Data Mining Project

IBM's Watson Question Answering System

Sisteme distribuite în Internet, Grupa 244

Diaconu Horia
Moldovan Bogdana
Filipaș Răzvan

Importanța proiectului

Proiectul are o importanță semnificativă în contextul mining-ului de date și cercetării informaționale. Prin indexarea eficientă a conținutului Wikipedia, acest proiect facilitează accesul rapid și structurat la informații extinse, permițând utilizatorilor să efectueze căutări și să recupereze pagini relevante în funcție de interogările lor.

Prin participarea la acest proiect, studenții au șansa de a lucra direct cu concepte și tehnologii relevante, precum indexarea și căutarea eficientă a informațiilor. Proiectul nu doar consolidează înțelegerea teoretică, ci oferă și experiență practică în gestionarea seturilor mari de date, extinzându-ne perspectiva asupra aplicațiilor practice ale conceptelor academice în lumea reală a cercetării și dezvoltării software.

Ce tehnologii am utilizat? De ce?

Pentru realizarea proiectului nostru, am ales să utilizăm limbajul de programare Python datorită versatilității sale și a bogatei colecții de biblioteci disponibile pentru prelucrarea textului și manipularea datelor.

Pentru a gestiona eficient căutarea și indexarea textului, am ales să folosim biblioteca Whoosh, care furnizează un cadru puternic pentru construirea și gestionarea indexării. Whoosh oferă facilități avansate pentru căutarea textului, inclusiv opțiuni de interogare flexibile și gestionarea eficientă a indexului.

Alegerea acestor două tehnologii ne-a permis să implementăm rapid și eficient funcționalitățile de indexare și căutare, contribuind la dezvoltarea unei soluții eficiente și ușor de dezvoltat pentru proiectul nostru.

Cum am realizat indexarea?

Am implementat un proces de indexare a datelor Wikipedia dintr-un director specificat. În prima parte, am definit un schema pentru index folosind Whoosh, care include un câmp pentru titlu și un câmp pentru conținut (text). Funcția `create_index` verifică dacă directorul de index există și, în caz contrar, îl creează.

În funcția `index_wikipedia_data`, am parcurs fișierele din directorul specificat `data_folder`, fiecare conținând informații despre articolele Wikipedia. Am divizat conținutul fiecărui fișier în articole distincte folosind separatorul "[[". Apoi, pentru fiecare articol, am extrase titlul și conținutul și le-am adăugat în index folosind obiectul `writer` asociat. Acest proces se repetă pentru toate fișierele din directorul dat.

În final, apelăm `writer.commit()` pentru a finaliza procesul de indexare. Astfel, indexul rezultat conține titlurile și conținuturile articolelor Wikipedia, pregătit pentru a fi utilizat în operațiuni de căutare ulterioare. Această metodă de indexare ne permite să organizăm și să accesăm rapid informațiile din colecția de date, facilitând căutarea și recuperarea eficientă a informațiilor relevante în cadrul proiectului nostru de minerit de date.

```python
from whoosh.index import create_in
from whoosh.fields import TEXT, ID, Schema
import os

def create_index(schema, index_dir):
    if not os.path.exists(index_dir):
        os.mkdir(index_dir)
    index = create_in(index_dir, schema)
    return index

def index_wikipedia_data(index, data_folder):
    writer = index.writer()
    k = 0
    for file_name in os.listdir(data_folder):
        k = k + 1
        print(f"Processing file: {file_name}")
        print(k)
        with open(os.path.join(data_folder, file_name), 'r', encoding='latin-1') as file:
            content = file.read()
            articles = content.split('[[')[1:]  # Split the content into articles
            for article in articles:
                title_end = article.find(']]')
                title = article[:title_end].strip()
                text = article[title_end + 2:].strip()
                writer.add_document(title=title, content=text)

    writer.commit()

if __name__ == "__main__":
    schema = Schema(title=ID(stored=True), content=TEXT)
    index_dir = "index"
    data_folder = "wikipediaDataMining"

    index = create_index(schema, index_dir)
    index_wikipedia_data(index, data_folder)
```

În procesul de evaluare a performanțelor sistemului nostru, am ales să ne focalizăm pe metrica Precision at 1 (P@1) pentru a obține o evaluare detaliată a eficacității căutării. P@1 este o măsură specifică, concentrată asupra acurateții primului rezultat returnat de către sistem în răspuns la o interogare. Alegerea acestei metrici a fost strategică în contextul proiectului nostru, unde relevanța primului rezultat este deosebit de importantă pentru utilizatori.

Această perspectivă detaliată asupra eficacității căutării ne-a permis să identificăm și să aducem îmbunătățiri, în vederea maximizării acurateței și relevanței primului rezultat furnizat de către sistem. În esență, folosirea metricii P@1 a oferit o transparență asupra performanței noastre într-un mod specific la nevoile proiectului nostru, contribuind astfel la dezvoltarea continuă a sistemului pentru a îndeplini cerințele utilizatorilor.

Analiza erorilor

Rezultate obținute

În procesul nostru de analiză, am constatat că, pe setul de date furnizat de problema inițială, am obținut o precizie concretă de doar 4%, semnalând astfel o performanță suboptimală în identificarea răspunsurilor corecte. În plus, am observat o tendință de furnizare a unor răspunsuri din categoria corectă, chiar dacă acestea nu corespundeau întotdeauna cu răspunsurile corecte.

Cu toate acestea, prin dezvoltarea și implementarea propriului set de date, am obținut îmbunătățiri semnificative. Astfel, pe noul nostru set de date, am reușit să răspundem corect la 14 din 20 de întrebări, atingând astfel o precizie notabilă de 70%. Această adaptabilitate și capacitate de îmbunătățire evidențiază importanța dezvoltării și evaluării sistemelor pe seturi de date personalizate pentru a asigura o performanță optimă în contextul specific al problemei abordate.

Vă mulțumim pentru atenție!