Crawlovanie a extrakcia relevantných častí webových portálov

Rudolf Pavel

3Ib, 2016 - 2017

Abstrakt. Práca je súčasťou školského projektu kapsa. Zaoberá sa crawlovaním webových portálov internetových obchodov a následnou extrakciou dát z týchto portálov. Kľúčovou časťou práce je zabezpečiť extrakciu relevantných častí webových portálov za pomoci automatického orezávania prehľadávania webového portálu prostredníctvom analýzy úspešnosti predchádzajúcich vetiev.

Kľúčové slová: crawlovanie, extrakcia dát, relevantný, analýza prehľadávania, xPath, url

1 Úvod

Projekt kapsa je vedecký projekt, ktorého cieľom je získavanie, extrakcia a následné spracovávanie dát z webových portálov internetových obchodov. Projekt kapsa sa skladá z viacerých častí. Táto práca sa venuje hlavne časti získavania dát.

Crawlovanie je proces, pri ktorom crawler vyhľadáva všetky linky na webovej stránke. Tieto linky môžu smerovať na ľubovoľné iné stránky a môže ich byť nespočetne veľa. Zároveň môžu viesť na stránky, ktoré sme už predtým crawlovali. To sú javy ktorým sa snažíme predchádzať a vyhýbať sa im. Cieľom je nasledovať iba odkazy, ktoré vedú na časti stránok crawlovaného internetového obchodu, ktoré sa nazývajú detailové stránky produkov. Detailová stránka produktu je webová stránka na nejakom internetovom obchode, ktorá obsahuje špecifické detaily a údaje o danom produkte internetového obchodu.

Samotnému procesu crawlovania predchádza anotácia. Anotácia prebieha v prehliadači na datailovej stránke produktu internetového obchodu za pomoci anotačného nástroja. My využívame anotačný nástroj Exago. Na detailovej stránke produktu daného e-shopu sa výberú atribúty, pomocou ktorých sa pri crawlovaní určuje, ktorá stránka je a ktorá nie je detailová. Takisto rozhodovanie o tom, ktorá stránka je relevantná pre crawler a ktorá nie, prebieha za pomoci regulárneho výrazu, ktorý sa vytvorí počas anotácie na danom internetovom obchode. Na základe týchto dát sa vytvoria pravidlá, ktoré sa využijú pri samotnom crawlovaní. Následne sa z týchto pravidiel vytvorí wrapper, do ktorého sa tieto pravidla uložia. Tento wrapper je súbor typu JSON, ktorý sa uloží do databázy. A teda podľa pravidiel z wrappera sa pri samotnom crawlovaní extrahujú vybrané atribúty, ktoré sú ukladané do databázy.

2 Návrh riešenia

2.1 Proces crawlovania

V práci sa využíva pre potreby crawlovania web crawler **Crawler4j.** Je to open source web crawler, určený pre javu a umožňuje multi threaded riešenie crawlovania. Pred samotným crawlovaním sa z databázy načíta wrapper, ktorý obsahuje údaje a atribúty z anotácie detailovej stránky produktu. Pre potreby simulovania preklikov na webovej stránke, využívame nástroj **Selenium,** ktorý umožňuje integráciu s webovým prehľadávačom Firefox a parsovanie HTML kódu.

Bežný prístup pri crawlovaní je ten, že crawler nasleduje všetky linky, ktoré nevedú mimo stránok internetového obchodu. To však stále predstavuje veľkú množinu stránok, ktoré nie sú relevantné pre extrakciu dát, pretože nevedú na detailové stránky produktov. Takisto je možné dostať sa na tú istú detailovú stránku produktu rôznymi cestami, s rôznymi dĺžkami ciest prehľadávania. Strom prehľadávania v takomto prípade bude obsahovať viacero slepých vetiev. Je to z toho dôvodu, že každý internetový obchod obsahuje okrem detailových stránok produktov aj ostatné stránky, ako napríklad stránka prihlásenia alebo registrácie a podobne. Čo je samozrejme situácia, ktorá je nežiaduca a chceme jej predchádzať. Naším hlavným problémom je nájsť všetky detailové stránky produktov čo najefektívnejšie.

Hlavným cieľom je teda nájsť všetky detailové stránky produktov a zároveň minimalizovať počet tých ostatných. A teda vyhnúť sa neúspešným vetvám v strome prehľadávania, ale sústrediť sa na tie relevantné a zároveň nevynechávať a vyhľadávať nové úspešné vetvy v kategóriách a produktoch internetového obchodu. K tomu je potrebné vytvoriť algoritmus pre nájdenie a označenie týchto slepých vetiev.

2.2 Analýza prehľadávania

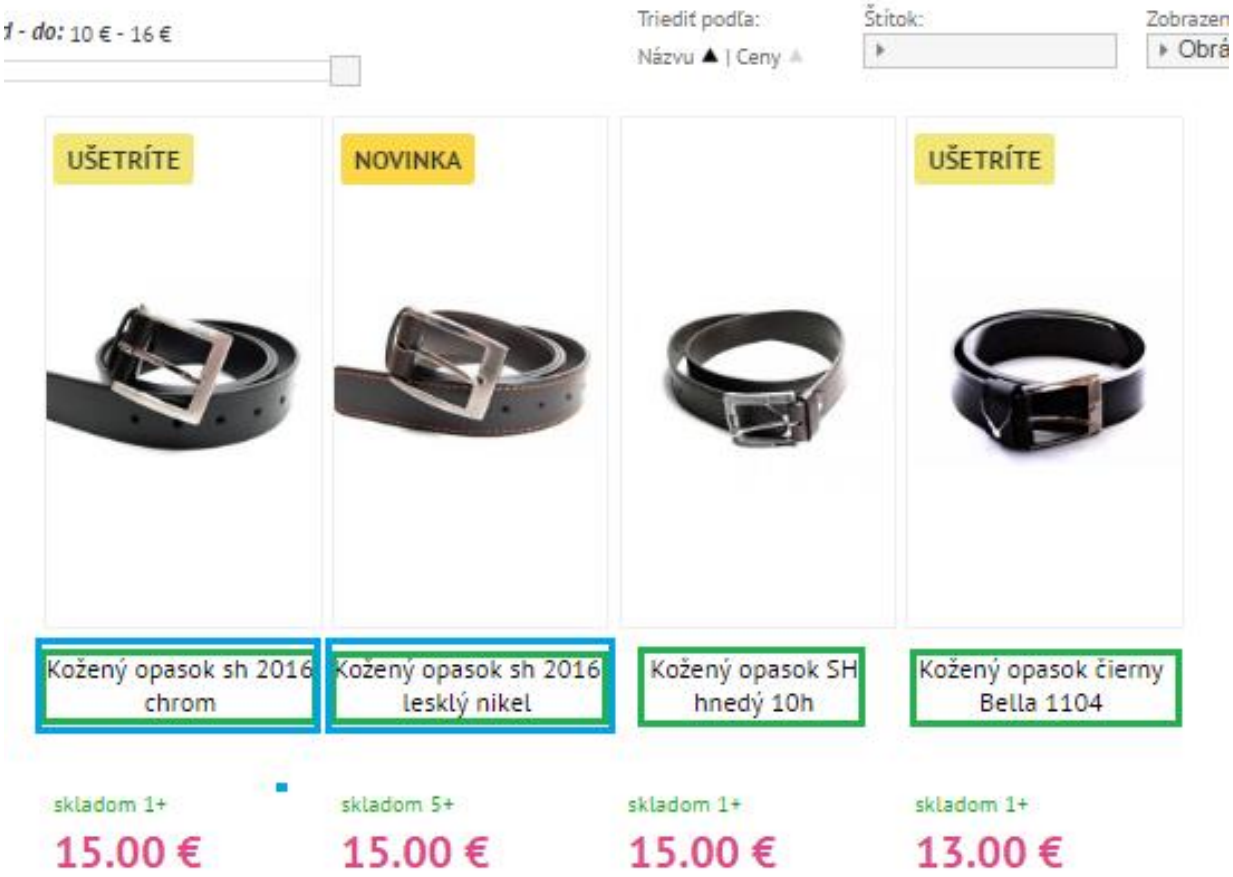
Aby sme mohli implementovať algoritmus nájdenia a označenia slepých vetiev, potrebujeme najprv poznať samotné dáta pri crawlovaní. Tieto dáta získame počas samotného procesu crawlovania. Zistíme kompletnú množinu liniek na danom internetovom obchode. Na základe tejto množiny crawlujeme jednotlivé webové stránky a získavame údaje ktoré sa ukladajú do databázovej tabuľky. Táto databázová tabuľka (**Tabuľka 1**.) obsahuje url adresu každej linky na danom internetovom obchode, ktorú crawlujeme. Ďalej obsahuje jednoznačný identifikátor rodiča danej url adresy v tejto tabuľke. Za pomoci tohto id vieme určiť rodiča pre danú url adresu. To využijeme pri rekonštrukcii stromu prehľadávania. Dôležitým údajom v tabuľke je xPath. Samotný xPath získavame za pomoci url adresy. V HTML kóde pomocou url adresy nájdeme element, ktorý obsahuje hľadanú linku. Následne z tohto elementu vygenerujeme xPath k nemu v HTML kóde. V stĺpci úspech sa nachádza boolean hodnota, do ktorej zapíšeme či sa jedná o úspešnú vetvu alebo nie.

Táto **tabuľka 1**. v reáli predstavuje kompletný strom prehľadávania pri crawlovani internetového obchodu. Teda obsahuje všetky url adresy, ktoré crawler spracovával. Na základe týchto údajov z tabuľky analyzujeme kompletný strom prehľadávania a zisťujeme, ktoré vetvy v strome sú úspešné. Čo znamená že vedú k detailovým stránkam produktov.

Tabuľka 1. Tabuľka s údajmi samotného crawlovania.

id	parent Id	url	xpath	úspech
1	0	http://penazenkyshop.sk/	null	true
2	1	http://penazenkyshop.sk/cart/	/body/div[2]/div[1]	false
3	1	http://penazenkyshop.sk/eshop/login	/body/div[2]/div[3]/div[2]	false
4	1	http://penazenkyshop.sk/opasky/	/body/div[2]	true
5	4	http://penazenkyshop.sk/kozene-opasky/c1022	/body/div[2]/div[4]/div[3]	true
6	5	http://penazenkyshop.sk/kozeny-opasok-sh-hnedy-10h/p665947c1023	/body/div[2]/div[4]/div[3]/a[1]	true
7	5	http://penazenkyshop.sk/kozeny-opasok-sh-2016-chrom/p666233c1024	/body/div[2]/div[4]/div[3]/a[2]	true

V **Tabuľke 1**. môžeme vidieť ukážku získaných údajov pri crawlovaní. V prvom riadku sa nachádza seed url, ktorá sa manuálne zadáva pre crawler. Na tejto adrese sa zaháji crawlovanie daného internetového obchodu. Keďže je počiatočná, nemá žiadneho rodiča ani xPath, teda je koreňom v strome prehľadávania. Ďalšie záznamy už predstavujú linky na danom internetovom obchode. Prvé dve url zvýraznené na červeno predstavujú linky pre košík a prihlásenie. Tieto linky teda nevedú priamo na detailové stránky produktov. To sú linky ktorým sa chceme vyhnúť označením za slepé vetvy. Ďalej v riadku 4 sa nachádza url adresa zo zoznamu kategórií. Teda po kliknutí na túto linku sa dostaneme do časti webovej stránky, kde sa nachádzajú podkategórie pre opasky. V danom prípade sa preklikneme na linku kožené opasky, ktorá zobrazí už len produkty z podkategórie kožené opasky.



Obrázok 1. Ukážka dát z tabuľky na reálnom internetovom obchode.

Na **obrázku 1**. sú zobrazené produkty z **tabuľky 1**. s id 6 a 7. Na základe analýzy xPathov, ktoré sú súčasťou úspešných vetiev vieme zistiť určité podobnosti. Pozorujeme že obe xPathy úspešných liniek z jednej stránky, ktoré na stránke patria do nejakého zoznamu produktov, sú si navzájom veľmi podobné. Obe xPathy majú na zeleno zvýraznené časti rovnaké, aj keď sú to dva rôzne produkty. Teda môžeme pozorovať, že xPath umožňuje adresovať nie iba konkrétny element, ale aj viacero elementov. Teda spoločná zelená časť xPathov pre 6. a 7. záznam v tabuľke adresuje jediným xPathom nielen oba elementy z **tabuľky 1**. ktoré sú na **obrázku 1**. zvýraznené modrou farbou , ale aj všetky ostatné linky v zozname opaskov na stránke. Takže všetky na zeleno zvýraznené elementy na **obrázku 1**. sú adresované jediným xPathom /body/div[2]/div[4]/div[3]/a.

Na základe týchto vlastností, chceme vytvoriť algoritmus na hľadanie elementov, ktoré sú si takýmto spôsobom podobné. A teda tieto elementy budú obsahovať linky, ktoré vedú na podobné stránky. Za pomoci tohto algoritmu budeme môcť rozoznať slepé vetvy. Samotný algoritmus je zatiaľ vo fáze návrhu.

Pri ďalšej iterácii crawlovania sa teda množina crawlovaných liniek bude skladať len s relevantných, ktoré vedú čo najefektívnejšie k produktovým stránkam produktov na internetovom obchode. A teda samotné vetvy, ktoré boli označené ako slepé, už nebudú v ďalších iteráciách používané pri crawlovaní.

3 Záver

Zatiaľ sa v práci podarilo implementovať analýzu klikov pri crawlovaní. Vytvorenie a naplnenie databázovej tabuľky, ktorá obsahuje údaje o jednotlivých preklikoch počas crawlovania. Podarilo sa implementovať aj získavanie xPathov z HTML pre jednotlivé prekliky a naindexovať databázovú tabuľku..

Ďalej je v pláne implementácia algoritmu na odhaľovanie slepých vetiev v strome prehľadávania. Vytvorenie stromovej štruktúry a načítavanie dát z databázovej tabuľky do danej štruktúry. Následne nás čaká implementovanie identifikácie produktov na viacerých stránkach. Ďalej zavedenie politness a paralelného prehľadávania. A v neposlednom rade otestovanie a vylepšenie funkcionality.

Poďakovanie. Ďakujem RNDr. Petrovi Gurskému, PhD., za cenné rady pri tvorbe práce.

Literatúra

1. Selenium Documentation. Dostupné na internete: http://www.seleniumhq.org/docs/

2. Súčasný stav v metodológiách pre poloautomatické získavanie dát zo služieb či stránok webu, ich anotácia a konverzia do štruktúrovanej podoby a mapovanie na objekty z aplikačnej domény. Finálna správa projektu CeZIS. Košice. 2015

3. Návrh a popis metód pre poloautomatické získavanie dát zo služieb či stránok webu, ich anotácia a konverzia do štruktúrovanej podoby a mapovanie na objekty z aplikačnej domény. Finálna správa projektu CeZIS. Košice. 2015