adobe Omniture.*

Kapitola 3

# Experimentální část a validace přístupu

Analýza a interpretace struktury vlastního obsahu, vytvořeného daným editorem či vydavatelstvím, je prvním krokem ve strojovém zpracování textů a algoritmickém popsání v datech nalezených souvislostí. Jedná se o fázi prováděnou před započetím sledování obsahu z vnějších zdrojů – internetové zpravodajské portály, magazíny, blogy, tiskové zprávy apod.

Data získaná pomocí metod a postupů z kapitoly 2 musí projít určitými fázemi předzpracování. Jedná se především o parsování textů ze stažených html stránek, převedení textů do algoritmicky zpracovatelného formátu, transformace textové reprezentace (stemming, lemmatizace, filtrace apod.), propojení statistických dat z analytických systémů (počet zhlédnutí stránky, doba návštěvy atd.). Výsledek zpracování je následně potřeba vhodným způsobem zobrazit a nejlépe přehledně graficky vizualizovat.

V následujících kapitolách jsou uvedeny postupy pro zpracování textových dat, generování pokročilých příznaků, připojení získaných statistických dat a výslednou vizualizaci výstupu.

## 3.1 Zpracování textů

Algoritmy strojového učení vyžadují specifický formát vstupních dat – využívá se reprezentace objektů pomocí příznakových vektorů. Cílem zpracování textů je tento vektor připravit tak, aby co nejlepším možným způsobem popisoval obsah sdělení v textu.

**Tokenizace**: V první fázi zpracování textu se provádí *tokenizace* – rozdělení bloku textu na *tokeny*, tedy slova nebo struktury jako datum, tele-

fonní číslo apod. Tokenizaci lze provádět buď jednoduchým segregováním slov pomocí mezer, nebo lépe použít chytrý tokenizační algoritmus. Pro české texty se nabízí např. nástroj *czechtok*, anglické texty lze tokenizovat např. pomocí nástroje *Stanford Tokenizer* z knihovny *Stanford NLP*. Tyto nástroje zároveň umožňují identifikovat větší struktury typu věta a odstavec. Rozdělení textu na věty se nazývá *segmentace*.

**Bag of words**: Nejpřímější způsob reprezentace jednotlivých článku je sestavení *příznakového vektoru* o délce počtu slov uložených ve *slovníku* (počet unikátních slov v textu) a vyplnění hodnot počtu výskytu každého slova na odpovídající index. Tímto způsobem vytvořené vektory mají nevýhodu především ve velké redundanci informace, pokud je v textu stejné slovo použito v různých tvarech – typicky např. jednotné a množné číslo. Dále tento způsob reprezentace přiřazuje každému slovu stejnou váhu důležitosti, což pro účely dalšího zpracování není příliš vhodný postup.

**Stemming a lemmatizace**: Pomocí stemmingu či lemmatizace se jednotlivá slova převádějí do základních tvarů. *Stemming* se snaží nalézt kořen slova oddělováním předpon, přípon a koncovek. *Lemmatizace* převádí slovo do základního morfologického tvaru – pro podstatná jména se jedná o první pád čísla jednotného, u sloves o infinitiv atd. Tímto způsobem dochází ke značné redukci počtu slov ve slovníku a zároveň k lepší deskripci textu jako takového.

**Filtrace slov**: Slovník vytvořený z prostého textu obsahuje velké množství slov, která pro další analýzy nemají téměř žádný význam, nebo výstupy přímo znehodnocují. Jedná se o slova typu spojky, citoslovce, často se vyskytující slovesa apod. Jednoduchou metodou pro jejich odstranění je použití slovníku *stop slov* – slova obsažená ve slovníku se při zpracování ignorují. Tímto způsobem se dále značně redukuje dimenze příznakových vektorů. Další pokročilou možností filtrace je využití morfologických kategorií slov, tedy přiřazení příznaků typu podstatné jméno, přídavné jméno, sloveso, spojka atd. Existují nástroje, které s jistou mírou přesnosti dokáží automaticky text analyzovat a morfologické kategorie přiřadit (*part-of-speech tagging*). Pro české texty se jedná např. o morfologické analyzátory *morfo*, *morphodita*, *ajka*, *majka*. Anglický text lze zpracovat např. pomocí *POS Taggeru* z knihovny *Stanford NLP*. Filtrační pravidla mohou být následně nastavena např. na ponechání podstatných a přídavných jmen a sloves. Opět dochází ke značné redukci dimenze příznakových vektorů článků.

**Tf-idf reprezentace**: Každému článku lze přiřadit ideálně velmi malou množinu slov, která z větší části popisuje jeho obsah. Slova v textu mají různou váhu důležitosti a relevance, kterou lze algoritmicky vyjádřit např. pomocí hodnoty *tf-idf* statistiky. (viz teorie) Příznakový vektor pak může obsahovat váhy slov namísto konstantní hodnoty násobené počtem výskytů.

Slova seřazená sestupně podle hodnoty *tf-idf* mají nejdůležitější postavení v daném článku vzhledem k celé databázi textů nebo její vybrané relevantní části.

**Kosínová podobnost**: Podobnost dvou textů lze definovat a vyčíslit mnoha způsoby. Vzdálenost mezi příznakovými vektory popisuje např. kosínová vzdálenost. Vyčíslením vzdálenosti jednoho textu vůči všem ostatním v databázi lze jednoduše nalézt tématicky nejpodobnější články.

## 3.2 Stažení textových dat

Obecnému stahování článků se věnuje kapitola 2.1. Získání textů konkrétního portálu může být provedeno několika cestami:

- *export dat z redakčního systému*: z pohledu zpracování dat nejjednodušší varianta, která zároveň nabízí nejlepší možnou kvalitu textových dat.
- *stažení HTML stránek dle dat z analytického systému*: ze systému pro sledování návštěvnosti webových stránek se vyexportují URL adresy jednotlivých článků. Následně je potřeba stáhnout veškeré HTML stránky a vyfiltrovat z nich texty článků. Tato metoda přináší výhodu v seznamu konkrétních URL, zároveň je možné s mírnou chybou zjistit data vydání článků. Často je ale problém udržet rozumnou kvalitu získaných dat – filtrace textů, nadpisů, autora z HTML kódu není oproti přímému exportu tolik spolehlivá.
- *doménově omezené spuštění crawleru*: na konkrétním portálu je spuštěn crawler, který skrze prolinkování stránek projde celý web a uloží veškeré dostupné HTML stránky. Kvalita dat závisí na struktuře odkazového profilu daného webu a způsobu implementace prolinkování jednotlivých stránek. Některé redakční systémy např. nezobrazují odkazy na starší články (přitom příslušná url článku je stále dostupná), nebo neumožňují listovat větším množstvím článků. Crawler tak nemusí být schopen získat data za delší historické období.
- *postupné stažení článků z rss*: zahrnuje stahování pouze nových dat bez importu historických článků.

Export dat z redakčního systému je nejlepší možnou variantou stažení historických dat. Ne vždy je však možné exporty zpřístupnit, nebo vůbec provést. Pak své uplatnění nacházejí další postupy, které obecně dokáží data získat.

## 3.3 Statistická data

Pro analýzu chování a preferencí čtenářů je nezbytné propojit textová data s daty o návštěvnosti. Ty je možné získat skrze používaný analytický nástroj, např. *Google Analytics*. Jednorázové stažení dat lze provést přes ruční export. Pro pravidelnou aktualizaci je nutné využít dostupných API a programově data stahovat. Spojení statistických a textových dat se provádí přes url adresy jednotlivých článků. Každá url, pokud byla stránka navštívena, má záznam v analytickém systému s údaji o počtu *zobrazení stránek* (*pageviews*), *návštěv* (*visits*), *času stráveném na stránce* (*time on site*) apod. Tyto údaje se dají využít ke zjištění zajímavých závislostí uvnitř dat a společně s textovou analýzou vytvořit report takové zavislosti popisující, nebo je transformovat do podoby grafické vizualizace, viz následující kapitola.

## 3.4 Vizualizace

Vytváření grafické vizualizace textových dat řeší problém převedení obecně vysoce dimenzionálního prostoru textových (např. bag of words) vektorů do prostoru o dvou dimenzích. Pro účely vykreslení struktury obsahu konkrétního zpravodajského portálu je možné využít předpočítaných vektorů jednotlivých textů a jejich vzájemných podobností – vhodný postup pro připravení vektorů je uveden v kapitole 3.1. Vizualizace je následně vytvořena formou orientovaného grafu, kde uzly reprezentují články a hrany hodnotu vyčíslené podobnosti článků.

Samotné vytvoření orientovaného grafu pouze připravuje strukturu, kterou je potřeba pomocí dalších technik a nástrojů vykreslit do podoby smysluplné a prakticky přínosné vizualizace. K tomuto účelu je možné využít např. program *Gephi* [3] (freeware), který slouží právě k interaktivní vizualizaci a exploraci různých druhů sítí, grafů, hierarchických grafů apod. Umožňuje pomocí různých typů *layoutovacích* algoritmů upravovat rozložení uzlů grafu v rovině tak, aby vynikly důležité spojitosti v datech obsažené. Dále program nabízí možnosti obarvení uzlů a hran a nastavení velikosti uzlů a hran, což umožňuje zanášet do vizualizace další dimenze různých metrik.

Gephi požaduje dva datové importy:

- *seznam uzlů*: s předdefinovanými atributy *id*, *label* a možností libovolného počtu uživatelsky definovaných atributů.
- *seznam hran*: každá hrana je určena zdrojovým uzlem (atribut *source*) a cílovým uzlem (atribut *destination*) zadaných pomocí *id* daných uzlů. Dále jsou předdefinovány atributy *label*, *weight* (implicitně je

používají layoutovací a další algoritmy) a možnost libovolného počtu uživatelsky definovaných atributů.

Data je možné jednoduše importovat pomocí dvou csv souborů.

Výsledné rozložení grafu v prostoru určuje volba a nastavení parametrů *layoutovacího algoritmu*. Gephi nabízí velkou paletu různých algoritmů, kde každý je vhodný pro určitý typ grafu a cíl, ke kterému je vizualizace určena. Z předpřipravených algoritmů se, dle empirického testování, pro textová data zpravodajského portálu zdá být nejvhodnější algoritmus *Force Atlas 2* [9]. Dostupné *Gephi pluginy* umožňují přidávat další layoutovací algoritmy. Po provedení různých testů se velmi vydařeným algoritmem ukazuje plugin *OpenOrd Layout*[1].

Dynamické obarvení uzlů resp. hran a nastavení velikosti uzlů resp. tloušťky hran podle hodnot importovaných atributů přináší možnost ve vizualizaci předávat informace např. z analytických systémů. Jedním ze základních atributů je *počet zobrazení* konkrétního článku. Uzly s velikostí danou *počtem zobrazení* tak na první pohled zvýrazní nejpopulárnější články a také umožní sledovat distribuci *pageviews* přes celou databázi. Tloušťka hran pak může indikovat např. hodnotu podobnosti článků a tím prezentovat tématickou souvislost mezi danou dvojicí článků.

Gephi dále nabízí export vytvořeného grafu v podobě HTML stránky, což ulehčuje následné sdílení, případně prezentaci dosažených výsledků.

## 3.5 Případová studie 1

První z analyzovaných portálů je americký zpravodajský server gokicker.com. GoKicker poskytl XML export svých historických textových dat a přístup do analytického systému *Google Analytics*. Proběhlo zpracování textů podle výše popsaného postupu, stažení statistik ze systému *Google Analytics* a následně import připraveného datasetu do programu Gephi.

Celkově export obsahuje 1292 článků. Každý článek má následující atributy: datum vydání, titulek, text, tématická kategorie. Zastoupení článků z jednotlivých tématických kategorií není rovnoměrné, kategorie jsou navíc typu lokální/světové události, lze tedy předpokládat mísení uzlů a tvorbu lokálních shluků podle témat na nižších úrovních.

Výstup shlukování a vyznačení tématických kategorií je ilustrován obrázek 3.1. Layoutovací algoritmus *Force Altas 2* v programu Gephi dokáže v rovině separovat příslušné shluky článků. Pro účely zpřehlednění vizua-

[1]Dostupný z: `https://marketplace.gephi.org/plugin/openord-layout/`

lizace byly odstraněny hrany s váhou kosínové podobnosti menší než 0.1 (empiricky nalezená hodnota).

Obrázky 3.2 a 3.3 porovnávají články z hlediska jejich dlouhodobého přínosu na návštěvnost serveru. Po vyznačení článků, které mají vysokou hodnotu návštěv i dlouho po svém vydání, je možné nalézt témata, na která by bylo vhodné se při tvorbě nového obsahu zaměřit.

Obecně lze na vizualizace nahlížet jako na ukazatel struktury, vazeb a např. popularity jak jednotlivých článků, tak tématických celků, které lze z vizualizace jednoduše vyčíst. Možností, jak vizualizace dále modifikovat, je nepřeberné množství a je potřeba určit konkrétní cíl, za kterým je každá vizualizace připravena.

## 3.6 Případová studie 2

Druhý analyzovaný portál je americký server thedailybeast.com. DailyBeast poskytl datový export počtu zobrazení článků. Tento export obsahuje *url adresu* článku a statistiky *pageviews*, *visits*, *time on site*. Bylo zapotřebí z jednotlivých url adres stáhnout nadpis a text článku, což se provedlo pomocí jednoduchého, ručně připraveného skriptu.

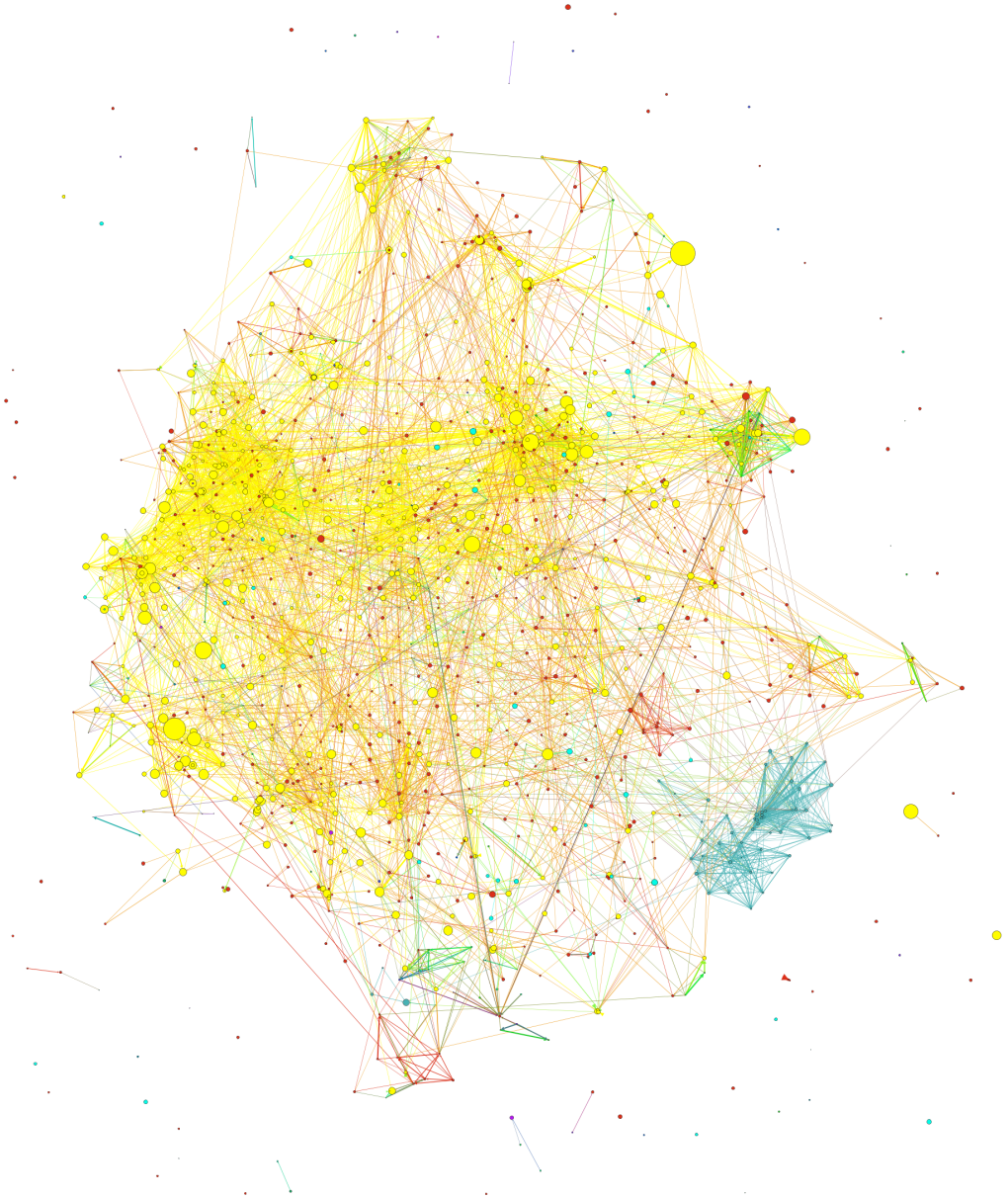
Celkově export obsahuje 7518 článků, které mají pouze výše zmíněné atributy. Předzpracování textů a příprava datové matice vektorů článků proběhly stejným způsobem jako u serveru gokicker.com.

Obrázek 3.4 zobrazuje obecný pohled na data podle počtu pageviews článků. Využit zde byl layoutovací algoritmus *OpenOrd* a pro vizuální zpřehlednění odstraněny hrany s hodnotou míry kosínové podobnosti menší než 0.15 (empiricky nalezená hodnota).

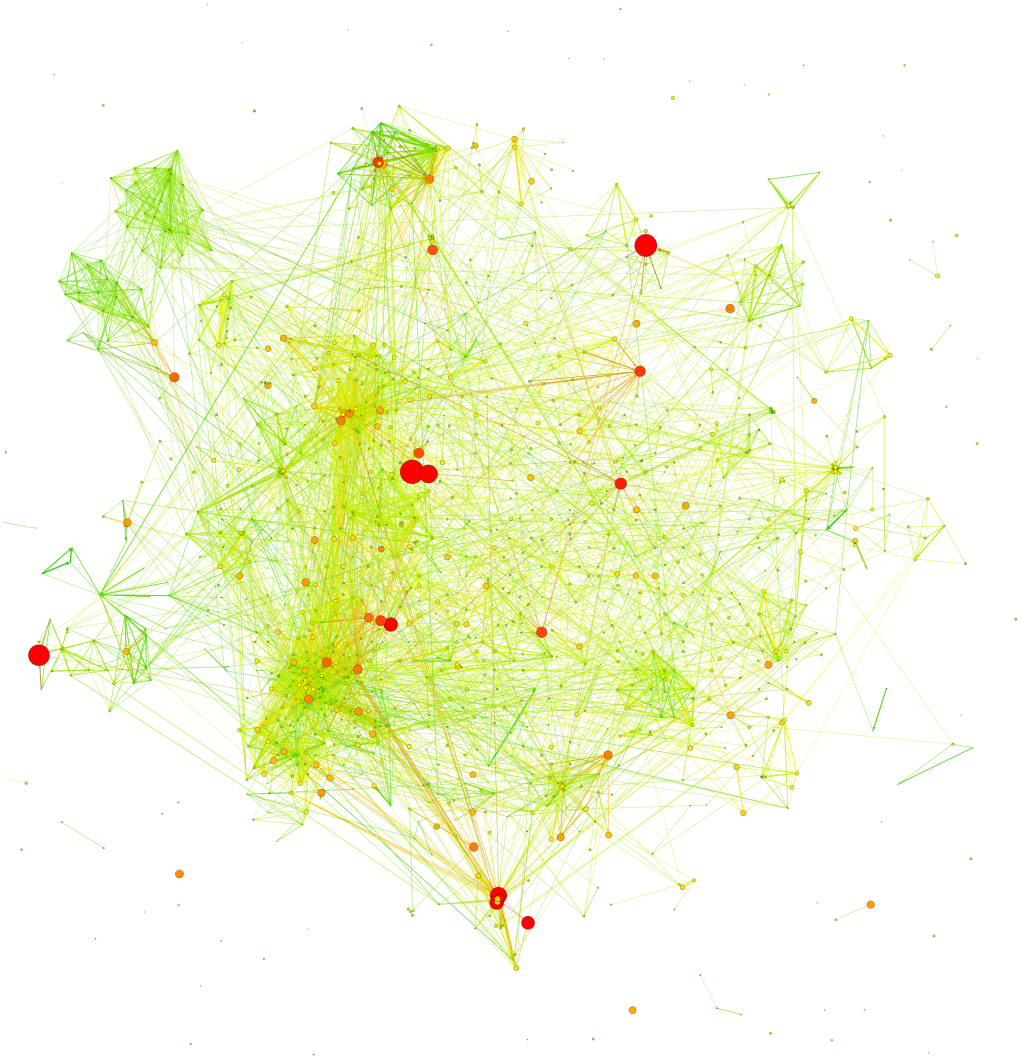
Zásadní krok ve zpracování textových dat je automatický clustering grafu. Obrázek 3.5 ukazuje výsledek po spuštění algoritmu Chinese whispers na předpřipravenou grafovou reprezentaci článků. Algoritmus je schopný nalézt smysluplné clustery a označit příslušné uzly. Krokem, který může následovat, je „kondenzace“ všech uzlů clusteru do jediného uzlu a analýza nově vzniklé struktury a jejích vlastností.

Obrázek 3.7 ukazuje možnost zobrazení dalších vypočtených atributů článků. Předzpracované texty byly analyzovány pomocí slovníkové metody – slovník (množina slov) definuje pojmy z určité oblasti, např. právo, politika, cestování, nebo slova, která znamenají příklon k určitým vlastnostem textu, např. senzace, dojemnost, zmatení, agresivnost. Pomocí slovníku jsou vypočteny atributy dané počtem výskytů slov ze slovníku v textu jednotlivých článků. Pro další analýzy a hledání závislostí mohou být takové příznaky velmi přínosné. Konkrétně na obrázku 3.7 je provedena základní

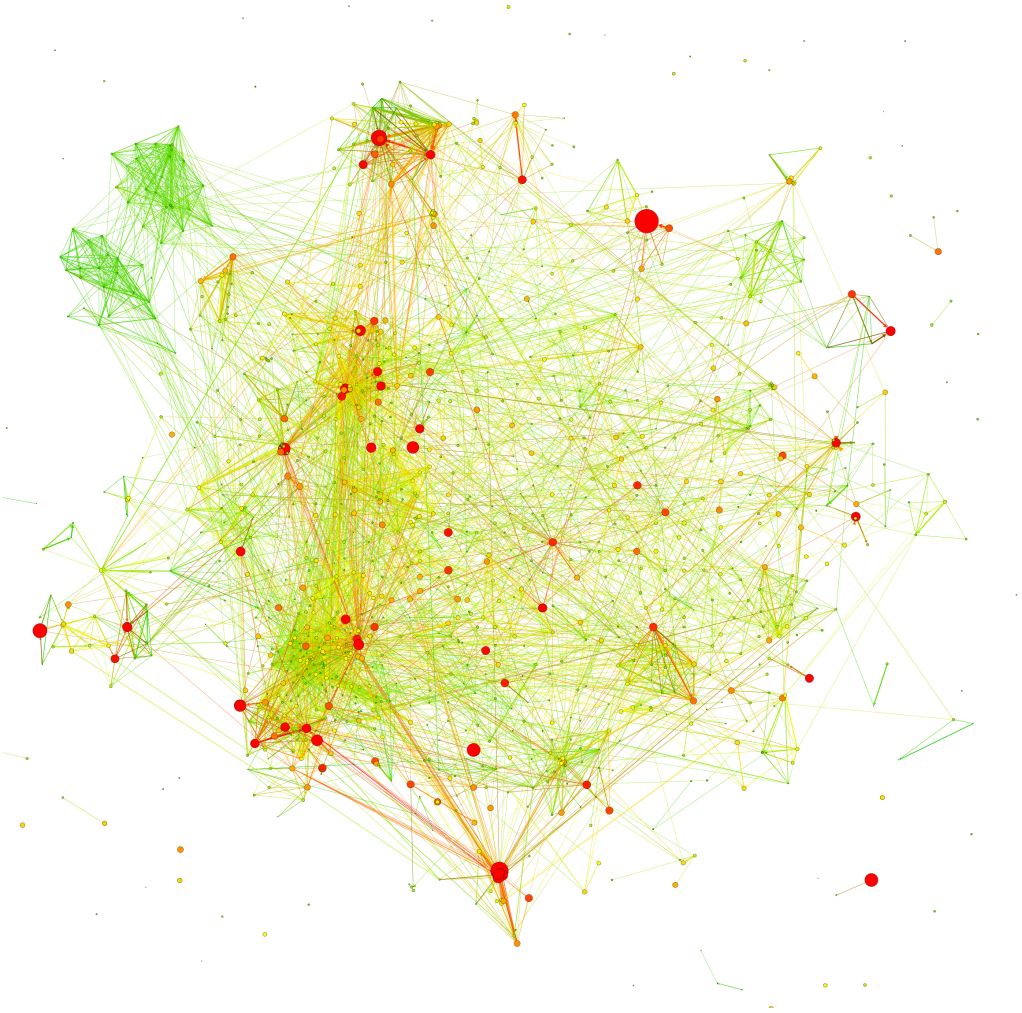
analýza sentimentu – určení pozitivního resp. negativního ladění textu – daná slovníkem pozitivních resp. negativních slov. Zajímavé je pozorovat zda sentiment pozitivně či negativně ovlivňuje počet zhlédnutí stránky a tím do jisté míry určit, o co mají zájem čtenáři daného portálu.



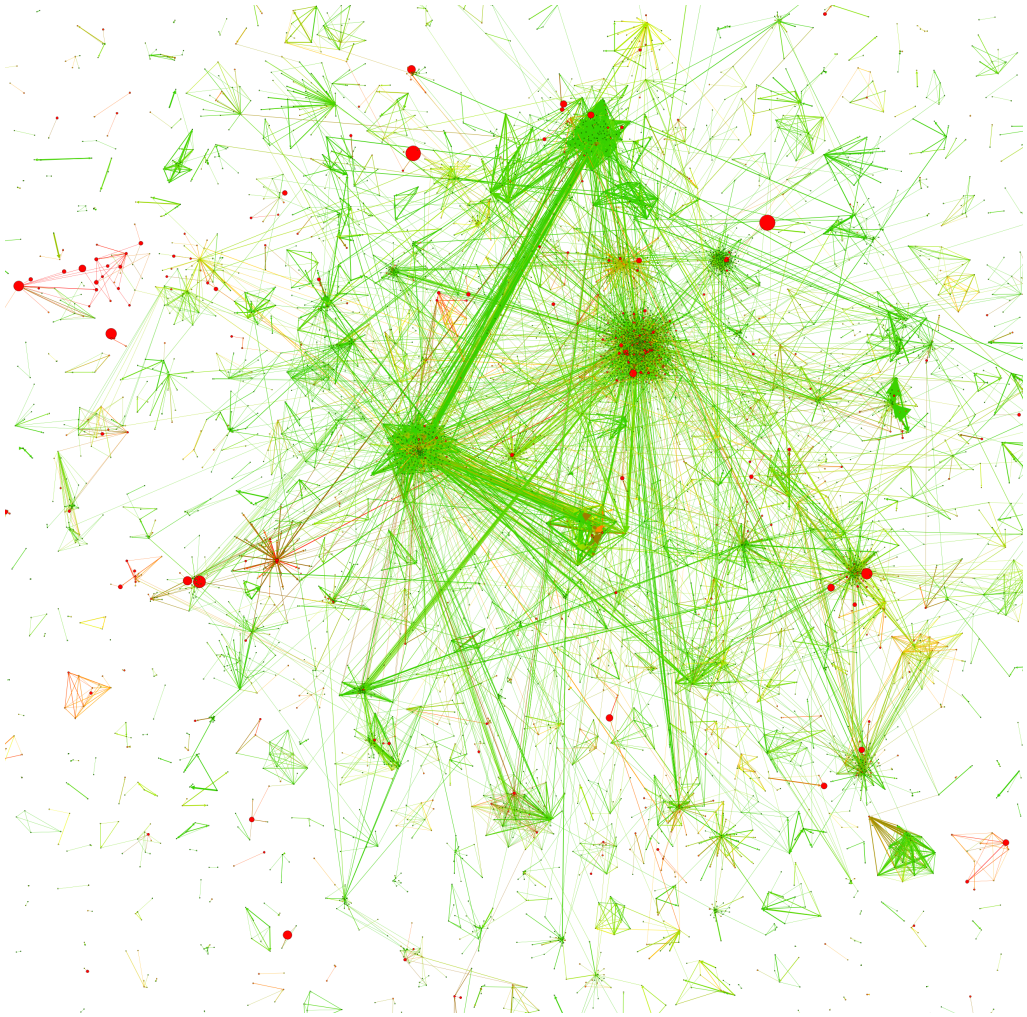
Obrázek 3.1: *Vizualizace článků se zvýrazněním tématických kategorií serveru gokicker.com. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu zobrazuje počet zobrazení článku a obarvení uzlu příslušnost ke konkrétní kategorii. Graf je layoutován pomocí algoritmu Force Atlas 2. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.1. Zastoupení jednotlivých kategorií není rovnoměrné, ale lze pozorovat shluknutí uzlů určitých kategorií – především modré uzly s tématem celebrit.*



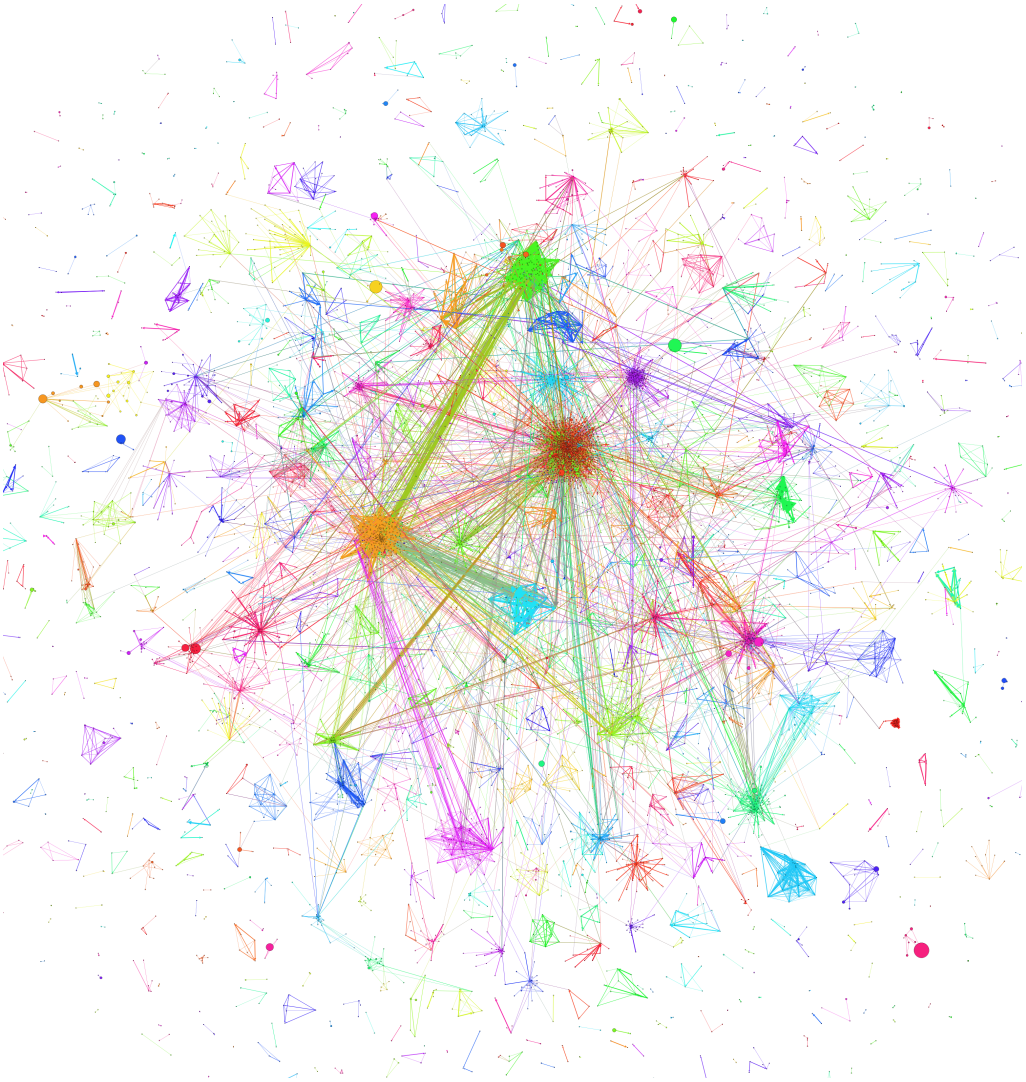
Obrázek 3.2: *Vizualizace článků se serveru gokicker.com s vyznačením počtu zobrazení krátce po vydání. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu a obarvení směrem k červené škále zobrazuje počet zobrazení článků, které se udály během prvního týdne po vydání. Barva hrany respektuje barvu uzlu. Graf je layoutován pomocí algoritmu OpenOrd Layout. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.1. Lze pozorovat poměrně dobře separované tématické shluky. Článků, které měly hodnotu počtu zhlédnutí během prvního týdne výrazně vyšší, než činí průměr, není mnoho. Zajímavé je porovnání s počtem dlouhodobých zhlédnutí, viz obrázek 3.3.*



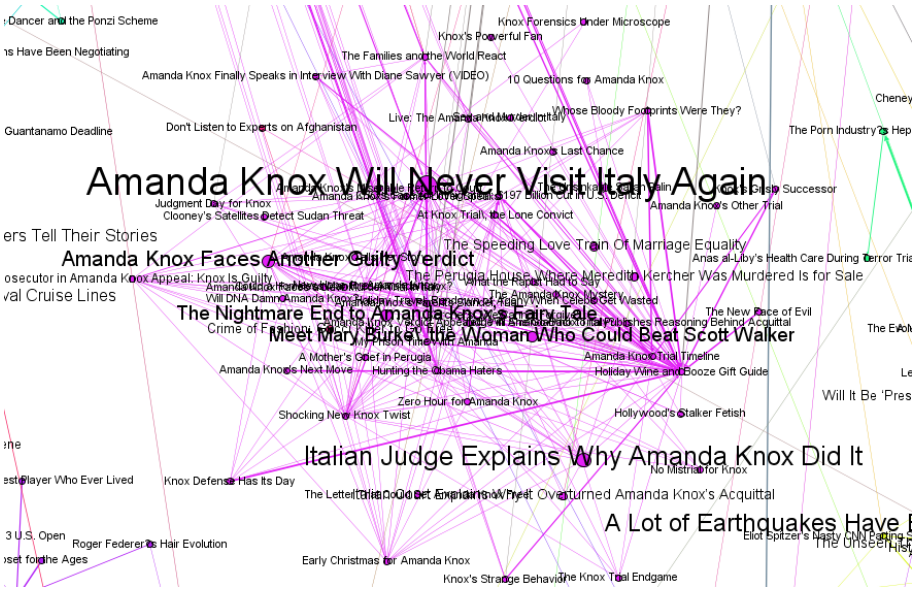
Obrázek 3.3: *Vizualizace článků se serveru gokicker.com s vyznačením počtu zobrazení dlouhodobě po vydání. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu a obarvení směrem k červené škále zobrazuje počet zobrazení článků, které se udály po době delší než týden od vydání článku. Barva hrany respektuje barvu uzlu. Graf je layoutován pomocí algoritmu OpenOrd Layout. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.1. Lze pozorovat poměrně dobře separované tématické shluky. Při porovnání s obrázkem 3.2 lze vypozorovat, že dlouhodobě navštěvované články se liší od článků s vysokým počtem zobrazení za první týden. Zároveň jsou tyto články často součástí nebo přímo hlavním propojením výraznějších clusterů.*



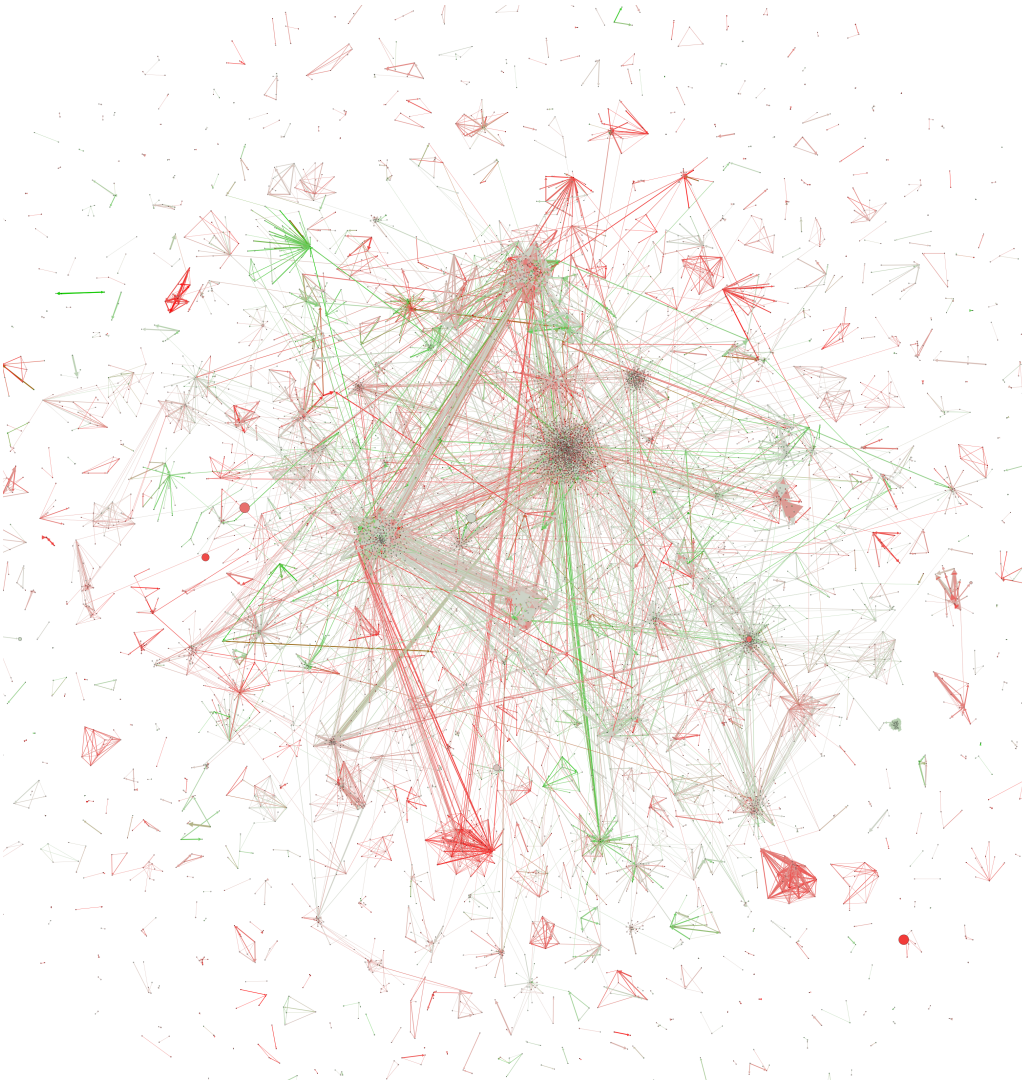
Obrázek 3.4: *Vizualizace článků se serveru dailybeast.com s vyznačením počtu zobrazení. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu a obarvení směrem k červené škále zobrazuje počet zobrazení článku. Barva hrany respektuje barvu uzlu. Graf je layoutován pomocí algoritmu OpenOrd Layout. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.15. Lze pozorovat výrazně separované tématické shluky a nalézt témata, která přinášejí značně více pageviews oproti průměru.*



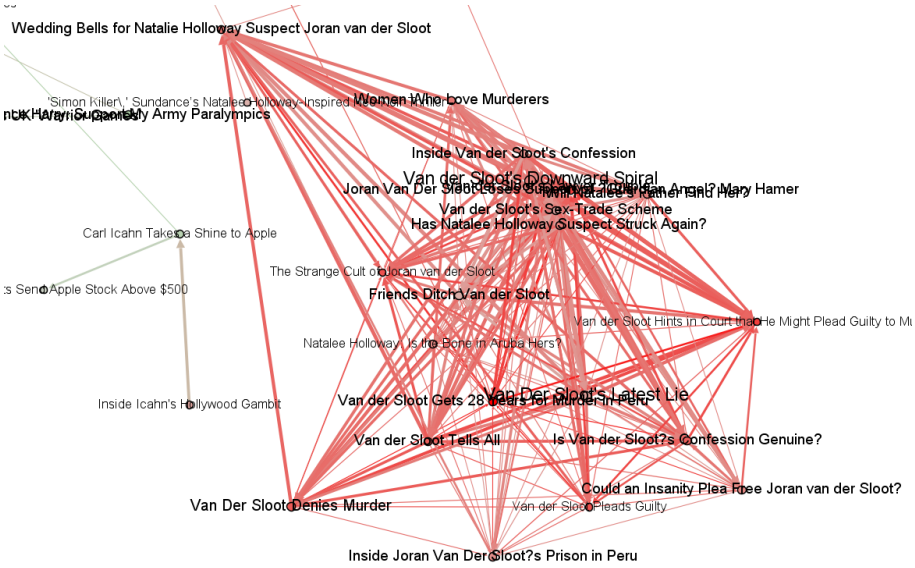
Obrázek 3.5: *Vizualizace článků se serveru dailybeast.com po provedení clustrování pomocí algoritmu Chinese whispers. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu zobrazuje počet zobrazení článku. Graf je layoutován pomocí algoritmu OpenOrd Layout. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.15. Pro strojovou detekci tématických clustrů byl použit grafový algoritmus Chinese whispers a vizualizace ukazuje, že navržený postup pro provedení tématického clustrování dosahuje velmi uspokojivých výsledků. Detail jednoho z clustrů je ukazuje obrázek 3.5.*



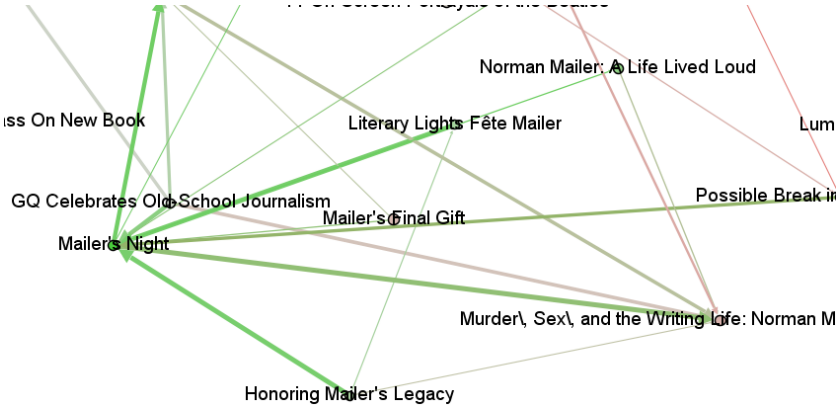

Obrázek 3.6: *Detail shluku článků rezebírající téma okolo Amandy Knox. Cluster byl identifikován pomocí algoritmu Chinese whispers na předpřipravené grafové reprezentaci textů.*



Obrázek 3.7: *Vizualizace článků se serveru dailybeast.com po provedení určení sentimentu – pozitivní/negativní ladění textu článku. Uzel reprezentuje článek, hrana obsahovou podobnost spojených článků. Velikost uzlu zobrazuje počet zobrazení článku. Graf je layoutován pomocí algoritmu OpenOrd Layout. Hrany jsou zobrazeny pouze, pokud míra kosínové podobnosti textů článků přesáhne hodnotu 0.15. Sentiment byl určen pomocí slovníkové metody a je zobrazen barvou uzlu – červené barva negativní, resp. zelená pozitivní sentiment. Lze pozorovat jak shluky negativních tak pozitivních témat.*




Obrázek 3.8: *Shluk článků s nalezeným negativním sentimentem. Sentiment byl spočten na základě slovníku slov považovaných za negativní. Hodnota sentimentu je naznačena barvou hran – červená znamená negativní hodnotu. Přečtením titulků článků lze zjistit, že se jedná o téma vyšetřování vraždy.*




Obrázek 3.9: *Shluk článků s nalezeným pozitivním sentimentem. Sentiment byl spočten na základě slovníku slov považovaných za pozitivní. Hodnota sentimentu je naznačena barvou hran – zelená znamená pozitivní hodnotu. Přečtením titulků článků lze zjistit, že se jedná o téma okolo spisovatele Normana Mailera.*

Kapitola 4

# Návrh nástroje pro žurnalistiku nové generace

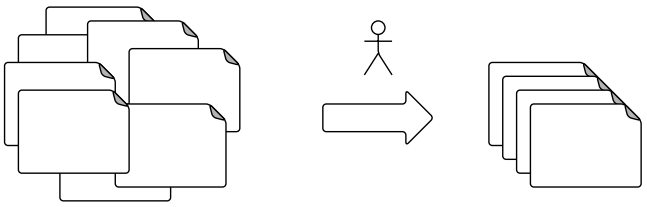
Následující kapitola se věnuje použití přístupů strojového učení v oblasti žurnalistiky. Navazuje na analýzu dat z vlastního webového portálu, na jejímž základě představuje postupy, jak vyhledávat a identifikovat obsah vhodný pro specifické cílové publikum a to zároveň s ohledem na osobní preference jednotlivých autorů. Dále navrhuje přístup k meření výsledků dosažených na základě použití zmíněných metod.

## 4.1 Doporučování obsahu

Každý autor obsahu sleduje mnoho zdrojů, ze kterých čerpá informace pro tvorbu nových článků. Obrovské množství denně publikovaných zpráv s sebou ale přináší hrozbu potenciálního informačního přetížení. Vhodným nasazením systému pro *doporučování* obsahu je možné filtrovat nebo alespoň seřadit příchozí informace podle zajímavosti pro konkrétního autora.

Propojení s inteligentním rekomendačním systémem umožňuje tento proces automatizovat a zároveň poskytnout dostatečnou kvalitu výstupů doporučení. Systém na základě uživatelských interakcí přizpůsobuje řazení obsahu zjištěným preferencím uživatele. Se zvyšujícím se počtem zaznamenaných interakcí je možné doporučení provádět na stále lepší a lepší úrovni.

Využívá se pouze *filtrování na základě obsahu*, protože předpoklad velkého počtu autorů s podobnými preferencemi je prozatím nereálný. Systém pro rekomendaci používá nástroj, který vyvinul Ing. Tomáš Řehořek. Vybavení doporučení probíhá pomocí použití algoritmu *kNN* na datech online aktualizovaných touto aplikací.



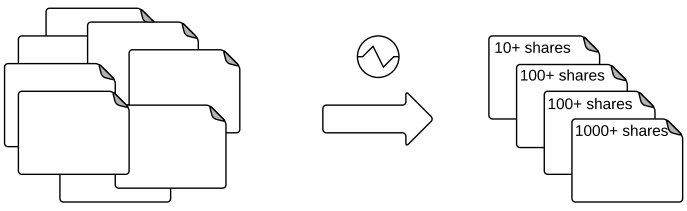
Obrázek 4.1: *Rekomendační systém řadí a filtruje vstupní dokumenty na základě interakcí konkrétního uživatele.*

## 4.2 Predikce popularity

Atributy a signály přiřazované ke článkům stahovaným z internetových zdrojů (viz kapitola 2) indikují současnou popularitu např. na sociálních sítích. Těchto dat je možné využít k vytvoření *predikčního modelu*, který je schopný predikovat nějakým způsobem definovanou *cílovou proměnnou*.

Z historických dat o popularitě různých témat lze sestavit *dataset* obsahující vektory s průběhem vývoje popularity a finální popularitou jako cílovou proměnnou. Takovým způsobem se definují vstupy a výstup predikčního modelu, který se *naučí* tuto cílovou proměnnou predikovat a tím předpovídat budoucí popularitu konkrétního článku.

Přidáním dalších atributů článku, jako např. titulek, text, autor, zdroj, mezi vstupy predikčního modelu se navíc může model naučit predikovat popularitu článku na základě samotného obsahu a tím předpovídat úspěch článku ještě před tím, než takovou informaci ukáží sociální či jiné signály.




Obrázek 4.2: *Predikce popularity předpovídá např. úspěch na sociální síti Facebook vyjádřený pomocí počtu sdílení (shares).*

Predikci lze provádět s cílovou proměnnou dvojího druhu: obecná popularita získaná pomocí dat ze sociálních sítí; zajímavost článku pro čtenáře konkrétního portálu – měřeno např. počtem zobrazení. Druhá možnost se řídí přímo preferencemi návštěvníků konkrétního portálu a umožňuje tím vyhledávat přesně ty dokumenty, které mají největší potenciál být zajímavé pro vlastní čtenáře. Oba přístupy je však vhodné kombinovat – obecná popularita vyhledává témata, o kterých zatím nemuselo být na vlastním serveru nic napsáno, a historicky populární témata jsou upřednostněna díky druhé predikci.

Systém pro predikci využívá nástroj, který vyvinul Ing. Jan Černý [16]. Ten používá evoluční algoritmy pro vyšlechtění kombinace modelů a nastavení jejich parametrů přesně podle charakteru vstupních dat. Výsledný model je tak *ensamble* [8, 1, 5] více druhů modelů např. lineární model, polynomiální model, sigmoidní model, s různým nastavením jejich parametrů.

## 4.3 Shlukování

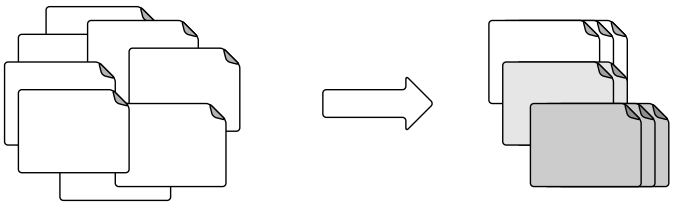
Z hlediska praktické použitelnosti systému je nezbytné, aby články, které se věnují jednomu tématickému celku, byly upořádané do jedné množiny, zastupující konkrétní téma jako celek. Je zapotřebí editorům nabídnout náhled na statistiky témat a aktuálního světového dění tak, aby na základě shromážděných dat mohli nalézt téma se zajímavým potenciálem a rychle reagovat např. sepsáním článku a informováním vlastních čtenářů.

Inteligentní sledování světového dění by zároveň svou analýzou a doporučeními mělo pokrývat právě takovou množinu témat, která pro čtenáře konkrétního vydavatelství bude dostatečně zajímavá. Metody pro analýzu vlastního publika byly uvedeny v kapitole 3, kde bylo zároveň provedeno shlukování článků do tématických celků. Postup, použitý pro zpracování staticky exportovaného obsahu jednoho portálu, nicméně není příliš dobře použitelný pro dynamicky přibývající obsah ze světových zdrojů. Postup shlukování článků je tak potřeba vhodným způsobem modifikovat.

Řešení popsané v kapitole 3.4 – shlukování grafové reprezentace článků pomocí algoritmu Chinese whispers – není vhodné především z důvodu, potřeby neustálého přepočtu kosínové podobnosti vektorů jednotlivých článků a slabé možnosti ovlivnit způsob shlukování např. nastavením různých parametrů algoritmu. Možné je pouze upravit grafovou reprezentaci dat na vstupu, např. provést nelineární transformaci vyčíslených podobností, ignorovat hrany s váhou menší než zvolený práh apod.

Použité řešení shlukování spočívá v extrakci jmenných entit z titulků a textů a jejich indexaci na konkrétní články. Algoritmus shlukování následně

na vstupu dostává id konkrétního článku a na základě výskytu jmenných entit vrací články tématicky podobné vstupnímu. Základním předpokladem pro korektní fungování tohoto postupu je fakt, že systém reálně pracuje pouze s daty z posledních 24 hodin. Nepředpokládá se tedy, že by daná kombinace jmenných entit vytvářela různé tématické celky. Proces zclustrování článků do témat poté funguje na principu postupného procházení množiny článků a vyřazování těch, které se již objevily v nějakém z tématických shluků.



Obrázek 4.3: *Shlukování identifikuje tématicky podobné články a zařadí je do jedné množiny.*

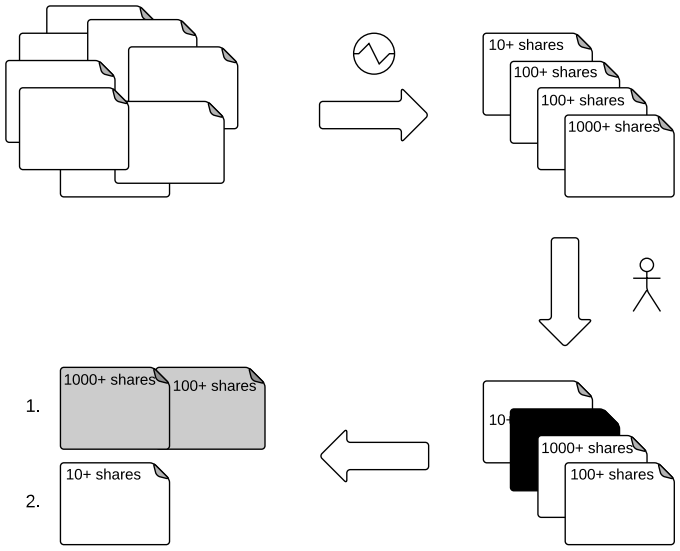
## 4.4 Spojení doporučování a predikce

Doporučení témat konkrétním autorům se může provádět pouze na základě atributů dostupných článků (titulek, text, zdroj atd.). Takové doporučení přesně respektuje osobní preference autora. Nicméně pro doporučování, které má plnit funkci předpovídání popularity témat, je nutné výslednou množinu doporučených témat přeskupit právě podle predikované popularity. V použitém rekomendačním nástroji k tomuto účelu slouží funkce *boosting*, která podle zadaného vzorce může upravit pořadí výstupních objektů – článků.

Do boostingu v tomto konkrétním případě vstupuje predikovaná popularita na sociálních sítích a také popularita predikovaná pro web autora. Hodnota vypočtené podobnosti objektů v rámci algoritmu kNN se vždy přenásobuje nastaveným *boosterem* a tím se může měnit výsledné pořadí objektů na výstupu.

## 4.5 Systém pro doporučování (v budoucnu) populárních témat

Výsledný systém doporučení kombinuje přístupy zmíněné v předchozích kapitolách a spojuje je do jednoho funkčního celku. Aplikace provádí predikci popularity stahovaných článků, kterou následně ovlivňuje doporučování konkrétním autorům. Zároveň provádí shlukování článků, čímž namísto jednotlivých článků na podobné téma může doporučovat přímo (v budoucnu) populární témata.




Obrázek 4.4: *Finální doporučení probíhá v postupných krocích: predikce popularity článků, doporučení pro konkrétního uživatele a tématické shlukování.*

## 4.6 Měření výsledků doporučování

Při používání vytvořeného systému je zapotřebí měřit, zda nové postupy při tvorbě obsahu přinášejí kýžené výsledky v podobě určených cílů – nejčastěji zvýšení počtu zobrazení stránek. Každý článek má příslušné statistiky uložené v analytickém systému, odkud jsou stahovány do vlastního úložiště

implementované aplikace. Díky tomu je možné snadno určit např. průměrný počet zobrazení článku, medián počtu zobrazení článku, statistiky počtu zobrazení pro kategorii, statistiky počtu zobrazení pro jednotlivé autory apod.

Při použití doporučení jako zdroje pro napsání článku není slušné započítat veškeré *pageviews* jako zisk doporučovacího algoritmu. Naopak některé články, napsané na základě špatného doporučení, mohou mít velice podprůměrné statistiky. Je tedy nutné sledovat rozdíl počtu zobrazení článku vůči nějakým způsobem vypočtenému průměru napříč všemi články. Jelikož distribuce počtu zobrazení nabývá exponenciálního charakteru, je vhodným měřítkem *medián* zobrazení článků.

Každý editor v systému označuje, které doporučení použil a následně je možné vyčíslit přínos doporučení díky rozdílu počtu zobrazení oproti medián. Systém by měl v celkovém součtu těchto rozdílů dosahovat minimálně kladných, optimálně co nejvyšších čísel. Pak lze konstatovat, že doporučení přináší danému portálu nějakou přidanou hodnotu a tu konkrétně vyčíslit např. procentuálním zvýšením celkového počtu zobrazení stránek.

Kapitola 5

# Implementační část

## 5.1 Shrnutí a vyhodnocení požadavků

### 5.1.1 Zdroje obsahu

Aplikace zajišťuje pravidelný přísun nového obsahu v podobě textů ze sledovaných zpravodajských portálů, magazínů, blogů a jiných internetových zdrojů. Množina zdrojů je definována seznamem *rss feedů* z jednotlivých webů. Tyto rss feedy je možné ručně spravovat – přidávat, mazat dle potřeby. Články, které se nově objeví v rss, se stahují a ukládají do vlastního datového úložiště aplikace, odkud dochází k dalšímu zpracování. Získání textu je docíleno stažením html stránky z adresy uvedené v rss feedu a vyparsováním relevantní textové oblasti dokumentu. Html stránka dále obsahuje objekty typu nadpis, jméno autora, datum vydání a aktualizace článku, ilustrační obrázek, fotogalerie apod. – tyto části budou prozatím ignorovány (díky uložení stránky ve vlastním úložišti je umožněno s nimi v budoucnu pracovat), protože mohou být nahrazeny daty z rss feedu, kde jsou důležité informace typu nadpis, datum vydání a autor uvedeny.

Dalšími možnostmi sběru dat je přímo napojení na API redakčního systému daného webu (pokud toto umožňuje), stahování generovaných exportů nebo export ze zpřístupněné relační databáze. Výhoda těchto přístupů spočívá především ve vyšší datové kvalitě, příp. sběru dalších metadat typu např. kategorického zařazení textu.

Pro zabránění vzniku duplicit v textových datech – jeden článek může být uveden ve více rss feedech, stažen z rss a následně z API redakčního systému apod. – je nutné systém připravit na automatickou detekci již známého obsahu. Triviální a prozatím dostačující řešení může využívat částečné unikátnosti url adres.

### 5.1.2 Sociální signály

Sociální sítě Facebook, Twitter, Reddit umožňují interagovat s internetovým obsahem a zaznamenávají takové aktivity, jakými jsou např. komentáře, sdílení, „lajkování“, „tweetování“ apod. Tyto údaje, vázané k unikátním url, je možné získávat skrze dostupná API. Aplikace bude zaznamenávat sociální interakce s obsahem – k dané url zjistí počet *likes*, *shares*, *comments* z Facebooku, *count* z Twitteru a počty *ups*, *downs*, *comments* a *score* z Redditu. Tyto metriky v pravidelných časových intervalech uloží do vlastního datového úložistě. Cílem je získat historický průběh počtu interakcí – žádná ze sociálních sítí data v takové podobě nenabízí.

### 5.1.3 Příspěvky ze sociálních sítí

Systém umožňuje sledovat příspěvky vybraných profilů na sociálních sítích. Množinu profilů je možné spravovat dle potřeby. Příspěvky jsou vkládány do vlastního datového úložiště ve strukturovaném formátu – informace o zdroji, datu publikování, obsah, specifické struktury typu hashtag nebo odkaz na profil, url adresy obsažené v textu.

### 5.1.4 Data z analytických nástrojů

Aplikace provádí pravidelný export statistik z analytických nástrojů a párování dat se články stahovanými z rss a dalších zdrojů. Analytické nástroje musí být zpřístupněny konkrétním webovým portálem a musí umožňovat strojové čtení dat. Hlavními zástupci nejrozšířenějších nástrojů jsou: Adobe Omniture, DAX Comscore, Google Analytics. Každý z nich vyžaduje kvůli rozdílným filosofiím správy dat i rozdílný postup při získávání dat.

### 5.1.5 Rekomendační a predikční služba

Vedle samotné akvizice dat jsou nejdůležitějšími komponentami systému rekomendační a predikční služba. Systém pracuje s daty z posledních 24 hodin, provádí na nich pravidelnou aktualizaci predikce popularity a nabízí doporučení na základě profilu uživatele, které navíc ovlivňuje právě predikcí popularity.

Spojení s rekomendační službou musí zajišťovat především: přidávání nových textových dat, mazání starých dat, vytváření uživatelů, aktualizaci interakcí uživatelů, aktualizaci současné sociální popularity, aktualizaci predikcí.

Predikční služba slouží jako mocné orákulum k odhadu budoucí popularity daného tématu. Přijímá články obohacené o atributy jako např. autor, zdroj, sociální statistiky, na jejichž základě samotnou predikci a jistotu predikce vypočítává.

## 5.2 Implementace

### 5.2.1 RSS crawler

Aplikace udržuje interní seznam url adres *rss feedů*, ze kterých má pravidelně stahovat odkazy na nové články. U každého záznamu podporuje základní atributy: název zdroje, odkaz na hlavní stránku webu, jazyk (automaticky se přenáší na získané články), datum poslední aktualizace.

Standardní *rss crawler* by v pravidelných intervalech kontroloval obsah XML každého rss feedu a ukládal nově přibývající url odkazy na články k dalšímu zpracování. Tento proces není komplikovaný, ale je možné využít snadnější cesty, jak data z rss feedů získávat. Existují služby, tzv. *rss čtečky*, které umožňují skrze přívětivé uživatelské rozhraní spravovat a následně sledovat konkrétní množinu rss feedů. Mezi dnes nejpopulárnější rss čtečky patří Feedly.com [7]. Feedly vedle klasického uživatelského rozhraní nabízí možnost strojového čtení dat pomocí *Feedly API* [6].

Pravidelné stahování dat ze všech sledovaných rss feedů je tedy nahrazeno voláním jediného API endpointu. Dalším velkým přínosem je správa množiny rss feedů pomocí komfortního rozhraní Feedly aplikace. Implementační a provozní náročnost se tak značně snižuje.

Feedly API umožňuje skrze endpoint

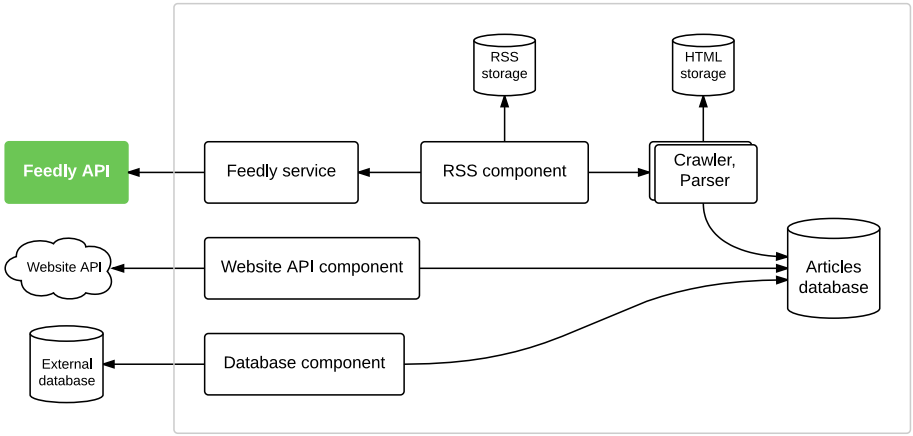
```
POST user/{clientId}/category/global.all
```

získat odkazy na články ze všech zdrojů a zároveň použít filtr na čas stažení. Komponenta pro komunikaci s Feedly API tak udržuje pouze časovou známku posledního volání API, jejíž použitím nedochází ke zbytečnému stahování odkazů známých z předchozích cyklů.

Veškeré získané rss položky jsou ukládány do datového úložiště včetně titulku, odkazu, autora, popisku a data publikace.

### 5.2.2 Redakční systémy a databázové pohledy

Mnohé redakční systémy umožňují stažení databáze článků skrze vlastní API. Jedná se o alternativu stahování obsahu k rss crawleru a následnému zpracování odkazovaných html stránek. Nesmírná výhoda spočívá v kvalitě




Obrázek 5.1: *Komponenty zdrojů textových dat a jejich vzájemné vazby. RSS komponenta využívá API online služby Feedly.com pro stahování nových dat a kompletní správu rss feedů. Získaná data – titulek článku, odkaz na zdroj, datum publikace, autor a popisek – se archivují ve vlastním datovém úložišti. Odkazy na jednotlivé články se ihned předávají dalším částem systému. Crawler provádí stažení cílových html stránek a jejich perzistentní uložení. Parser umnožňuje ze surového html vyfiltrovat samotný text článku, který je uložen k příslušnému záznamu v databázi článků. Komponenta zajišťující komunikaci s API redakčních systémů jednotlivých webů a komponenta pro přístup k databázi mohou získané texty ukládat přímo do databáze článků.*

získaných dat – nadpisy a texty článků jsou kompletní a dobře formátované. API dále umožňují získat velké množství jinak nedostupných metadat, např. kategorické zařazení článku v rámci webu, štítky, zmíněné události nebo osoby v textu, počet návštěv článku atd.

Napojení na vlastní API není možné provádět automaticky pro všechny weby. Jedná se o možnost poskytovanou konkrétním portálům, které API zpřístupní a povolí používat.

Jedním z dalších možných přístupů pro weby, které nepodporují rss, nebo nemají vlastní API, zůstává zpřístupnění relační databáze – vytvoření přístupu pro uživatele a udělení práv čtení databázového pohledu. Opět jsou získaná data v nejlepší možné kvalitě a mohou obsahovat i další přidaná metadata.

Komponenta, zajišťující stahování textových dat, v pravidelných inter-

valech komunikuje s definovanými API, nebo přistupuje k databázovým pohledům a ukládá získaná data přímo do vlastního datového úložiště článků. Základní komponenty a jejich vzájemné vazby popisuje schéma na obrázku 5.1.

### 5.2.3 Crawler

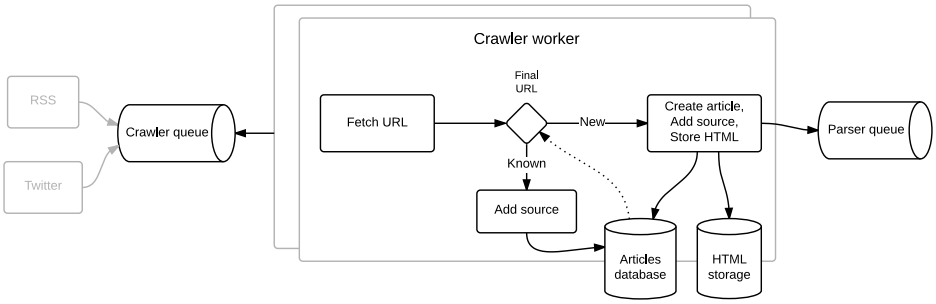
Stažení článků z url získaných pomocí čtení rss feedů obstarává samostatná součást systému nazvaná *crawler*. Funguje na principu fronty příchozích požadavků na stažení konkrétních url. Z fronty požadavky vybírají paralelně pracující procesy, které provedou stažení a uložení dat, popř. navrácení požadavku do fronty po neúspěšném pokusu o stažení. Nejedná se tedy o crawler podle obecné definice [10] – robota, který automaticky prochází web a skrze prolinkování stránek stahuje stránky nově nalezené. Díky návrhu jako samostatné a nezávislé komponenty však není vyloučeno možné budoucí rozšíření.

Crawler, jak je naznačeno na obrázku 5.2, dále provádí nejdůležitější proces systému – vytváření nových článků. V momentě vložení požadavku na stažení url do fronty crawleru není možné určit, zda odkaz směřuje na článek již stažený, nebo se jedná o stránku se článkem neznámým. To je způsobeno díky používání tzv. *zkračovačů url*[2], přidáváním speciálních parametrů do url (využívají např. některé analytické nástroje) a především faktem, že stejný článek může být umístěn na více rozdílných url adresách. Současné implementace zjištění duplicitních požadavků na stažení totožného článku využívá url adresu jako párovací klíč.

Systém vždy musí provést HTTP request na konkrétní url z fronty požadavků a zaznamenat poslední url v řetězci přesměrování – ta se následně považuje za efektivní url článku. Adresa před samotným uložením prochází navíc procesem předzpracování:

- rozdělení na části *host*, *path*, *query* (doménové jméno, cesta, přidané get parametry)
- odstranění *www* subdomény
- seřazení get parametrů podle abecedního pořadí
- filtrace konkrétních get parametrů, např. *action_type_map*, *action_object_map*, *ncid*, *ITO*, *TrackID* apod.

[2]Zkracovač url je služba, která umožňuje pomocí přesměrování skrze vlastní doménu zkrátit délku url adresy. Pro každou vstupní url vygeneruje novou unikátní url s kódem malé délky.




Obrázek 5.2: *Schéma crawler komponenty zajišťující stahování nových článků. Ze vstupní fronty každý, paralelně běžící, crawler worker aktivně vybírá požadavky, které obsahují url pro stažení článku. Proces pomocí kontrol url adres zajišťuje, že nedochází k duplikaci již stažených článků. Pokud se objeví více položek obsahující url na stejný zdrojový článek, je k existujícímu článku pouze zaznamenán nový zdroj – např. jiný rss feed, nebo profil z Twitteru. Při vytvoření nového článku v databázi se do fronty parser komponenty umisťuje požadavek na zpracování stažené html stránky.*

– filtrace get paramterů podle vzoru, např. *utm_**, *fb_**, *og.**, *_sm_b**

– opětovné složení jednotlivých částí.

Cílem je odstranit rozdíly vzniklé rozdílným zápisem stejné url. Filtrování se provádí s účelem vynechání redundatních parametrů, které by při ponechání znemožnily kompletní párování adres. (Jedná se samozřejmě o heuristický přístup, který rozhodně nepokrývá veškeré případy. Pro základní použití ale dostačuje.)

V momentě, kdy do fronty požadavků na stažení vstupují odkazy získané z příspěvků na sociální síti Twitter, je nutné zajistit, aby se na cílových stránkách skutečně nacházely texty článků. Zde se opět používá heuristický přístup: stránka nesmí být hlavní stránkou dané domény a text získaný parsováním html musí být delší než 100 slov. Tyto podmínky splňuje mnoho čistě obsahových statických stránek. Ty se ale (oproti odkazům na skutečné články) objevují v tak malém množství, že výstupy nijak neovlivňují.

Dalším důležitým procesem, který crawler zajišťuje, je přiřazování množiny zdrojů k jednotlivým uloženým článkům. Pokud je nalezena duplicita

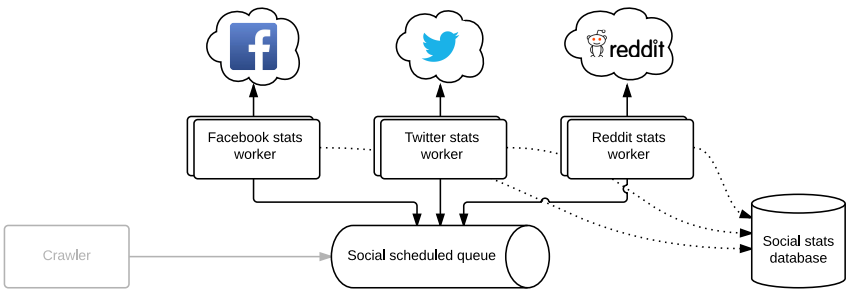
v url ve vstupní frontě crawleru, nevytváří se nový článek, ale existujícímu článku je přidán další zdroj podle toho, odkud byl záznam do fronty umístěn – každá položka ve frontě má atributy: typ zdroje, zdrojový feed, zdrojový příspěvek.

### 5.2.4 Crawler statistik ze sociálních sítí

Stahování hodnot tzv. sociálních signálů probíhá pomocí API jednotlivých soicálních sítí (popsáno v kapitole 2.3). Ihned po uložení nového článku do vlastní databáze se do *rozvrhované fronty* umístí požadavky na stažení hodnot signálů v definovaných časových odstupech:

- Facebook: aktuální (ihned), 30 minut, 1 hodina, 2, 4, 8, 12, 16, 24 hodin
- Twitter: aktuální (ihned), 30 minut, 1 hodina, 2, 4, 8, 12, 16, 24 hodin
- Reddit: aktuální (ihned), 30 minut, 1 hodina, 2, 4, 24 hodin

Tím je docíleno stažení historického průběhu hodnot všech potřebných metrik.




Obrázek 5.3: *Crawler statistik ze sociálních sítí využívá sdílené „rozvrhované“ fronty, odkud pracující procesy, schopné napojit se na API jednotllivých sociálních sítí, vybírají požadavky na stažení statistik ke konkrétnímu článku. Po stažení dat každý proces uloží hodnoty do databáze sociálních statistik.*

Použitá „rozvrhovaná“ fronta má funkcionalitu umožňující, že požadavky do fronty umístěné mohou obsahovat časovou známku, od které je

požadavek dostupný k výběru z fronty. Tím je implementováno stahování statistik ve zmíněných časových odstupech. Na tuto frontu jsou napojeny jednotlivé procesy pro komunikaci s API sociálních sítí a ukládání statistik do datového úložiště aplikace. Tyto procesy jsou spuštěny paralelně v několika instancích.

### 5.2.5 Twitter crawler

Získávání příspěvků ze sociální sítě Twitter zajišťuje další samostatná komponenta systému. Dle uloženého seznamu profilů, ze kterých se mají tweety stahovat, dochází ke komunikaci s Twitterem skrze REST API (viz kapitola 2.2) a kontrole přibývání příspěvků v daných profilech. Nové tweety se ukládají do datového úložiště aplikace a url odkazy z tweetů se předávají do crawleru, jak je ilustrováno na obrázku 5.2.

Komponenta pracuje v několika instancích, přičemž díky sdílené frontě požadavků na stažení jednotlivých profilů umožňuje kontrolovat limity na volání Twitter API. Limity v současné době činí 300 požadavků v klouzavém okně posledních 15 minut.

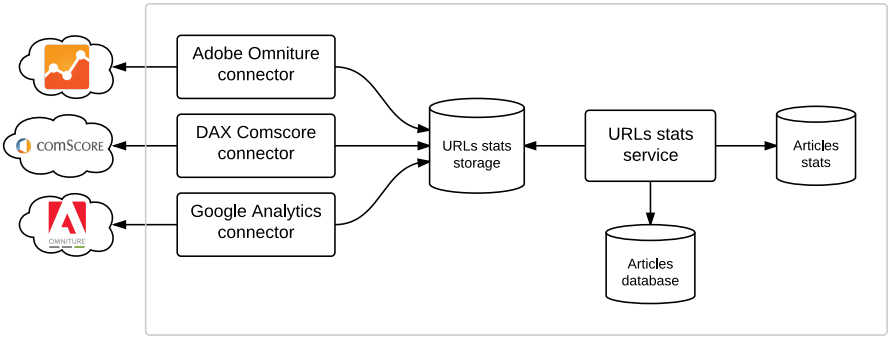
### 5.2.6 Data z analytických nástrojů

Stahování dat o návštěvnosti stránek zajišťují tzv. konektory pro analytické systémy. Každý konektor je samostatně běžící proces, který je schopen stáhnout data z konkrétního analytického systému – pomocí dostupných API, která jsou popsána v kapitole 2.4.

V definovaných časových intervalech (např. každou hodinu) konektor stáhne data a uloží v podobě souboru na disk. Další služba, obstarávající parsování výstupů (každý analytický systém používá jiný formát) z analytických systémů, tyto soubory zpracovává a ukládá získané hodnoty ke konkrétním článků v databázi společně s příslušnou časovou známkou. Díky tomuto postupu má aplikace historizovány údaje o návštěvnosti každého článku v čase.

### 5.2.7 Parsování textů

Komponenta *parser* zajišťuje automatické vyfiltrování čistého textu článku ze staženého html souboru. Využívány k tomu jsou zdarma dostupné nástroje *boilerpipe* a *readabily*, viz kapitola 2.1.1, ke kterým byl vytvořen obalující konektor. Konektor běží paralelně v několika nezávislých procesech a čeká na požadavky ve frontě, viz obrázek 5.2, které postupně zpracovává a získané texty ukládá ke článkům v datovém úložišti.



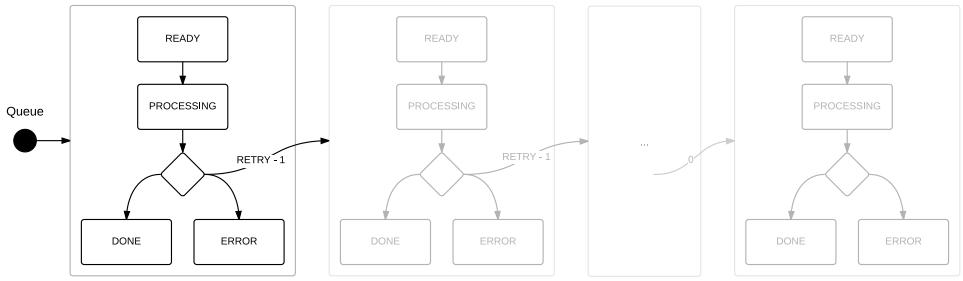

Obrázek 5.4: *Stahování dat z analytických nástrojů zajišťují samostatné konektory s implementovaným napojením na API analytických systémů. Po stažení dat se výstup ukládá do souboru na diskové úložiště. Odtud další nezávislý proces soubory zpracovává a páruje data o násvštěvnosti s jednotlivými články a ukládá příslušné statistiky do datového úložiště.*

## 5.3 Implementace fronty

Zmíněné použití front pro komunikaci mezi jednotlivými, nezávislými částmi aplikace umožňuje dobře škálovat jednotlivé části systému. V případě nalezení úzkého hrdla v některé z komponent stačí danou komponentu spustit ve větším množství instancí.

Fronta nevyužívá žádného proprietálního řešení, ale je implementována pomocí relační databáze. To především z důvodu, že na trhu neexistuje žádný rozšířený nástroj nebo framework, který umožňuje provozovat „rozvrhovanou“ frontu nebo umožňoval opakované umístění požadavku do fronty potřebným způsobem. Dalším důvodem je historizace požadavků ve frontách.

Každá fronta je reprezentována jednou tabulkou v relační databázi, do které se vkládají záznamy o požadované akci. Každý záznam má definovaný stav, ve kterém se nachází. Při vložení je nastaven stav *READY* a požadavek tak může být vyjmut z fronty (převeden do stavu *PROCESSING*) jakýmkoliv z paralelně běžících procesů. Při neúspěšném provedení požadavku nebo po vypršení časové dotace procesu, je požadavek převeden do stavu *ERROR* a znovu vložen do fronty se sníženou hodnotou *RETRY* (ta když nebude hodnoty 0, požadavek se zahodí). Při úspěšném odbavení se nastaví stav *DONE*. Přechod jednotlivých stavů ilustruje obrázek 5.5.




Obrázek 5.5: *Přechody stavů požadavků ve frontě. Při umístění do fronty se požadavek dostává do stavu READY, podkud při vybrání z fronty nějakým procesem přechází do stavu PROCESSING. Při neúspěšném provedení je nastaven stav ERROR a do fronty umístěn nový požadavek se sníženou hodnotou RETRY – při dosažení hodnoty 0 se požadavek již neopakuje.*

### 5.3.1 Propojení s rekomendační službou

Napojení na rekomendační službu je řešeno skrze definované REST API [18]. Aplikace při uložení nového článku zároveň článek vloží pomocí API do rekomendačního enginu. Dále aktualizaje veškeré nově získané údaje o hodnotách sociálních statistik a hodnotách predikce. Aplikace dále řeší zakládaní uživatelů a nahrávání jejich interakcí do rekomendační služby.

Pro získání doporučení pro konkrétního uživatele slouží speciální API volání, které vrací seznam *id* doporučených článků seřazený podle vypočteného pořadí.

```
GET /{databaseId}/users/{userId}/recomms/?count={count}
[&filter=<filter>][&booster=<booster>]
```

*Booster* slouží k definování vzorce pro ruční upravení pořadí objektů (např. vynásobení predikcí popularity). *Filter* slouží k filtrování objektů dle jejich atributů.

### 5.3.2 Predikce

Stahování predikcí je řešeno pomocí definovaného REST API [17]. V těle požadavku se předává 2D pole se seznamen objektů a všech jejich atributů. Návratovou hodnotou je pole se seznamem predikcí pro objekty na vstupu. Formát vsupu je dán nastavením konkrétního modelu, reps. jeho datasetu.

Predikční request má následující formát:

```
POST {customerId}/model/{modelId}/predict
```

*ModelId* je identifikátor naučeného predikčního modelu pro konkrétní úlohu.

### 5.3.3 Exporty predikčních datasetů

Predikční služba potřebuje mít k dispozici data, ze kterých vytvoří model pro vybavování doporučení. Tato data se nahrávají pomodí dalšího API volání:

```
POST {customerId}/dataset/{datasetId}/data
```

Předáno je 2D pole s daty v definovaném formátu – počet sloupců, datové typy atd.

Data jsou aktualizivány v pravidelných časových intervalech. Přípravu do potřebného formátu zajišťují komplikované sql dotazy a transformační skripty.

### 5.3.4 Clustrování

Shlukování článků do tématických celků zajišťuje další externí služba s vlastním REST API. Clustering API funguje na stejném pricipu jako predikční API, pouze používá jiný formát výstupu. Navrací json objekt, který ze vstupního seznamu *id* článků vytvoří strukturu s množinami článků spadajícími do stejného tématu.

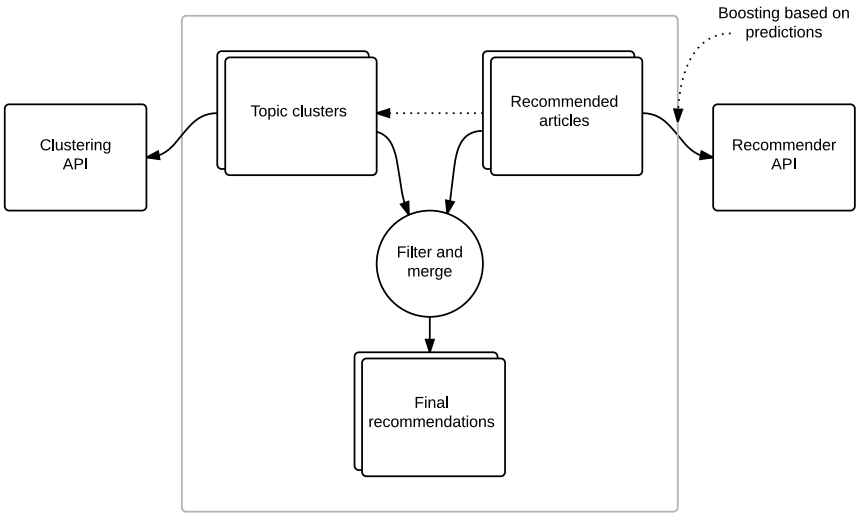
Clustering API využívá pravidelně aktualizovaná data pro naučení clustrovacího modelu. Data jsou nahrávána v podobě 2D pole stejným způsobem jako u predikčního API. Aktualizace těchto dat ale probíhá v mnohem kratších časových intervalech.

### 5.3.5 Seznam doporučení

Generování doporučení pro jednotlivé uživatele aplikace spočívá v kombinaci rekomendační a clustrovací služby. Predikce popularit jsou pravidelně na pozadí aktualizovány a aktuální data se již nacházejí v rekomendační službě. Tím je umožněno *boostovat* doporučení pomocí predikovaných popularit článků.

Výsledný seznam doporučených článků se generuje postupem ilustrovaným na obrázku 5.6. První dotaz směřuje na rekomendační API, seznam

získaných id je následně předán do clustrovacího modulu, který pomocí clustering API provede shlukování do témat. Nádledně proběhnou filtrace dat podle potřeb pro konkrétní zobrazení a návratem je seznam doporučených článků.



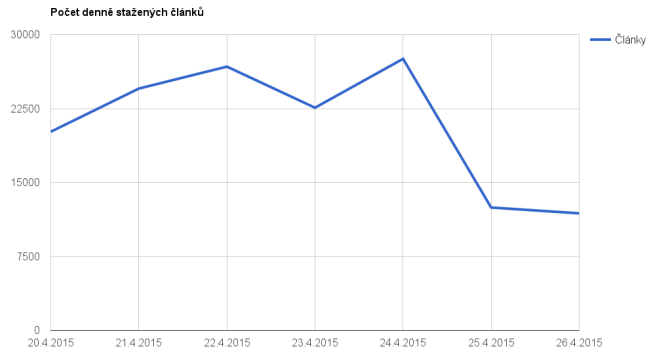

Obrázek 5.6: *recommendations clustering*

## 5.4 Testovací provoz

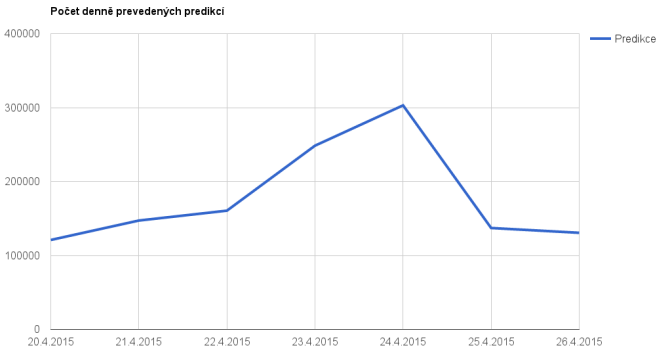
Testování provozu probíhalo na datech pravidelně stahovaných z cca 2000 zdrojů (převážně zpravodajské portály). Systém denně zpracoval zhruba 20 tisíc článků a provedl až 300 tisíc predikčních dotazů. (V testovacím provozu bylo nasazeno nejdříve 6 a následně 11 různých predikčních modelů.) Ukázky z týdenního provozu ilustrují obrázky 5.7 a 5.8.

Dále probíhalo stahování sociálních signálů k jednotlivým článkům. Ukázku hodnot pro jeden vybraný článek uvádí tabulka 5.1.

Ukázky doporučení jsou vypsány v tabulkách 5.3 a 5.4, kde tabulka 5.3 uvádí doporučení pouze na základě interakcí uživatele a tabulka 5.4 doporučení ovlivněné predikcí popularity. Zadány byly interakce z technické oblasti, konkrétní seznam uvádí tabulka 5.2.




Obrázek 5.7: *Počet denně stažených článků.*




Obrázek 5.8: *Počet denně provedených predikcí. Množina modelů byla během tohoto období rozšířena.*

Tabulka 5.1: *Hodnoty stažených Facebook sociálních signálů pro ukázkový článek. Článek není starší než 24 hodin, proto zatím čeká na stažení statistik v 1440 minutách (24 hodin).*

| **Minut po stažení** | **Shares** | **Likes** | **Comments** |
|---|---|---|---|
| 0 | 7 | 0 | 5 |
| 30 | 103 | 99 | 98 |
| 60 | 189 | 177 | 211 |
| 120 | 384 | 335 | 383 |
| 240 | 559 | 530 | 624 |
| 480 | 1455 | 1293 | 1252 |
| 720 | 1702 | 1510 | 1425 |
| 1440 | | | |

Tabulka 5.2: *Zadané interakce – označení článků, že o téma má uživatel zájem – z technické oblasti.*

| **Téma** |
|---|
| Google Ventures' Bill Maris Says Uber Is The Fastest-Growing Company |
| Snapchat's first Discover update makes it easier to share stories |
| Netflix giving Adam Sandler creative control is totally backfiring |
| Facebook Messenger had 1 million video calls in the first 2 days |
| Apple to unveil redesigned Apple TV remote at WWDC, report says |
| Microsoft's new browser has a name only a mother could love |
| Apple and Microsoft's visions for the future are delightfully different |

Tabulka 5.3: *Doporučení vygenerovaná na základě zadaných interakcí z tabulky 5.2 bez ovlivnění predikcí popularity.*

| **Téma** | **Pred.** |
|---|---|
| The Next Apple TV Could Get A Touch-Friendly Remote Redesign | 10+ |
| Microsoft says it has no plans to open-source its new Edge browser | 10+ |
| These may be the first details on Apple's next big iPhone update iOS 9 | 100+ |
| Salesforce Shares Pop As Rumors Of Microsoft Buying The CRM Shop | 100+ |
| Snapchat now lets you share Discover posts | 10+ |
| Volkswagen's Apple Watch App Will Notify When Your Driver Speeds | 100+ |
| Google is planning to spend $150 million to make its workforce diverse | 1000+ |

Tabulka 5.4: *Doporučení vygenerovaná na základě zadaných interakcí z tabulky 5.2 s ovlivněním predikcí popularity, konkrétně predikce počtu Facebook shares. Booster pro rekomendační službu byl nastaven na násobení hodnotou log10(predikce Facebook shares).*

| **Téma** | **Pred.** |
|---|---|
| How the NSA Converts Spoken Words Into Searchable Text | 1000+ |
| This Is How The Controllers Are Engineering A Fake Apocalypse | 100+ |
| Black Widow's feminist heroism | 1000+ |
| Indiana Jones Sequel Confirmed | 1000+ |
| Nucleus Scientific Launches To Revolutionize Batteries | 100+ |
| Postmodern Jukebox: From YouTube covers to a world tour | 100+ |

# Závěr

Vytvořená aplikace propojuje svět zpracování dat a *strojového učení* s online žurnalistikou. Provádí stahování velkého množství článků ze světových zpravodajských portálů, online magazínů a blogů. K jednotlivým článkům připojuje pravidelně aktualizované statistiky ze sociálních sítí Facebook, Twitter a Reddit. Veškerá data jsou následně využívána ke sledování světového dění – aktuálně populárních témat a událostí. Aplikace poskytuje uživatelům systému (redaktorům) personalizované doporučování článků a témat, které mají potenciál být zajímavé pro jejich čtenáře. Tato doporučení jsou založená nejen na osobních preferencích redaktorů, ale jsou také ovlivněna predikcí budoucí popularity jednotlivých témat. Jak rekomendace, tak predikce jsou zajišťovány externím systémem. Aplikace do těchto systémů nahrává předzpracovaná data a veškeré dostupné příznaky, které jsou pro dané procesy užitečné. Generované výstupy se buď ukládají do vlastního datového úložiště pro další zpracování, nebo se ihned zobrazují uživateli. Další důležitou komponentou systému je služba, která pomocí clustrovacího algoritmu provádí tématické shlukování článků. To umožňuje prezentovat uživateli nejvíce zajímavá témata namísto dlouhého seznamu jednotlivých článků. Systém díky své modulární architektuře umožňuje odbavení velkého množství dat a je připraven pro možné budoucí škálování.

Dále byly představeny a implementovány postupy pro zpracování surových textových dat zahrnující: získání textů článků z html souborů, předzpracování textů do podoby příznakových vektorů, vypočtení podobnosti jednotlivých textů a následnou vizualizaci převedením do grafové struktury. Zpracovaná data byla následně obohacena o příznaky z analytických systémů (návštěvnost stránek), které dále rozšířily vytvořené vizualizace. Tyto postupy byly následně využity pro tvorbu výše zmíněného online systému.

Práce se také zabývá tématiku akvizice textových dat, dat ze sociálních

sítí, dat z analytických systémů a jejich propojení a strukturalizací. Konkrétně popisuje získání dat ze sociálních sítí Facebook, Twitter, Reddit a analytických systémů Google Analytics, DAX Comscore a Adobe Omniture.

Vytvořená aplikace v době testování provedla denně stažení a zpracování zhruba 30 tisíc článků z 2000 zdrojů a denně zaslala zhruba 300 tisíc predikčních požadavků. Pro jednotlivé uživatele pak bylo umožněno zobrazit výpis doporučených témat, který byl vytvořený pomocí netriviální kombinace pokročilé predikční analýzy a doporučování obsahu.

# Literatura


[1] Alpaydin, E.; Kaynak, C.: Cascading classifiers. *Kybernetika*, ročník 34, č. 4, 1998: s. 369–374.

[2] Baburov, Y.: Fast python port of arc90's readability tool [online]. [2011], [Cit. 2015-04-21]. Dostupné z: `https://pypi.python.org/pypi/readability-lxml`

[3] Bastian, M.; Heymann, S.; Jacomy, M.: Gephi: An Open Source Software for Exploring and Manipulating Networks. 2009. Dostupné z: `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154`

[4] Blei, D. M.; Ng, A. Y.; Jordan, M. I.: Latent dirichlet allocation. *the Journal of machine Learning research*, ročník 3, 2003: s. 993–1022.

[5] Breiman, L.: Bagging predictors. *Machine learning*, ročník 24, č. 2, 1996: s. 123–140.

[6] DevHD, I.: The feedly Cloud API [online]. [2015], [Cit. 2015-04-23]. Dostupné z: `https://developer.feedly.com/`

[7] DevHD, I.: Feedly.com [online]. [2015], [Cit. 2015-04-23]. Dostupné z: `http://feedly.com/`

[8] Hastie, T.; Tibshirani, R.; Friedman, J.; aj.: *The elements of statistical learning*, ročník 2. Springer, 2009.

[9] Jacomy, M.; Venturini, T.; Heymann, S.; aj.: ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. *PLoS ONE*, ročník 9, č. 6, 06 2014: str. e98679, doi:10.1371/journal.pone.0098679.

Dostupné z: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0098679

[10] Kobayashi, M.; Takeda, K.: Information Retrieval on the Web. *ACM Comput. Surv.*, ročník 32, č. 2, Červen 2000: s. 144–173, ISSN 0360-0300, doi:10.1145/358923.358934. Dostupné z: http://doi.acm.org/10.1145/358923.358934

[11] Kohlschütter, C.; Fankhauser, P.; Nejdl, W.: Boilerplate Detection Using Shallow Text Features. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, New York, NY, USA: ACM, 2010, ISBN 978-1-60558-889-6, s. 441–450, doi:10.1145/1718487.1718542. Dostupné z: http://doi.acm.org/10.1145/1718487.1718542

[12] Kohlschütter, C.: Boilerplate Removal and Fulltext Extraction from HTML pages [online]. [2011], [Cit. 2015-04-21]. Dostupné z: https://code.google.com/p/boilerpipe/

[13] Mitchell, T. M.: *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc., první vydání, 1997, ISBN 0070428077, 9780070428072.

[14] Smola, A.; Vishwanathan, S.: *Introduction to Machine Learning*. SCambridge University Press, 2008.

[15] Weiss, S. M.; Indurkhya, N.; Zhang, T.; aj.: *Text mining: predictive methods for analyzing unstructured information*. Springer Science & Business Media, 2010.

[16] Černý, J.: Metody pro kombinování modelů a klasifikátorů. [2010], [Cit. 2013-04-29].

[17] Černý, J.: Prediction API documentation [online]. [2015], [Cit. 2015-04-29]. Dostupné z: https://wiki.modgen.net/api/prediction-api

[18] Řehořek, T.: Recommender API documentation [online]. [2015], [Cit. 2015-04-29]. Dostupné z: https://wiki.modgen.net/api/generic-api

Příloha A

# Seznam použitých zkratek

**GUI** Graphical user interface

**XML** Extensible markup language

**HTML** Extensible markup language

Příloha **B**

# Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF