Data Mining: Proiect Default
Building (a part of) Watson

Sisteme Distribuite în Internet, Grupa 244
Echipă: Ateodoresi Denisa, Anton Claudia,
Back Andrei-Gheorghe

Scopul proiectului:

Sistemul Watson al IBM este un sistem de Răspuns la Întrebări (QA) care poate concura la nivelul campionilor umani în timp real în cadrul concursului TV de tip quiz, Jeopardy. Acesta, după cum vom vedea în clasă, este un demers complex. Totuși, răspunsurile la multe dintre întrebările din Jeopardy sunt de fapt titluri de pagini de Wikipedia. De exemplu, răspunsul la indiciul "Această femeie care a câștigat heptatloane consecutive la Olimpiadă a mers la UCLA cu o bursă de baschet" este "Jackie Joyner-Kersee", care are o pagină de Wikipedia cu același titlu: http://en.wikipedia.org/wiki/Jackie_Joyner-Kersee. În aceste situații, sarcina se reduce la clasificarea paginilor de Wikipedia, adică, găsirea paginii care este cel mai probabil răspuns la indiciul dat.

Cuprins

1. Descrierea codului

Proiectul este împărțit în 7 clase, acestea fiind descrise pe larg în cele ce urmează.

1. Lemmatizer

- Această clasă folosește librăria stanford.nlp pentru prelucrarea stringurilor primite ca date de intrare.
- Pentru realizarea prelucrării, stringurile sunt trecute prin următoarele trei procese: tokenizare, identificarea părților de vorbire (parts of speech), respectiv lematizarea.

2. Utils

- Această clasă conține funcții utilitare utilizate în: procesarea textului (normalizare și eliminarea caracterelor speciale), respectiv procesarea răspunsurilor.

3. IndexBuilder

- Această clasă are ca rol generarea indexului.
- Clasa pornește de la un set de date format din 80 de fișiere, fiecare dintre acestea conținând o suită de pagini Wikipedia. Ulterior, fiecare fișier este prelucrat, astfel încât sunt extrase articole, care sunt transformate în documente ale librăriei Apache Lucene și adăugate în index.
- La final procesului, indexul este salvat pentru utilizare ulterioară.

4. MetricsHelper

- Această clasă conține metode pentru calcularea metricilor de performanță, precum: P@K (în cazul nostru, K=1 și K=5) și MRR (Mean Reciprocal Rank).

5. SearchBuilder

- Această clasă se ocupă cu procesul de interogare asupra indexului creat anterior.
- Pentru fiecare dintre cele 100 de întrebări, se citesc categoria, întrebarea și răspunsul, apoi se creează o interogare care se aplică asupra indexului cu ajutorul clasei IndexSearcher din librăria Apache Lucene.
- Ulterior, se parcurg toate documentele găsite în urma interogării și se află rangul documentului care conține răspunsul întrebării.

6. ChatGPTQuestionBuilder

- Această clasă are rolul de a pregăti textul care va fi folosit ulterior în interogările adresate aplicației ChatGPT (pentru optimizarea de la punctul 4).

7. MetricsComparison

- Această clasă compară rezultatele obținute în urma interogării pe index cu rezultatele obținute în urma optimizării realizate folosind aplicația ChatGPT.
- Pentru aceasta, clasa folosește un fișier cu rezultatele date de ChatGPT, fișier care a fost completat manual cu răspunsurile oferite de aplicație.

2. Rezultate

Rezultatele obținute în urma interogării pe index au fost salvate în fișierul **index-results.txt**.

Formatul rezultatelor obținute este următorul:

```
<număr_întrebare>: <întrebare>
Obtained rank: <rang_obținut>
```

Un fragment din fișierul cu rezultate se poate observa în următoarea captură de ecran:

```
35:Title residence of Otter, Flounder, Pinto & Bluto in a 1978 comedy
Obtained rank: 1

36:Neurobiologist Amy Farrah Fowler on "The Big Bang Theory", in real life she has a Ph.D. in neuroscience from UCLA
Obtained rank: 1

37:In "The Deadlocked Election of 1800", James R. Sharp outlines the fall of this dueling vice president
Obtained rank: 1

38:He served in the KGB before becoming president & then prime minister of Russia
Obtained rank: 13

39:When asked to describe herself, she says first & foremost, she is Malia & Sasha's mom
Obtained rank: 2
```

3.a. Indexare

1. Descrieți cum ați pregătit termenii pentru indexare

Pentru a crește eficiența indexului, a fost nevoie ca textul primit să fie prelucrat. Pentru procesarea textului au fost efectuați următorii pași:

a) înlăturarea elementelor specifice Wikipedia (descrise în cadrul următoarei întrebări)
b) extragerea articolelor din fișiere, știind că fiecare dintre acestea au forma:

```
[[titlu]]

conținut
```

Pentru realizarea acestui pas folosim:

```
Matcher matcher = Pattern.compile( regex: "\\[\\[.*]]").matcher(data);
```

c) aplicarea procesului de normalizare asupra titlului și a conținutului fiecărui articol
- transformarea literelor în lower case

```
input = input.toLowerCase();
```

- eliminarea caracterelor speciale

```
input.replace( target: ".", replacement: "")
 .replace( target: ",", replacement: "")
 .replace( target: ":", replacement: "")
 .replace( target: ";", replacement: "");
```

- eliminarea spațiilor nefolositoare

```
input = input.strip();
```

- eliminarea cuvintelor de legătură (stop words) și transformarea în token-uri

```
StandardTokenizer tokenizer = new StandardTokenizer();
tokenizer.setReader(new StringReader(input));
TokenStream tokenStream = new StopFilter(tokenizer, EnglishAnalyzer.ENGLISH_STOP_WORDS_SET);

CharTermAttribute charTermAttribute = tokenStream.addAttribute(CharTermAttribute.class);
tokenStream.reset();
```

- lematizarea - pentru acest proces se folosește clasa specializată Lemmatizer

2. **Ce probleme specifice conținutului Wikipedia ați descoperit și cum le-ați abordat?**

În procesul de prelucrare al fișierelor, am întâlnit o serie de probleme, după cum urmează:

- Existența unor tag-uri specifice Wikipedia, și anume: `[[File:...]]`, `[[Image:...]]`. Apariția acestor tag-uri face imposibilă identificarea, în anumite cazuri, a titlului și a conținutului unor articole.
- În anumite cazuri, tag-urile specifice Wikipedia pot apărea fără să fie închise corespunzător, îngreunând, de asemenea, procesul de parsare.

Pentru rezolvarea acestor probleme, am utilizat o serie de expresii regulare pentru identificarea și eliminarea tag-urilor specifice.

Următoarea secțiune de cod a fost utilizată în scopul acestei soluții.

```
data = data.replaceAll( regex: "\\[\\[File:.*]]]]", replacement: "");
data = data.replaceAll( regex: "\\[\\[File:.*]]", replacement: "");

data = data.replaceAll( regex: "\\[\\[Image:.*]]]]", replacement: "");
data = data.replaceAll( regex: "\\[\\[Image:.*]]", replacement: "");
```

3. **Descrieți cum ați construit interogarea din indiciu**

Am observat că rezultatele obținute sunt mai bune în momentul în care, pentru realizarea interogării, folosim atât categoria, cât și indiciul (clue).

Pentru realizarea textului interogării, am alipit categoria și indiciul, ulterior aplicând normalizarea asupra stringului obținut.

```
query(parser, searcher, clue: category.trim() + " " + clue, directoryReader.maxDoc());
```

3.b. Măsurarea performanței

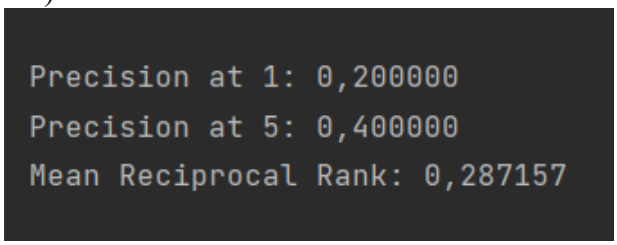
În cadrul proiectului nostru, am folosit următoarele metrici de performanță:

- **P@1 (precision at 1)** - această metrică indică numărul de întrebări pentru care răspunsul corect s-a aflat pe prima poziție. Acest tip de metrică este relevant în contextul unui joc de tip Jeopardy, deoarece dorim să obținem răspunsul corect pentru fiecare întrebare de cultură generală.

- **P@5 (precision at 5)** - această metrică indică numărul de întrebări pentru care răspunsul corect s-a aflat în primele 5 poziții. Această metrică este relevantă în contextul în care vrem să folosim un element de Machine Learning pentru îmbunătățirea rezultatelor. Astfel de sisteme (precum ChatGPT) au o performanță mai ridicată pentru volume mai mici de date.

- **MRR (Mean Reciprocal Rank)** - această metrică se concentrează pe rangul primului răspuns relevant din cadrul unei liste cu răspunsuri returnate de sistem. În contextul unui concurs de tip Jeopardy, în care scopul este obținerea răspunsului corect pentru fiecare întrebare, capacitatea unui sistem de a returna cel mai bun răspuns este crucială. Metrica MRR măsoară exact această capacitate, făcând-o esențială pentru determinarea eficienței sistemului nostru.

Rezultatele obținute de sistemul nostru prin interogarea inițială a indexului (fără vreo îmbunătățire utilizând ML) sunt următoarele:



3.c. Analiza erorilor

1. **La câte întrebări s-a răspuns corect/incorect?**

Interogările s-au efectuat pentru 100 de întrebări, iar răspunsul corect s-a obținut la 20 dintre acestea. De asemenea, răspunsul corect s-a aflat în primele 5 articole returnate de către interogare pentru 20 dintre cele 80 de întrebări greșite.

2. **De ce crezi că la întrebările corecte se poate răspunde printr-un sistem atât de simplu?**

Însăși structura pe care o au întrebările concursului Jeopardy reprezintă unul dintre motivele pentru care un sistem atât de simplu poate da răspunsuri corecte. Aceste întrebări reprezintă, în multe cazuri, chiar titlul sau primele fraze din articolele Wikipedia. Astfel că un sistem de tipul Information Retrieval poate să găsească cu ușurință răspunsurile corecte din articolele puse la dispoziție.

Un astfel de exemplu a fost identificat în întrebarea:

This woman who won consecutive heptathlons at the Olympics
went to UCLA on a basketball scholarship

Răspunsul acestei întrebări - **Jackie Joyner-Kersee** - se găsește chiar în prima frază a articolului, articol care are ca titlu chiar numele sportivei.

3. **Ce probleme observați pentru întrebările la care s-a răspuns incorect?**

În urma analizei întrebărilor la care s-a răspuns incorect, am identificat mai multe posibile cauze ale acestor erori. Acestea vor fi descrise pe larg în cele ce urmează, fiind însoțite de exemple.

a) Eliminarea caracterelor speciale poate face ca înțelesul unei fraze să se piardă. Un astfel de exemplu sunt citările, a căror ghilimele sunt eliminate în procesul de normalizare.

O întrebare identificată de noi care intră în această categorie este:

1980: "Rock With You"

În cadrul acestei întrebări, ghilimelele indică titlul unei melodii, iar în lipsa acestora, sistemul pune accentul pe genul muzical rock din cadrul întrebării, astfel returnând, ca prim răspuns, numele unui album rock: **time burn giant album**. Întrebarea presupunea recunoașterea autorului piesei aflate între ghilimele, și anume **Michael Jackson**.

b) Uneori, sistemul poate să interpreteze greșit întrebarea datorită unor ambiguități, a unor construcții complexe sau a unor nuanțe specifice. Astfel, în cazul în care sistemul nu reușește să recunoască elemente cheie care ar trebui identificate în același context, atunci documentele care abordează aceste aspecte împreună nu vor fi prioritizate.

O întrebare ce exemplifică acest aspect este următoarea:

He served in the KGB before becoming president & then prime minister of Russia.

Deși răspunsul acestei întrebări este **Vladimir Putin**, sistemul a considerat separat elementele KGB, president și prime minister, returnând răspunsuri precum: **Vladimir Alganov**, care a fost spion KGB, și **Keizō Obuchi**, care a fost prim-ministru al Japoniei.

c) În anumite cazuri, un grad ridicat de ambiguitate a întrebării poate îngreuna găsirea unui răspuns potrivit. La această ambiguitate se adaugă, uneori, și completări ale indiciului.

Un astfel de exemplu este întrebarea:

The Naples Museum of Art

unde categoria și completarea indiciului sunt: **STATE OF THE ART MUSEUM (Alex: We'll give you the museum. You give us the state.)**. În acest caz, răspunsul dorit este **Florida**, însă sistemul returnează drept răspuns un alt muzeu de arte, și anume **maryhill museum art**.

d) În cadrul frazelor mai lungi, unde logica devine mai complicată, iar fraza e împărțită în mai multe propoziții, sistemul identifică doar componente ale frazei, fără a vedea imaginea de ansamblu.

Întrebare exemplificativă pentru această situație este:

The Royal Palace grounds feature a statue of King Norodom, who in the late 1800s was compelled to first put his country under the control of this European power; of course, it was sculpted in that country

a cărei categorie este: **CAMBODIAN HISTORY & CULTURE**. Întrebarea are drept răspuns **Franța**, însă sistemul se concentrează mai mult pe categorie și returnează răspunsuri legate de politica Cambodgiei: **politics cambodium** și **norodom buppha devi** (fost ministru).

3.d. Îmbunătățirea extragerii (retrieval)

După cum s-a menționat anterior, în urma interogării asupra indexului, 20 de întrebări dintre cele 100 au dat răspunsul corect (acesta s-a aflat pe prima poziție). Pe lângă acestea, am observat că, pentru alte 20 de întrebări, răspunsul corect s-a aflat în primele 5 articole returnate de sistem.

Pentru jocul Jeopardy, idealul este să obținem răspunsul corect pe prima poziția, scopul final fiind de a atinge un scor cât mai mare. Așadar, ne-am gândit să încercăm o metodă de optimizare pentru cele 20 de întrebări cu răspunsurile în top 5, în speranța de a îmbunătăți scorul anterior.

Tehnologia aleasă pentru această optimizare este ChatGPT (GPT 4.0). Pentru fiecare dintre cele 20 de întrebări am generat un string pentru interogările ulterioare. În acest string am solicitat reordonarea fișierelor date ca input, astfel încât răspunsul corect să poată fi identificat, oferind și indiciul pentru acesta.

Pentru fiecare întrebare s-a generat un fișier corespunzător câte unui articol din cele top 5. Ulterior, interogările ChatGPT au fost făcute manual, folosind ca text string-urile generate anterior, alături de fișierele corespunzătoare fiecărei întrebări.

Interogările, respectiv răspunsurile oferite de ChatGPT pot fi consultate accesând următorul link: https://chat.openai.com/share/9dcd7be8-1a04-459a-a27a-f04b5ea85a39. Acestea pot fi găsite și în proiectul Git, în directorul **chatGPT_results**.

1. **Care este performanța sistemului dumneavoastră după această îmbunătățire?**

Răspunsurile oferite de ChatGPT au fost extrase într-un fișier text. După ce le-am analizat, am observat că ChatGPT a reușit să plaseze pe primul loc răspunsul corect la toate întrebările date.

Astfel, rezultatele obținute de sistemul nostru prin interogarea inițială a indexului alături de optimizarea cu ChatGPT sunt următoarele:

```
Precision at 1: 0,400000
Precision at 5: 0,400000
```