Πανεπιστήμιο Δυτικής Αττικής
Σχολή Μηχανικών
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

Ανάκτηση Πληροφορίας

ΘΩΜΑΣ ΝΙΚΟΛΑΟΣ - ΑΜ: 21390068
ΠΑΠΑΓΕΩΡΓΙΟΥ ΦΙΛΙΠΠΟΣ - ΑΜ: 21390174

GitHub

ΑΘΗΝΑ
Τρίτη, 5 Δεκεμβρίου 2023

Περιεχόμενα:

Εισαγωγή.

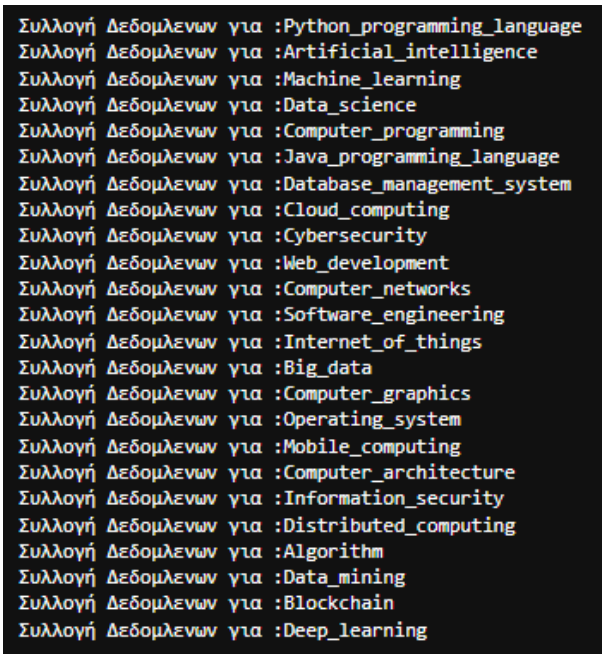
Στην παρούσα εργασία, υλοποιήσαμε μία μηχανή αναζήτησης που ανακτά έγγραφα από το Wikipedia με βάση τα ερωτήματα των χρηστών (για ευκολία τα συγκεκριμένα ερωτήματα γίνονται απευθείας στον κώδικα).

Στόχοι της εργασίας:

- Σχεδιασμός και ανάπτυξη ενός αποδοτικού συστήματος για την ευρετηρίαση και αναζήτηση εγγράφων κειμένου.
- Υλοποίηση και αξιολόγηση αλγορίθμων ανάκτησης χρησιμοποιώντας μετρικές όπως ακρίβεια, ανάκληση και F1-score.
- Δημιουργία φιλικής διεπαφής για εύκολη αναζήτηση και ανάκτηση εγγράφων από τους χρήστες.
- Απόκτηση πρακτικής εμπειρίας σε τεχνικές ανάκτησης πληροφοριών, όπως Boolean retrieval, Vector Space Model και Probabilistic retrieval models.

1. Συλλογή δεδομένων.

Για την συλλογή των δεδομένων από το Wikipedia, υλοποιήθηκε ένας web crawler βασισμένος στις βιβλιοθήκες BeautifulSoup και requests. Για αρχή, ο κώδικας χειρίζεται τα HTTP αιτήματα, αναλύει της HTML σελίδες και εξάγει το περιεχόμενο των άρθρων. Στην συνέχεια, τα αποθηκεύει στο “webpage_data.json”.



Εικόνα 1: Συλλογή δεδομένων.

2. Προεπεξεργασία κειμένου (Text Processing).

1. Διάσπαση των κειμένων σε ξεχωριστές λέξεις (tokenization) - word_tokenize.
2. Αφαίρεση κοινών λέξεων (stop-word removal) - stopwords.

3. Αναγωγή των λέξεων στη βασική τους μορφή (Stemming/Lemmatization) – WordNetLemmatizer.
4. Αφαίρεση ειδικών χαρακτήρων με κανονικές εκφράσεις – regex.

3. Ευρετήριο (Indexing).

Δημιουργήθηκε μία ανεστραμμένη δομή δεδομένων που αντιστοιχεί στις λέξεις-κλειδιά στα έγγραφα από το Wikipedia στα οποία εμφανίζονται. Το ευρετήριο αποθηκεύεται σε αρχεία JSON επιτρέποντας εύκολη ανάγνωση, εγγραφή και ανταλλαγή δεδομένων. Το JSON επιλέχθηκε για την απλότητά του.

4. Μηχανή αναζήτησης (Search Engine).

Υλοποιήθηκε μία διεπαφή γραμμής εντολών για την αναζήτηση όρων χρησιμοποιώντας την Python.

a) Για την επεξεργασία των ερωτημάτων, υλοποιήθηκαν λειτουργίες Boolean (AND, OR, NOT).
b) Για την κατάταξη των αποτελεσμάτων χρησιμοποιήθηκαν οι αλγόριθμοι TF-IDF (Term Frequency-Inverse Document Frequency) και Okapi BM25 (για πιο ακριβή κατάταξη αποτελεσμάτων)

Επιπλέον, υλοποιήθηκε το VSM για τη μέτρηση της ομοιότητας μεταξύ ερωτήματος και εγγράφων.

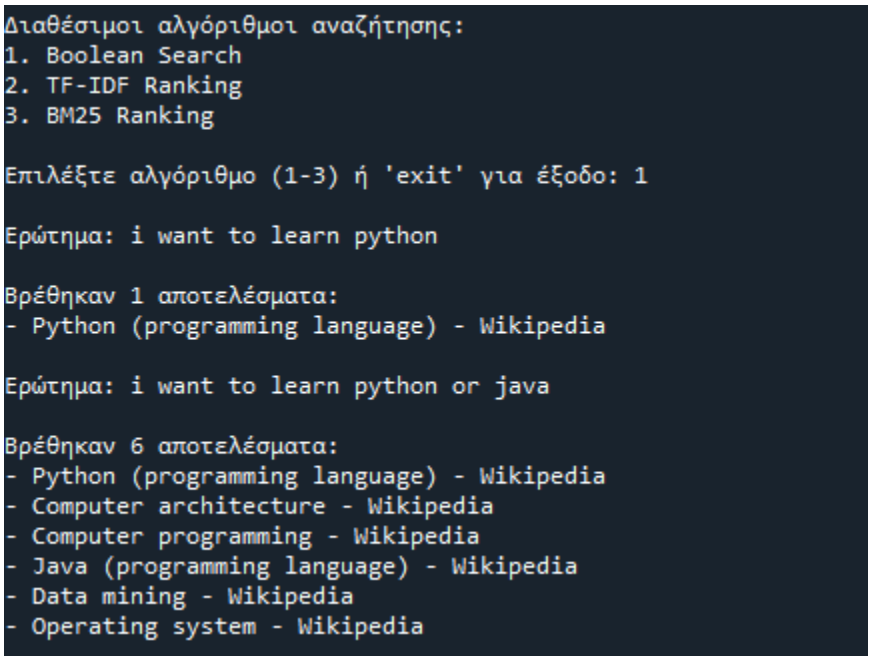
Σχεδιασμός της Βασικής Δομής

- Δημιουργήθηκε η κλάση RankingEngine που διαχειρίζεται όλους τους αλγόριθμους κατάταξης
- Χρησιμοποιήθηκε ένα ανεστραμμένο ευρετήριο (inverted index) ως βασική δομή δεδομένων

- Υπολογίζονται και αποθηκεύονται τα μήκη των εγγράφων για αποδοτικότερη επεξεργασία

Υλοποίηση Βασικών Μετρικών:

- Term Frequency (TF): Υπολογίζεται ως η συχνότητα εμφάνισης ενός όρου σε ένα έγγραφο, κανονικοποιημένη ως προς το μήκος του εγγράφου
- Inverse Document Frequency (IDF): Υπολογίζεται χρησιμοποιώντας τον λογαριθμικό τύπο log(N/(df + 1)), όπου N είναι ο συνολικός αριθμός εγγράφων και df ο αριθμός εγγράφων που περιέχουν τον όρο.



Εικόνα 2: Εκτέλεση Boolean Search 1.

```
Ερώτημα: learn python or react

Βρέθηκαν 6 αποτελέσματα:
- Python (programming language) - Wikipedia
- Artificial intelligence - Wikipedia
- Machine learning - Wikipedia
- Computer security - Wikipedia
- Web development - Wikipedia
- Operating system - Wikipedia

Ερώτημα: python not react

Βρέθηκαν 2 αποτελέσματα:
- Python (programming language) - Wikipedia
- Artificial intelligence - Wikipedia
```

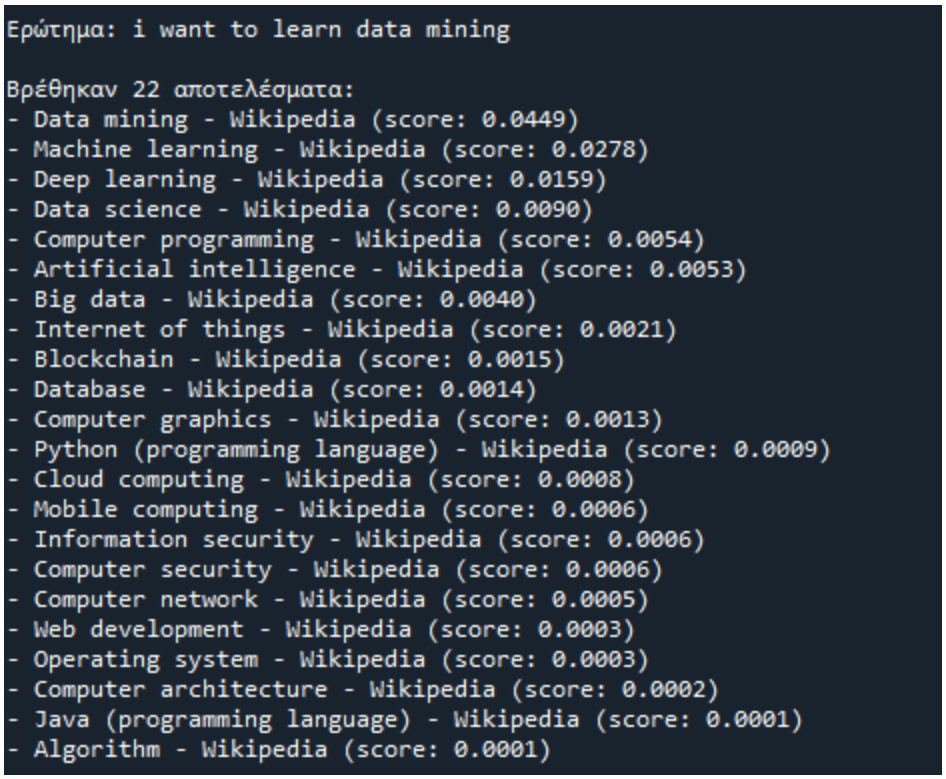
Εικόνα 3: Εκτέλεση Boolean Search 2.

```
Επιλέξτε αλγόριθμο (1-3) ή 'exit' για έξοδο: 2

Ερώτημα: web sockets

Βρέθηκαν 18 αποτελέσματα:
- Web development - Wikipedia (score: 0.0104)
- Java (programming language) - Wikipedia (score: 0.0011)
- Computer network - Wikipedia (score: 0.0007)
- Computer security - Wikipedia (score: 0.0007)
- Python (programming language) - Wikipedia (score: 0.0005)
- Mobile computing - Wikipedia (score: 0.0005)
- Internet of things - Wikipedia (score: 0.0002)
- Cloud computing - Wikipedia (score: 0.0002)
- Data mining - Wikipedia (score: 0.0001)
- Database - Wikipedia (score: 0.0001)
- Computer programming - Wikipedia (score: 0.0001)
- Computer graphics - Wikipedia (score: 0.0001)
- Deep learning - Wikipedia (score: 0.0001)
- Operating system - Wikipedia (score: 0.0001)
- Information security - Wikipedia (score: 0.0001)
- Big data - Wikipedia (score: 0.0001)
- Machine learning - Wikipedia (score: 0.0000)
- Artificial intelligence - Wikipedia (score: 0.0000)
```

Εικόνα 4: Εκτέλεση TF-IDF Ranking 1.



Εικόνα 5: Εκτέλεση TF-IDF Ranking 2.

```
Ερώτημα: what a computer do

Βρέθηκαν 23 αποτελέσματα:
- Computer graphics - Wikipedia (score: 0.1522)
- Computer architecture - Wikipedia (score: 0.1517)
- Computer security - Wikipedia (score: 0.1491)
- Distributed computing - Wikipedia (score: 0.1490)
- Software engineering - Wikipedia (score: 0.1489)
- Computer programming - Wikipedia (score: 0.1486)
- Computer network - Wikipedia (score: 0.1472)
- Operating system - Wikipedia (score: 0.1471)
- Algorithm - Wikipedia (score: 0.1400)
- Information security - Wikipedia (score: 0.1380)
- Database - Wikipedia (score: 0.1366)
- Data science - Wikipedia (score: 0.1341)
- Deep learning - Wikipedia (score: 0.1324)
- Artificial intelligence - Wikipedia (score: 0.1314)
- Data mining - Wikipedia (score: 0.1305)
- Machine learning - Wikipedia (score: 0.1297)
- Internet of things - Wikipedia (score: 0.1283)
- Blockchain - Wikipedia (score: 0.1254)
- Big data - Wikipedia (score: 0.1231)
- Mobile computing - Wikipedia (score: 0.1219)
- Python (programming language) - Wikipedia (score: 0.1159)
- Cloud computing - Wikipedia (score: 0.1056)
- Java (programming language) - Wikipedia (score: 0.0722)
```

Εικόνα 6: Εκτέλεση TF-IDF Ranking 3.

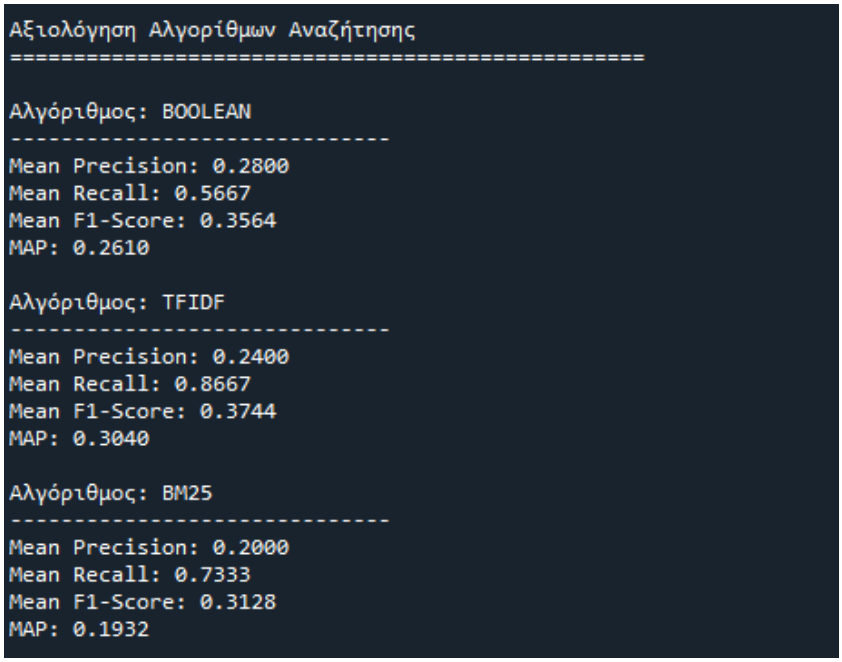
```
Ερώτημα: smart contracts

Βρέθηκαν 7 αποτελέσματα:
- Blockchain - Wikipedia (score: 6.7817)
- Computer security - Wikipedia (score: 4.4733)
- Internet of things - Wikipedia (score: 3.5820)
- Information security - Wikipedia (score: 2.3999)
- Web development - Wikipedia (score: 2.2644)
- Operating system - Wikipedia (score: 2.1970)
- Java (programming language) - Wikipedia (score: 1.7663)
```

Εικόνα 7: Εκτέλεση BM25 Ranking.

5. Αξιολόγηση συστήματος.

Για την αξιολόγηση του συστήματος, χρησιμοποιήθηκαν ακρίβεια (precision), ανάκληση (recall), F1-score και MAP (Mean Average Precision). Όπως επίσης, δημιουργήθηκε ένα σύνολο ερωτημάτων για τον έλεγχο της απόδοσης του. Αναλυτικότερα στην εργασία, υλοποιήσαμε ένα ολοκληρωμένο σύστημα αξιολόγησης μηχανής αναζήτησης μέσω της κλάσης SearchEvaluator, η οποία ενσωματώνει ένα σύνολο προκαθορισμένων ερωτημάτων ελέγχου για την αξιολόγηση της απόδοσης. Χρησιμοποιήθηκαν τέσσερις βασικές μετρικές αξιολόγησης: η ακρίβεια (precision) που μετρά το ποσοστό των σχετικών εγγράφων στα αποτελέσματα, η ανάκληση (recall) που υπολογίζει το ποσοστό των σχετικών εγγράφων που ανακτήθηκαν, το F1-score που συνδυάζει τις δύο προηγούμενες μετρικές, και το Mean Average Precision (MAP) που αξιολογεί την ποιότητα της κατάταξης των αποτελεσμάτων. Το σύστημα υποστηρίζει τη σύγκριση τριών διαφορετικών αλγορίθμων κατάταξης (Boolean, TF-IDF και BM25), παρέχοντας λεπτομερή αποτελέσματα για κάθε αλγόριθμο. Η υλοποίηση έγινε με έμφαση στην αποδοτικότητα, χρησιμοποιώντας κατάλληλες δομές δεδομένων όπως sets για γρήγορους υπολογισμούς, ενώ παράλληλα δόθηκε προσοχή στην επεκτασιμότητα του κώδικα μέσω καλά τεκμηριωμένων μεθόδων και ευέλικτης αρχιτεκτονικής που επιτρέπει την εύκολη προσθήκη νέων μετρικών και αλγορίθμων στο μέλλον.



Εικόνα 8: Αξιολόγηση αλγορίθμων αναζήτησης.

6. Αναφορά και τεκμηρίωση.

Στην εργασία υλοποιήσαμε ένα ολοκληρωμένο σύστημα συλλογής και επεξεργασίας δεδομένων από το Wikipedia, το οποίο η δομή του ξεκινά από την κλάση WikipediaCrawler για την συλλογή των δεδομένων. Στην συνέχεια υλοποιήσαμε την κλάση DataStorage για την διαχείρηση και αποθήκευση δεδομένων και την InvertedIndex όπως αναφέρθηκε και προηγουμένως. Το σύστημα χρησιμοποιεί προηγμένες τεχνικές επεξεργασίας φυσικής γλώσας (NLP) μέσω της βιβλιοθήκης NLTK. Επιπλέον, υλοποιήθηκε ένα αποτελεσματικό σύστημα αποθήκευσης σε μορφή JSON και ένα ανεστραμμένο ευρετήριο που καταγράφει λεπτομερείς πληροφορίες για κάθε όρο, συμπεριλαμβανομένης της συχνότητας εμφάνισης και των εγγράφων στα οποία εμφανίζεται. Για αλγορίθμους αναζήτησης δημιουργήθηκαν οι κλάσεις SearchEngine η οποία υλοποποιέι το boolean search και την RankingEngine η οποία υποστηρίζει των TF-IDF ΚΑΙ BM25 που εχούν αναλύθει προηγούμενος. Τέλος βάση των παραδειγμάτων που εχούμε συνάψει , παρατήρουμε πως εχούμε άναπτυξει ένα σύστημα μήχανης άναζητησης.

7. Βελτιώσεις και Δυσκολίες.

- Βελτιστοποίηση ταχύτητας αναζήτησης.
- Καλύτερα αποτελέσματα στους αλγόριθμους αναζήτησης
- Κατανοητή δομή του κώδικα , για να είναι εύκολη στην συντήρηση και την ανάπτυξη και από τα δύο μέλη της ομάδας
- Ξεκάθαρος διαχωρισμός ευθυνών μεταξύ των κλάσεων
- Διαχείριση άκυρων ερωτημάτων
- Δυσκολία: Ο υπολογισμός του AP απαιτούσε την παρακολούθηση της θέσης κάθε σχετικού εγγράφου στη λίστα αποτελεσμάτων
- Δυσκολία: Έπρεπε να δημιουργήσουμε αξιόπιστα test queries με τα αναμενόμενα σχετικά έγγραφα
- Δυσκολία: Η διαχείριση σύνθετων queries με AND, OR, NOT και παρενθέσεις ήταν περίπλοκη
- Δυσκολία: Οι υπολογισμοί των metrics ήταν χρονοβόροι για μεγάλο αριθμό εγγράφων
- Δυσκολία: Ο εντοπισμός προβλημάτων στους υπολογισμούς των metrics ήταν δύσκολος