Συστήματα Ανάκτησης Πληροφορίας

Ερωτήματα Κειμένων (Query by Document)

Ομάδα 1

Τσόγκας Ευάγγελος

3150185

Περιεχόμενα

ΦΑΣΗ 1

Παραδοτέο

- Ο φάκελος **QueryByDocumentProject** αποτελεί το διαθέσιμο προς εκτέλεση maven IntelliJ project. Περιέχει δύο main μεθόδους. Η μία βρίσκεται στην κλάση CreateFilesMain και χρειάζεται να εκτελεστεί πρώτη για να δημιουργηθούν τα απαραίτητα αρχεία. Η άλλη βρίσκεται στην κλάση ElasticSearchMain και αναλαμβάνει την εκτέλεση των λειτουργιών σχετικές με την ElasticSearch. Τα πάντα είναι αυτοματοποιημένα, ακόμα και η δημιουργία και φόρτωση δεδομένων στο ευρετήριο. Στον φάκελο input του project υπάρχει το αρχείο testingQueries.txt με τα ερωτήματα. Οι έξοδοι του προγράμματος αποθηκεύονται στον φάκελο ouput ο οποίος δημιουργείται κατά την εκτέλεση της CreateFilesMain.
- Στον φάκελο **Έξοδοι** υπάρχει το αρχείο system_qrels.txt με τις απαντήσεις στα ερωτήματα που δημιουργεί το πρόγραμμα μετά την εκτέλεσή του, καθώς και το αρχείο eval.txt που παράχθηκε από την αξιολόγηση με το εργαλείο trec_eval. Η συλλογή με τα καινούρια xml και το αρχείο json παραλείπονται λόγω το όγκου τους, αλλά φυσικά μπορούν εύκολα να δημιουργηθούν με την εκτέλεση του προγράμματος.

Αξιολόγηση Αποτελεσμάτων

Η αξιολόγηση των αποτελεσμάτων των ερωτημάτων έγινε με το εργαλείο trec_eval για το σύνολο των ερωτημάτων και βρίσκεται στο αρχείο eval.txt:

MAP (Mean Average Precision) = **0.4901**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.7000**
- Για **k = 10**: P_10 = **0.6100**
- Για **k = 15**: P_15 = **0.5267**
- Για **k = 20**: P_20 = **0.4750**

Κώδικας

Class CreateFilesMain

Με την εκτέλεσή της δημιουργείται η καινούρια συλλογή των xml με το πεδίο text, καθώς και το αρχείο texts.json από το οποίο θα φορτωθούν, στη συνέχεια, τα κείμενα στο ευρετήριο της ElasticSearch.

- **void main(String args[])**: Αρχικά ζητείται να δοθεί το μονοπάτι στο φάκελο που περιέχει τη συλλογή των xml. Για κάθε xml που διαβάζει, δημιουργεί το αντίστοιχο με το πεδίο text και γράφει τα περιεχόμενα στο αρχείο texts.json.
- **void createXMLWithTextElement(String fileName)**: Δημιουργεί ένα αρχείο xml με το πεδίο text.
- **void writeJSON()**: Γράφει στο αρχείο texts.json τα περιεχόμενα ενός xml.

Class ElasticSearchMain

- **void main(String args[])**: Χρησιμοποιεί την κλάση Client για τη δημιουργία του ευρετηρίου (το όνομα του οποίου δίνεται ως input) εάν δεν υπάρχει, φορτώνει τα δεδομένα από το αρχείο texts.json, εκτελεί τα ερωτήματα και γράφει τα αποτελέσματά τους.
- **ArrayList<String> readQueries()**: Διαβάζει τα ερωτήματα από το αρχείο testingQueries.txt στο φάκελο input και τα αποθηκεύει σε ένα πίνακα αφού κάνει replace κάποια σύμβολα που δημιουργούν πρόβλημα κατά την εκτέλεση των ερωτημάτων, όπως τα \, /, :, & κλπ.
- **void writeReplies(ArrayList<float[]> replies, int queryId)**: Για καθένα ερώτημα γράφει τις απαντήσεις του στο αρχείο system_qrels.txt με τη μορφή που χρειάζεται για την αξιολόγησή τους με το trec_eval.

Class Client

Αυτή η κλάση χρησιμοποιείται ως client της ElasticSearch.

- **Client(String indexName)**: Κατασκευαστής που ανοίγει τον high level rest client της ElasticSearch.
- **void close()**: Κλείνει τον client.
- **boolean indexExists()**: Ελέγχει εάν υπάρχει το ευρετήριο χρησιμοποιώντας low level client εντολή.
- **void createIndex()**: Δημιουργεί το ευρετήριο με low level client εντολή. Χρησιμοποιεί τον standard analyzer της ElasticSearch ο οποίος παρέχει tokenization, μετατροπή σε lowercase και stopwords removal (δηλώνεται ρητά, καθώς by default είναι απενεργοποιημένο). Επίσης χρησιμοποιεί φίλτρο για stemming. Όσον αφορά τα βάρη γίνεται χρήση των BM25 που αν και είναι το default δηλώνεται ρητά.
- **void deleteIndex()**: Διαγράφει το ευρετήριο.
- **void insertData()**: Φορτώνει τα κείμενα στο ευρετήριο από το αρχείο texts.json.
- **ArrayList<float[]> fullTextQuery(String query)**: Εκτελεί ένα ερώτημα και επιστρέφει τις απαντήσεις σε μια λίστα όπου κάθε στοιχείο είναι ένα ζευγάρι τιμών id κειμένου και score κειμένου. Επιστρέφει 20 απαντήσεις εξαιρώντας το ίδιο κείμενο του ερωτήματος.

ΦΑΣΗ 2

Παραδοτέο

- Ο φάκελος **QueryByDocumentProject** αποτελεί το διαθέσιμο προς εκτέλεση maven IntelliJ project, όπως και στη φάση 1, αλλά με τις κατάλληλες προσθήκες κώδικα.
- Στον φάκελο **Έξοδοι/Phrases** υπάρχουν τα αρχεία system_qrels30%.txt, system_qrels60%.txt, system_qrels90%.txt, με τις απαντήσεις στα ερωτήματα (για το αντίστοιχο ποσοστό φράσεων) που δημιουργεί το πρόγραμμα μετά την εκτέλεσή του. Τα αρχεία eval30%.txt, eval60%.txt και eval90%.txt αποτελούν τις αντίστοιχες αξιολογήσεις.
- Στον φάκελο **Έξοδοι/MLT** υπάρχουν τα αρχεία απαντήσεων και αξιολογήσεων για τα ερωτήματα More Like This.

Αξιολόγηση Αποτελεσμάτων

(a) Αξιολόγηση αποτελεσμάτων για ερωτήματα εξαγμένων φράσεων

Αξιολόγηση των αποτελεσμάτων για το 30% των φράσεων:

MAP (Mean Average Precision) = **0.3802**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.5800**
- Για **k = 10**: P_10 = **0.4800**
- Για **k = 15**: P_15 = **0.4267**
- Για **k = 20**: P_20 = **0.4000**

Αξιολόγηση των αποτελεσμάτων για το 60% των φράσεων:

MAP (Mean Average Precision) = **0.4322**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.6800**
- Για **k = 10**: P_10 = **0.5900**
- Για **k = 15**: P_15 = **0.5067**
- Για **k = 20**: P_20 = **0.4400**

Αξιολόγηση των αποτελεσμάτων για το 90% των φράσεων:

MAP (Mean Average Precision) = **0.4526**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.6400**
- Για **k = 10**: P_10 = **0.5700**
- Για **k = 15**: P_15 = **0.5333**
- Για **k = 20**: P_20 = **0.4450**

(b) Αξιολόγηση αποτελεσμάτων για More Like This ερωτήματα

Αξιολόγηση των αποτελεσμάτων για default τιμές των MLT:

(αρχεία system_qrelsMLTDefault.txt, evalMLTDefault.txt με τις απαντήσεις και την αξιολόγηση αντίστοιχα)

MAP (Mean Average Precision) = **0.2288**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.4800**
- Για **k = 10**: P_10 = **0.3900**
- Για **k = 15**: P_15 = **0.3200**
- Για **k = 20**: P_20 = **0.2650**

Αξιολόγηση των αποτελεσμάτων για ερωτήματα με πιο σημαντικούς όρους:

(αρχεία system_qrelsMLT1.txt, evalMLT1.txt με τις απαντήσεις και την αξιολόγηση αντίστοιχα)

Παράμετροι: min_term_freq: 3, min_doc_freq: 6, max_doc_frec: 1000

MAP (Mean Average Precision) = **0.3130**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.6200**
- Για **k = 10**: P_10 = **0.4500**
- Για **k = 15**: P_15 = **0.3733**
- Για **k = 20**: P_20 = **0.3150**

Αξιολόγηση των αποτελεσμάτων για ερωτήματα με λιγότερους όρους:

(αρχεία system_qrelsMLT2.txt, evalMLT2.txt με τις απαντήσεις και την αξιολόγηση αντίστοιχα)

Παράμετροι: max_query_terms: 10

MAP (Mean Average Precision) = **0.2851**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.4800**
- Για **k = 10**: P_10 = **0.4400**
- Για **k = 15**: P_15 = **0.3867**
- Για **k = 20**: P_20 = **0.3200**

Αξιολόγηση των αποτελεσμάτων για πιο χαλαρά ερωτήματα:

(αρχεία system_qrelsMLT3.txt, evalMLT3.txt με τις απαντήσεις και την αξιολόγηση αντίστοιχα)

Παράμετροι: max_query_terms: 100, min_term_freq: 1, min_doc_freq: 1, minimum_should_match: "10%"

MAP (Mean Average Precision) = **0.2683**

AvgPre@k (Μέση ακρίβεια για στα k πρώτα ανακτηθέντα κείμενα:

- Για **k = 5**: P_5 = **0.4800**
- Για **k = 10**: P_10 = **0.4000**
- Για **k = 15**: P_15 = **0.3533**
- Για **k = 20**: P_20 = **0.3050**

Κώδικας

Παρακάτω καταγράφονται οι μέθοδοι που προστέθηκαν στον υπάρχοντα κώδικα.

Class ElasticSearchMain

- **ArrayList<String> queriesFromExtractedPhrases(String directory, float percentage):** Διαβάζει τα 10 αρχεία της συλλογής και δημιουργεί τα τελικά ερωτήματα ανάλογα με το ποσοστό των φράσεων που έχει δοθεί ως παράμετρος, τα οποία στη συνέχεια αποθηκεύει σε ένα πίνακα τον οποίο και επιστρέφει. Η παράμετρος directory είναι το μονοπάτι στο φάκελο με τη συλλογή των αρχείων. Η εκτέλεση των ερωτημάτων αυτών (για ποσοστά φράσεων 30%, 60% και 90%) και η καταγραφή των αποτελεσμάτων τους γίνεται στη main μέθοδο.

Class Client

- **ArrayList<float[]> MLTQuery(String query):** Εκτελεί ένα ερώτημα MLT και επιστρέφει τις απαντήσεις σε μια λίστα όπου κάθε στοιχείο είναι ένα ζευγάρι τιμών id κειμένου και score κειμένου. Επιστρέφει 20 απαντήσεις εξαιρώντας το ίδιο κείμενο του ερωτήματος.