BI - VWM

Vektorový model

Jan Peřina & Tomáš Taro
FIT ČVUT LS 2019

Obsah

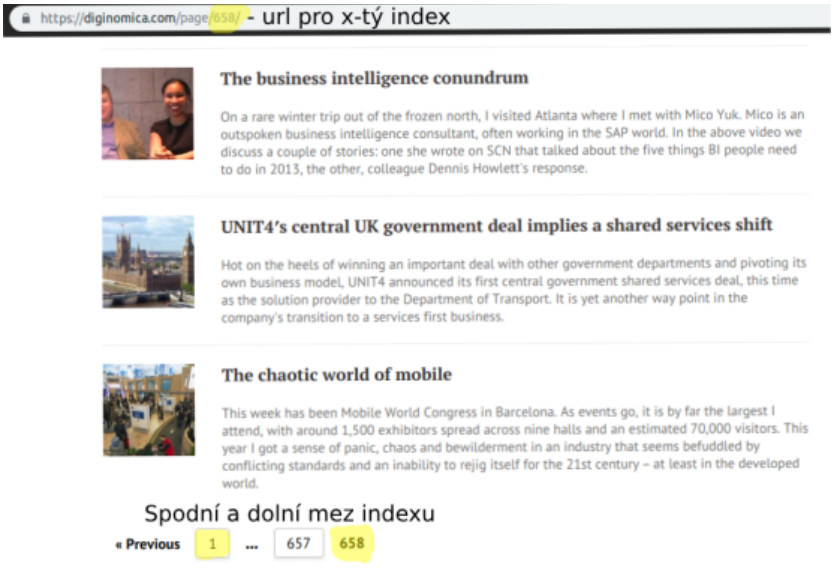
Popis

Cílem naší práce bylo vypracování vyhledávacího enginu, který funguje na bázi vektorového modelu. Náš engine pracuje nad statickou kolekcí článků z anglického technického blogu, které jsou předem stažené na filesystem a zpracované pomocí knihovny nltk (natural language toolkit).

Způsob řešení

Kolekce dokumentů

Naším prvním úkolem bylo vytvoření dostatečného velkého a kvalitního zdroje dokumentů pro náš engine. Rozhodli jsme se proto najít anglický blog, který má dostatek článků, se kterými budeme schopni pracovat. Pro anglické články jsme se rozhodli z toho důvodu, že podpora anglického jazyka v knihovnách pro práci s textem je větší než pro češtinu. Problém, na který jsme však narazili, je fakt že většina moderních webů načítá obsah dynamicky pomocí JavaScriptu (například lazy-loading textu, atp.), které se obecně hůře dolují a to z toho důvodu, že crawler musí být schopen interpretovat JavaScript, čímž je schopný se dostat k samotnému obsahu článku. Což má ale za následek větší využití systémových prostředků, než při scrapování statické HTML stránky. Proto jsme se rozhodli zpracovávat blog se statickým webem. Pro účely naší práce jsme vybrali [https://diginomica.com,](https://diginomica.com) který má své články členěné do stránek po přibližně jedenácti článcích, kde každá tato stránka je indexovaná v url a velikost indexu můžeme jednoduše dohledat (viz. obrázek). Což je pro naše účely ideální.



Následné pro každý článek, který je reprezentován párovým tagem <article> získáme jeho url adresu, která je v atributu href elementu <a>, který se nachází v elementu <h3> s třídou entry-title uvnitř elementu <article>.

Z každého takto získaného odkazu dále vyextrahujeme název článku, který se nachází v tagu <h1> s třídou entry-title a samotný text článku, který se nachází v tagu s třídou entry-content. Výsledný text ukládáme jako plain-textový dokument s názvem, který odpovídá názvu dokumentu.

Preprocessing

Všechny takto získané dokumenty následně zpracujeme a připravíme pro jednodušší indexaci a to pomocí odstranění takzvaných stop words, což jsou slova, která se v daném jazyce vyskytují často napříč tomu, že jejich informační význam v rámci dokumentu je minimální, nebo nulový. V angličtině jsou to například slova the, a, I, atp. Dále jsou dokumenty lemmatizovaný, což je proces normalizace textu v dokumentu, kde například pro anglické slovo **goes** a **went** dostaneme slovo **go**, tedy dostaneme jednotnou formu pro slova stejného významu. Což je výhoda oproti stemmingu, který vrací slova v základním tvaru, takže bychom pro stejný vstup dostali **go** a **went**. Ve výsledku tedy pomocí lemmatizace dostaneme obecnější přehled o obsahu dokumentu. Dále spočítáme počet výskytů daného zlemmatizovaného slova v dokumentu.

Pro takto zpracované dokumenty dále spočítáme relativní váhu termu vůči dokumentu, a to pomocí

$$f_{ij} = \text{frequency of term } t_i \text{in document } d_j$$

$$tf_{ij} = \frac{f_{ij}}{\max\{f_{ij}\}}$$

Díky čemuž dostaneme normalizované váhy v intervalu (0, 1>. Poté spočítáme inverzní frekvenci termu v kolekci a to vzorcem:

$$df_i = \text{document frequency of term } t_i \text{ in collection}$$

$$idf_i = \log_2(\frac{\mathrm{n}}{df_i})$$

Následně se vypočítá váha termu pro daný dokument vzorcem:

$$w = t_f * id_f$$

*Všechny vzorce pro vektorový model jsou přejaté z přednášky

Tyto váhy si uložíme v podobě invertovaného indexu do json souboru s následující strukturou:
term : { document_id : weight, ...}

```
{
  "forest": {"3": 0.3, "7": 0.2, "10": 0.1, "15": 0.8, "16": 0.3, "18": 0.4, "20": 0.7},
  "mountain": { "2": 0.2, "3": 0.6, "8": 0.6, "10": 0.6, "17": 0.5},
  "nature": {"2": 0.9, "7": 0.3, "8": 0.7, "9": 0.1, "16": 0.8, "17": 0.9}
}
```

Vyhledávání

Webová aplikace uživateli předloží náhodný dokument z kolekce, a vypíše uživateli 10 nejpodobnějších dokumentů na základě kosinové podobnosti jejich vektorů a to společně s jejich ID, váhou k aktuálnímu dokumentu a odkazem. Tyto vektory jsou sestaveny z 10 nejčetnějších slov v dokumentu. Nakonec aplikace vypíše i celkovou dobu hledání, a to jak u sekvenčního vyhledávání, tak u vyhledávání v invertovaném indexu.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

*Vzorec cosinove podobnosti z Wikipedie

Implementace

Webová aplikace

Je napsaná v jazyce PHP s využitím frameworku Nette a CSS balíčku Bootstrap 4.

Jednoduchý web crawler a preprocessing dokumentů

Je napsán v Pythonu (verze 3.6 a vyšší) s pomocí některých knihoven, pro spuštění je tedy podmínkou, aby Python interpreter měl k dispozici následující knihovny:

Urllib3

Knihovna Urlib3 slouží k síťové komunikaci pomocí protokolu HTTP/HTTPS.
Jedná se o nízkoúrovňové rozhraní. (uri)

Beautifulsoup4

Nejvíce používaná knihovna pro web scraping/crawling v jazyce Python, je uživatelsky přívětivá a pomocí několika řádků se dá i z komplexního webu vyextrahovat libovolná informace. Knihovna má vlastní parser a serializaci, který lze ale nahradit jinou externí knihovnou, například lxml, nebo HTMLParser. (uri)

Lxml

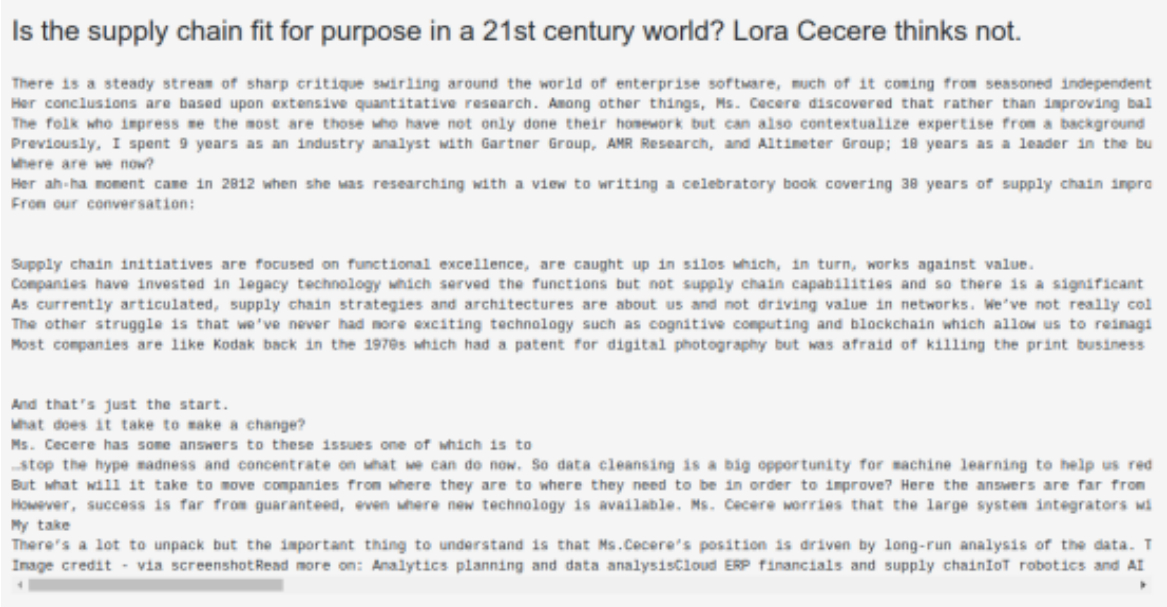
XML parser, který je schopný parsovat i HTML formát, jedná se o jednu z nejrychlejších knihoven, pracuje s C knihovnami libxml2 a libxslt. (uri)

Nltk

Natural language toolkit je jednou z nejpoužívanějších knihoven pro práci s lidským jazykem, podporuje velkou škálu jazyků. V našem případě jí používáme k odstranění stop words, a extrakci unikátních termů a jejich počtu v daném dokumentu. (uri)

Příklad výstupu

Po vybrání způsobu vyhledávání (sekvenční / invertovaný index) se uživateli zobrazí náhodný článek z kolekce.



Is the supply chain fit for purpose in a 21st century world? Lora Cecere thinks not.

There is a steady stream of sharp critique swirling around the world of enterprise software, much of it coming from seasoned independent
Her conclusions are based upon extensive quantitative research. Among other things, Ms. Cecere discovered that rather than improving bal
The folk who impress me the most are those who have not only done their homework but can also contextualize expertise from a background
Previously, I spent 9 years as an industry analyst with Gartner Group, AMR Research, and Altimeter Group; 10 years as a leader in the bu
Where are we now?
Her ah-ha moment came in 2012 when she was researching with a view to writing a celebratory book covering 30 years of supply chain impro
From our conversation:

Supply chain initiatives are focused on functional excellence, are caught up in silos which, in turn, works against value.
Companies have invested in legacy technology which served the functions but not supply chain capabilities and so there is a significant
As currently articulated, supply chain strategies and architectures are about us and not driving value in networks. We've not really col
The other struggle is that we've never had more exciting technology such as cognitive computing and blockchain which allow us to reimagi
Most companies are like Kodak back in the 1970s which had a patent for digital photography but was afraid of killing the print business

And that's just the start.
What does it take to make a change?
Ms. Cecere has some answers to these issues one of which is to
…stop the hype madness and concentrate on what we can do now. So data cleansing is a big opportunity for machine learning to help us red
But what will it take to move companies from where they are to where they need to be in order to improve? Here the answers are far from
However, success is far from guaranteed, even where new technology is available. Ms. Cecere worries that the large system integrators wi
My take
There's a lot to unpack but the important thing to understand is that Ms.Cecere's position is driven by long-run analysis of the data. T
Image credit - via screenshotRead more on: Analytics planning and data analysisCloud ERP financials and supply chainIoT robotics and AI

Kromě samotného článku se uživateli zobrazí 10 nejpodobnějších článků.

Similar documents:

ID: 813
Article weight: 0.95164161166
Article name: Enterprise hits and misses – peeling back the AI hype, and handing out this year's cheeky blogger awards

ID: 769
Article weight: 0.88061003730693
Article name: Enterprise hits and misses – job boards face critique, and a powerful AI tool gets put on probation

ID: 921
Article weight: 0.86716499393495
Article name: Enterprise hits and misses – fighting through digital nightmares; IT does matter after all

ID: 955
Article weight: 0.84753045234352
Article name: Enterprise hits and misses – retail gets a once-over, and Facebook asks for a privacy do-over

ID: 440
Article weight: 0.84498995877168
Article name: Enterprise hits and misses – KFC and Pizza Hut cook the omni-channel, while we debate the enterprise user experience

ID: 490
Article weight: 0.8433319194787
Article name: L'Oréal's CX – baking in the supply chain elements

ID: 130
Article weight: 0.8370194111769
Article name: Enterprise hits and misses – surviving tech projects and debating enterprise blockchains

ID: 456
Article weight: 0.81666497235523
Article name: Enterprise hits and misses – Thanksgiving can't save us from AIOps

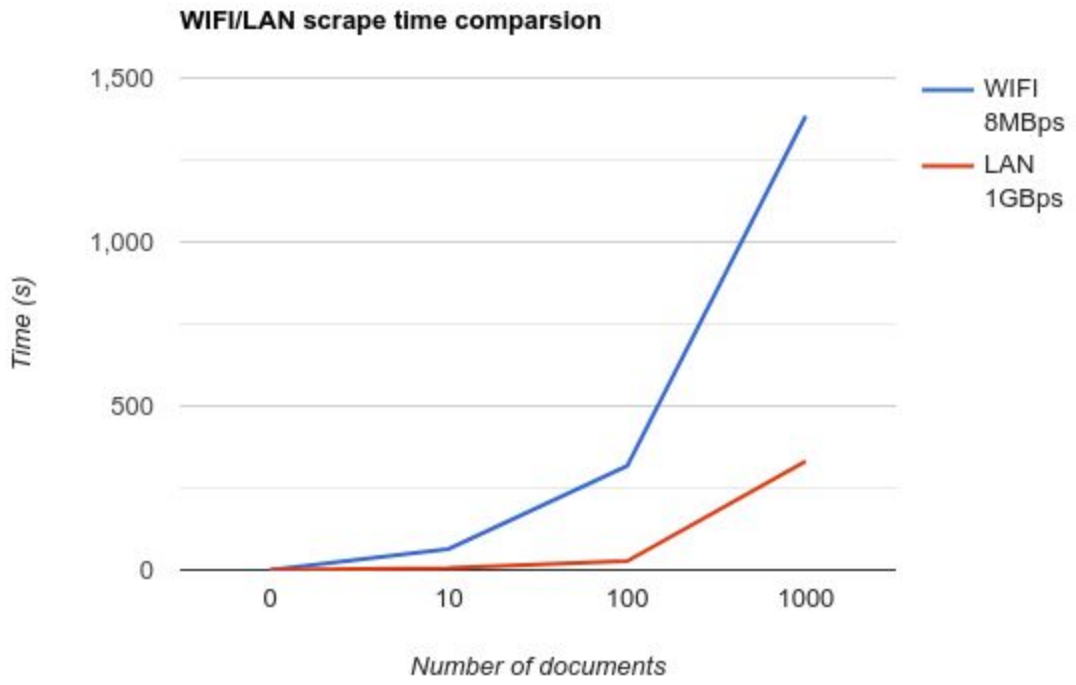
ID: 403
Article weight: 0.81112695219318
Article name: Enterprise hits and misses – blockchain is a paradox; AI is a customer service automater

ID: 260
Article weight: 0.80432301751884
Article name: Enterprise hits and misses – making sense of the Chinese server sabotage allegations, as silly season rolls on

Timing of sequential search: 1326 ms

Experimentalni sekce

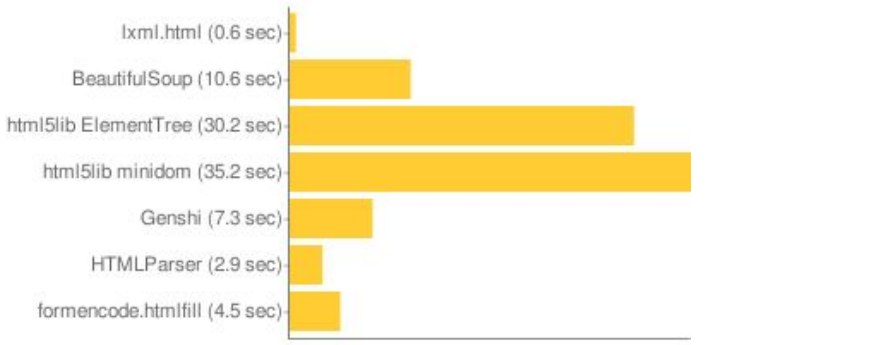
Tím, že náš engine si předem stahuje články z webů, zajímali jsme se o výkonnostní a časové statistiky našeho způsobu řešení. Časová náročnost na zaplnění kolekce se odvíjí například od rychlosti připojení k internetu zařízení, na kterém engine běží. Například jsme testovali čas, který náš engine potřebuje pro načtení 10/100/1000 dokumentů. Naše testy proběhly na WIFI/LAN síti kolejí Strahov.



Z grafu je vidět, že na WIFI síti se s větším objemem dat dostáváme do několikanásobně většího času (bottleneck effect).

Scraping

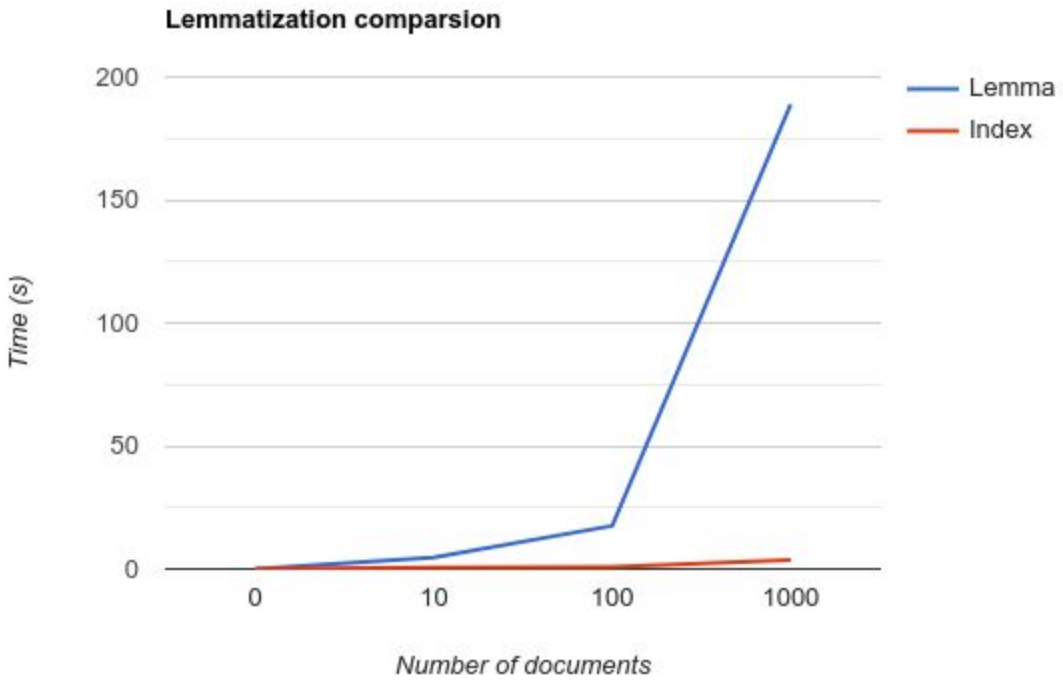
V našem případě jsme nejdříve používali HTML parser knihovny Beautifulsoup, který byl však pomalý, a proto byly naše výsledky daleko pomalejší, než jiné parsery, rozhodli jsme se proto použít knihovnu lxml, čímž jsme docílili zrychlení parsování stránek a tím i celkový běh programu.



Srovnání rychlosti parsování jednotlivých knihoven

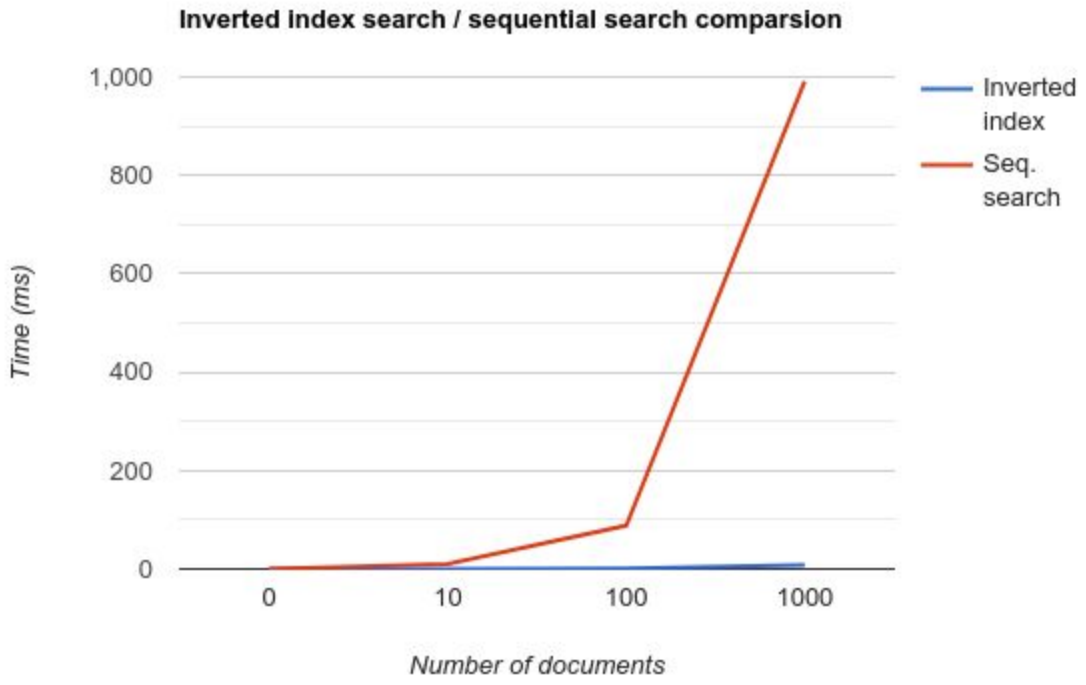
Lemmatizace a indexace

Z následujícího grafu, který srovnává dobu běhu fáze lemmatizace a indexace je vidět, že s větším počtem dokumentů je proces lemmatizace náročnější. Tento rozdíl by mohl být zmenšen větším množstvím operační paměti na koncovém zařízení, nebo lepší implementací lemmatizace.



Porovnání vyhledávání

V našem řešení jsme implementovali dva druhy vyhledávání, oba jsou založeny na cosinové podobnosti. Je zřejmé, že efektivita vyhledávání v invertovaném vektoru je násobně efektivnější, než sekvenční vyhledávání.



Diskuze

Naše práce v aktuálním stavu může pokrýt potřeby jednoduchého vyhledávacího enginu, avšak je možné, že některé části naší práce by mohly být zrychleny za pomoci nejrůžnějších progresivních algoritmů, nebo knihoven, čímž by časová a paměťová náročnost mohla klesnout. Nevýhodou našeho řešení je fakt, že shromáždění většího množství dokumentů je vcelku složité. Pokud nemá uživatel finanční prostředky k zakoupení kvalitního datasetu, je nucen (za předpokladu, že blog neobsahuje cílový počet článků) získávat články z různých webů, čímž může dojít k vytvoření částečných, nebo úplných duplicit. Návod k použití enginu naleznete v souboru HOWTO.md

Závěr

Práce na projektu nám byla rozhodně přínosem, ať už při vyzkoušení si problematiky vyhledávacích algoritmů, scrapováním článků z internetu, nabití poznatků ohledně zpracování textu, nebo optimalizaci, tak při spolupráci v týmu.

Zdroje

Vzorec kosinove podobnosti
https://en.wikipedia.org/wiki/Cosine_similarity
Srovnání rychlosti parsování knihoven
http://www.ianbicking.org/blog/2008/03/python-html-parser-performance.html