Fakulta informatiky a informačných technológií
Slovenskej technickej univerzity

Ilkovičova 2, 842 16 Karlova Ves

Mergovanie datasetov s mačkami

Autor:
AIS ID:

Motivácia

Existuje mnoho stránok s databázami mačiek, väčšina z nich sú však iba pre danú krajinu napríklad Rusko, Poľsko, Nórsko, Švédsko a mnoho ďalších. Ak by sme si chceli vyhľadať nejakú mačku, tak by sme potrebovali skúšať všetky tieto databázy pričom ani nevieme koľko všetkých takýchto databáz je. Bolo by oveľa jednoduchšie pre používateľa, keby boli všetky databázy spojené do jednej a dalo sa nad nimi vyhľadávať. Nastáva tu ale problém so zjednotením databáz nakoľko jedna mačka ak je predaná z napríklad Poľska do Ruska, tak sa bude nachádzať v obidvoch databázach, no jedna sa o jednu a tú istú mačku. Z pohľadu samostatných databáz je to v poriadku, avšak pri zjednotení databáz je potrebné túto zhodu nájsť a uložiť si danú mačku iba raz a vytiahnuť z dvoch záznamov čo najviac informácii, ak napríklad jedna z mačiek má informáciu, ktorá je pri druhej mačke nezadaná (napríklad chovnú stanicu).

Súčasné riešenia

Súčasne je najväčšou takouto databázou *pawpeds.com*. Táto databáza však nie je veľmi udržiavaná, zároveň sú dané mačky v rôznych formátoch, respektíve chýba farba, rasa, alebo iná informácia. Farby sú niektoré zapísané cez farebný kód, avšak nie všetky, niektoré sú popísané slovne a je potrebné teda tieto dáta vyčistiť. V dátach sa nachádzajú aj preklepy, ktoré je tak isto potrebné opraviť (napríklad rasa je napísaná ako RGA namiesto RAG a pod.). Nakoľko je potrebné aj stránky zlučovať, tak som ako druhú stránku použil ruskú databázu *tree.sibcat.info*. Zlučovanie datasetov nie je jednoduchý proces. Najväčší problém pri spájaní datasetov, ktoré majú prienik v rámci svojho obsahu je riešenie duplicít. Je potrebné určiť si na základe čoho budú data vyhlásené za duplicitné a ako bude s duplicitou naložené.

V článku *Schema matching using duplicates* sa venujú problematike spájania datasetov a identifikácii duplicít. Identifikáciu duplicít vysvetľujú na jednoduchom príklade, kedy síce 2 datasety sú v iných formátoch, ale našli prienik v niektorých atribútoch, ktorých podobnosť porovnávajú. Dané polia pred porovnaním premenia na malé písmena abecedy a následne vykonajú porovnanie. Pri hľadaní duplicít je možné robiť aj ďalšie operácie, ako napríklad počítanie editačnej vzdialenosti dvoch slov, alebo použiť podobnosť tokenov, kedy je slovo chápané ako „*bag of words*“ a [1].

Riešenie

Riešenie pozostáva zo 4-och častí crawlovanie, parsovanie, indexovanie a vyhľadávanie. Ako implementačné prostredie bolo zvolený Python s využitím frameworkov PySpark pre distribuované spracovanie a PyLucene pre vytvorenie indexu spolu s následným vyhľadávaním nad indexom.

Crawlovanie

Nakoľko sme nemali dáta vopred pripravené, bolo potrebné si ich nacrawlovať. Crawlovanie sme vykonali nad dvoma spomenutými stránkami *pawpeds.com* a *tree.sibcat.info* (neskôr referovaná ako ruská databáza). Stránky sme už počas crawlingu prvotne spracovali, teda sme si neukladali cele html stránky, ale extrahovali sme informácie o danej mačke. Pre extrakciu informácií sme použili knižnicu *re* pre prácu s regulárnymi výrazmi. Z pawpeds sa nám podarilo stiahnuť okolo jeden a pol milióna mačiek a z ruskej databázy sme stiahli okolo jedenásť tisíc mačiek.

Parsovanie

Po úspešnom crawlovaní sme sa pustili do spracovania získaných dát. Pri spracovaní dát z pawpeds nastalo mnoho problémov s nekonzistenciou údajov o jednej mačke. Údaje boli oddelené iba cez čiarku

bez akéhokoľvek popisu ktorý údaj je ktorý. Nedalo sa to určiť ani podľa poradia údajov, pretože niektoré mali iba 2 údaje napríklad meno a pohlavie, zatiaľ čo iné mali titul pred menom, meno, titul za menom, pohlavie, dátum narodenia a ďalšie informácie. Dáta bolo teda potrebné zanalyzovať a vytvoriť regexy pre zistenie o akú informáciu sa jedná. Po identifikovaní jednotlivých informácií nasledoval iný problém. Medzi informáciami bolo množstvo preklepov, ktoré bolo potrebné opraviť ručne, napríklad boli pri rase vymenené písmená, alebo boli štáty napísané miešane malými a veľkými písmenami, prípadne bol štát neznámi a boli raz zaznačený ako pomlčka, inde ako prázdny string a pod.

Parsovanie informácii z ruskej databázy nebolo problematické, údaje boli označené ktoré k čomu patria. Zmeny nad dátami boli tak isto minimalistické, bolo potrebné iba zmeniť formát dátumu, nakoľko mal prehodené dni a mesiace. Druhou zmenou bola zmena pohlaví z *male* na iba *M* a *female* na iba *F*.

Celý proces parsovania pawpeds prebiehal distribuovane za pomoci frameworku PySpark. Údaje sú načítane ako dataframe a následne prevedené do rdd formátu. Po prevedení nasleduje distribuované spracovanie, kedy ako prvé spracujeme jeden dokument o mačke extrahovaním všetkých potrebných informácií za pomoci už spomenutých regulárnych výrazov.

Spracovaním datasetov a prevedení ich do jednotného tvaru bolo možné tieto datasety zlúčiť. Rozhodli sme sa zlučovať tie mačky, ktoré majú rovnaké meno a zároveň majú aj rovnaké mená ich rodičia. Mená sme však neupravovali ani nijako nemenili, išlo čisto o porovnanie dvoch reťazcov a pri úplnej zhode nastalo zjednotenie.

Indexovanie

Indexovanie je implementované za pomoci frameworku PyLucene. Ako úložisko sme použili MMapDirectory a pre analýzu StandardAnalyzer v rámci IndexWritera. IndexWriter má zvolenú ako možnosť otvorenia CREATE, čo znamená, že pri každom spustení indexovania sa prepíše už existujúci index. Súbor spracovaných mačiek je uložený vo formáte, kde jeden riadok je jeden dokument, a teda sú aj takto po riadkoch načítavané a vytvárané dokumenty pre index. Pri indexovaní ukladáme iba ID mačky, ktoré ale neindexujeme. Preto sme použili nastavenia pre Field seStored(True) a IndexOptions.NONE. Do indexu vkladáme iba polia *name* a *breed*, nad ktorými chceme vedieť vyhľadávať, a nakoľko pre finálny výpis vyťahujeme informácie o danej mačke na základe jej ID z dictionary, tak *name* ani *breed* neukládame. Pre tieto polia sme použili možnosť TextField.TYPE_NOT_STORED, ktorý daný field indexuje, tokenizuje ale neuloží.

Vyhľadávanie

Pri vyhľadávaní sme použili IndexSearcher, ktorému sme dali priečinok, v ktorom sa index nachádza. Využili sme QueryParser pre spracovanie dopytu, kde sme ako analyzátor opäť zvolili StandardAnalyzer. Vďaka tomu môže užívateľ pri vyhľadávaní využiť QueryParser syntax a tak si upresniť svoj dopyt ako len chce. PyLucene používa ako základné vyhľadávanie cez OR, teda ak mu dáme viac Field-ov na vyhľadávanie, tak dostaneme viac výsledkov, preto ak chceme výber zúžiť je potrebné v QueryParser syntaxi pridať medzi Field-y *AND* operátor.

Popis dát

Základný súbor dát, ktorý je výstupom po crawlovaní je textový súbor, ktorý ma na jednom riadku jednu mačku. V prípade pawpeds je tento záznam vo formáte:

id mačky | informácia o mačke | tituly otca | id otca | informácie o otcovi | tituly matky | id matky | informácie o matke.

V prípade ruskej databázy, vieme už po crawlovaní rozdelené informácie o mačke, a tak aj tento záznam je vo formáte:

id mačky | meno mačky | zdrojová databáza | zdrojové id mačky | tituly pred menom | rasa | farebný kód | dátum narodenia | pohlavie |chovná stanica | id matky | id otca | meno matky | meno otca.

Po spracovaní parserom majú tieto dáta jednotný formát:

id mačky | meno mačky | zdrojová databáza | zdrojové id mačky | tituly pred menom | tituly za menom | rasa | farebný kód | dátum narodenia | pohlavie | krajina pôvodu | súčasná krajina | chovná stanica | id matky | id otca | meno matky | meno otca.

Väčšina pomenovaní sú samo vysvetľujúcich, akurát „id mačky“ nám hovorí o jej súčasnom id, ktoré je použité aj v referenciách na rodičov a ide sekvenčné od 1, zatiaľ čo „zdrojové id mačky“ referuje na id mačky v zdrojovej databáze, aby sme sa v prípade nezhody vedeli dopátrať k pôvodným dátam.

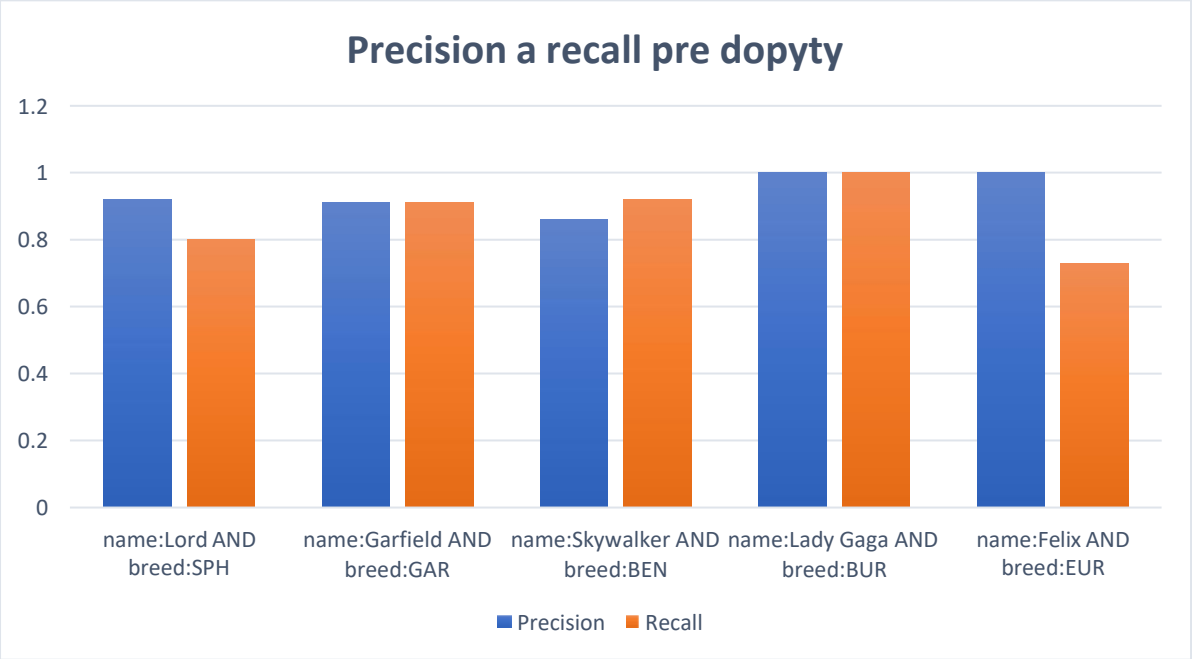
Vyhodnotenie

Pre evaluáciu sme použili ako referenčný model výsledky vyhľadávania na *pawpeds.com*, kde sme vyhľadávali iba nad jednou rasou. Pri vyhľadávaní sme brali všetky vrátené výsledky, ktoré sme následne porovnali so všetkými výsledkami vrátenými stránkou *pawpeds.com*. Ukážku vyhodnotenia si urobíme na dopyte „*name:Lord AND breed:SPH*“. Zvýraznené výsledky žltou farbou sú výsledky, ktoré nám náš vyhľadávač nevrátil. Výsledok tohto je, že slovo „*Lord“* sa nám nachádza ako podstring. V našom vyhľadávači však hľadanie podstringov nepodporujeme, a tak nedostaneme ani tieto výsledky. Šedo zvýraznený výsledok je chybný výsledok, nakoľko je to duplikát. Tento duplikát nebol počas čistenia odstránený, pretože sme porovnávali rovnaké mená rodičov a mačiek bez toho, aby sme vykonali nejaké zjednotenie na základe apostrofov a bodiek. Človek vie hneď na prvý pohľad vyhodnotiť, že sa jedná o tú istú mačku, no programu sa to pri takejto jednoduchej implementácii zjednocovania duplikátov odhaliť nepodarilo. Napriek tomu sa nám podarilo dosiahnuť pomerne vysokú hodnotu *precision* (0.92) a recall (0.8).

Pawpeds	**Náš projekt**
Apaws Good Lordy Miss Tortie	Bastetkatz Lord Vanderbilt
Bastetkatz Lord Vanderbilt	Bemisu Lord Bigglesworth
Bemisu Lord Bigglesworth	Softskins Lord Oolong
Britanya Lord E Im Naked	Malinkas Lord Vortigem
Kochav Ha-Carmel Lord of the Ring	Sayyads Lord O'Shea
Malinkas Lord Vortigem	Zoot'z Lord Loxley
Marlen Milord Le Grand Oray	Zoot'z Lord Lucan
Miki Sanukis de Lord Grand Animal	Zoot'z Lord Link
Sayyads Lord O'Shea	Britanya Lord E Im Naked
Sayyads Warlord	Britanya's Lord E. I'm Naked

Softskins Lord Oolong	Voila DJ Lord van Anubis
Voila DJ Lord van Anubis	Miki Sanukis de Lord Grand Animal
Zoot'z Lord Link	Kochav Ha-Carmel Lord of the Ring
Zoot'z Lord Loxley	
Zoot'z Lord Lucan	
	Precision: 12/13 = 0.92
	Recall: 12/15 = 0.80

Výsledky z ďalších vyhľadávaní a porovnania *precision* a *recall* môžeme vidieť v Tabuľke 1. Pri dopyte *„name:Lady Gaga AND breed:BUR“* sme dosiahli 100% zhodu, kedy nám našlo dokonca aj prehodené poradie Gaga Lady, ktorý sa vo výsledkoch nachádzal. Pri dopyte „name:Felix AND breed:EUR“ nám spadol recall až na 73%, pretože štvrtinu výsledkov tvorili mená kde bol Felix podreťazec a teda sa nášmu tieto výsledky nepodarilo nájsť.



Tabuľka 1

Používateľská príručka

Pre spustenie distribuovaného spracovania je potrebné spustiť script *„parsing.py“*, ktorý spustí distribuované spracovanie.

Pre spustenie indexovania a vyhľadávania je potrebné spustiť script *„final.py“*. Po spustení tohto scriptu sa zobrazí možnosť vytvoriť index (Obrázok 1.).

```
# python final.py
Do you want to create index? Type 1 for YES or 2 for NO
Your option: 1
```

Obrázok 1

Po vytvorení/nevytvorení indexu sa načítajú všetky mačky do dictionary pre finálne výpisy a používateľ môže začať svoje vyhľadávanie.

```
Loading cats...
Cats loaded, enjoy!

                You can search for cat by 'name' and by 'breed'. For that you need to use Query Parser syntax
                (e.g. 'name:Garfield breed:RAG'). Default search is above 'name' field.
                If you want to end program leave blank and press 'ENTER'.

Cat you are searching for: name:Garfield AND breed:RAG
```

Obrázok 2

Vyhľadávanie vráti prvých 20 výsledkov, pričom je možné si ďalšie výsledky zobraziť za pomoci stlačenia *ENTER*, alebo vrátiť sa na vyhľadávanie napísaním „exit“ (Obrázok 3).

```
For exit type 'exit', for more cats press ENTER: exit

                You can search for cat by 'name' and by 'breed'. For that you need to use Query Parser syntax
                (e.g. 'name:Garfield breed:RAG'). Default search is above 'name' field.
                If you want to end program leave blank and press 'ENTER'.

Cat you are searching for:
```

Obrázok 3

Výpis výsledkov vyhľadávania môžeme vidieť na obrázku 4.

```
|13| score: 4.402282238006592 | id: 84686 | name: Malmsjöstugans Garfield | breed: SBI | color: n | mother: Centauri's Cross My Heart | father: Qiri's Brown Focus |
|14| score: 4.402282238006592 | id: 91658 | name: Woodpecker Garfield | breed: MCO | color: ds 22 | mother: Woodpecker Allison | father: Nonsuch Geoffrey |
|15| score: 4.402282238006592 | id: 91800 | name: Granboets Garfield | breed: SBI | color: n | mother: Joyful Lucy-Lou av Ontario | father: Trampedals Mr Diversion |
|16| score: 4.402282238006592 | id: 98432 | name: Guldfakse's Garfield | breed: NFO | color: n 09 23 | mother: Tiger-Lilly | father: Røde Peer |
|17| score: 4.402282238006592 | id: 115414 | name: Brinzei Garfield | breed: MCO | color: a 09 22 | mother: Dreamhunter Big Surprise | father: El Tendido de Sol Manuel |
|18| score: 4.402282238006592 | id: 148367 | name: Supernova Garfield | breed: MCO | color: ds 22 | mother: Bosserne Puzzy | father: Kærgaarden's Lee Haley |
|19| score: 4.402282238006592 | id: 172583 | name: Oberon Garfield | breed: DRX | color: ds 09 21 33 | mother: Shaineh's E-Nocciola | father: Cartoon's Dixie King |
|20| score: 4.402282238006592 | id: 181666 | name: Baslatan Garfield | breed: MCO | color: n 09 22 | mother: Coonshine Sylvia | father: Coonshine Aquilla |
Number of matching documents in total: 20/396
```

Obrázok 4

V prípade, že výsledok obsahuje iba jednu mačku, tak sa zobrazí daná mačka tak, ako to môžeme vidieť na obrázku 5.

```
Cat you are searching for: name:"Baslatan Garfield" AND breed: MCO
| NAME: Baslatan Garfield                | ID: 181666                | BREED: MCO                |
| TITLE BEFORE NAME: GIC                 | TITLE AFTER NAME:         | COLOR: n 09 22            |
| MOTHER: Coonshine Sylvia               | FATHER: Coonshine Aquilla                             |
Number of matching documents in total: 1
```

Obrázok 5

Inštalácia softvéru

Pre správne fungovanie programu je potrebované mať nainštalovaný Python verzia 3.9+, PySpark, PyLucene. Zároveň je potrebné mať priložený súbor „*static_lists.py*“, z ktorého sa ťahajú statické zoznamy potrebné pre chod programu. Okrem *static_lists.py* je potrebný aj súbory „*cats_all.txt*“ a „*cats_russian.txt*“, ktoré slúžia ako vstup pre parsovanie vykonávané v programe „*parsing.py*“.

Výstupom parsovania je súbor „*cats_merged.csv*“, ktorý je potrebný ako vstup pre „*final.py*“ súbor, ktorý ho využíva na vytvorenie indexu. Poslednou požiadavkou pre správny chod je vytvorený index, ktorý je potrebný pre vyhľadávanie. Ak index neexistuje, je potrebné ho vytvoriť za pomoci scriptu „*final.py*“, ktorý sa na začiatku hneď po spustení spýta používateľa, či si želá vytvoriť index.

Súbory *cats_all.txt*, *cats_russian.txt cats_merged.csv* je možné nájsť na tomto linku na Google Drive:

https://drive.google.com/drive/folders/1CNr-zgjBXWxiAn6iVuFK_8nUyckEjhKj?usp=sharing

Zoznam použitej literatúry:

[1.] A. Bilke and F. Naumann, "Schema matching using duplicates," 21st International Conference on Data Engineering (ICDE'05), 2005, pp. 69-80, doi: 10.1109/ICDE.2005.126.