Προγραμματιστική εργασία: Σημασιολογική ανάκτηση

Φάση 1

Καραγιαννάκος Άριστείδης ΑΜ 3220066

Κοσμίδη Έλλη ΑΜ 3220086

Elastic serach: bin\elasticsearch.bat , curl http://localhost:9200/, notepad .\config\elasticsearch.yml , xpack.security.enabled: false

xpack.security.http.ssl.enabled: false

A1 και A2

Αρχικά δημιουργήσαμε την συνάρτηση **process_text** που δέχεται παράμετρο text και το επεξεργάζεται, δηλαδή αφαιρεί τα πολλά κενά και σύμβολα αλλαγής γραμμής, tabs κλπ (\r,\n,\t) και τα αντικαθιστά με ένα κενό.

Ύστερα για το index της Elasticsearch:

- Δημιουργήσαμε BM25_optimized_mapping που ορίζει τον τρόπο που θα επεξεργάζεται και θα αντιστοιχεί το index τα δεδομένα του. Συγκεκριμένα:

  Analyzer υπεύθυνος για να σπάει το text σε tokens, να αφαιρεί τα stopwords σύμφωνα με τη λίστα "_english_".

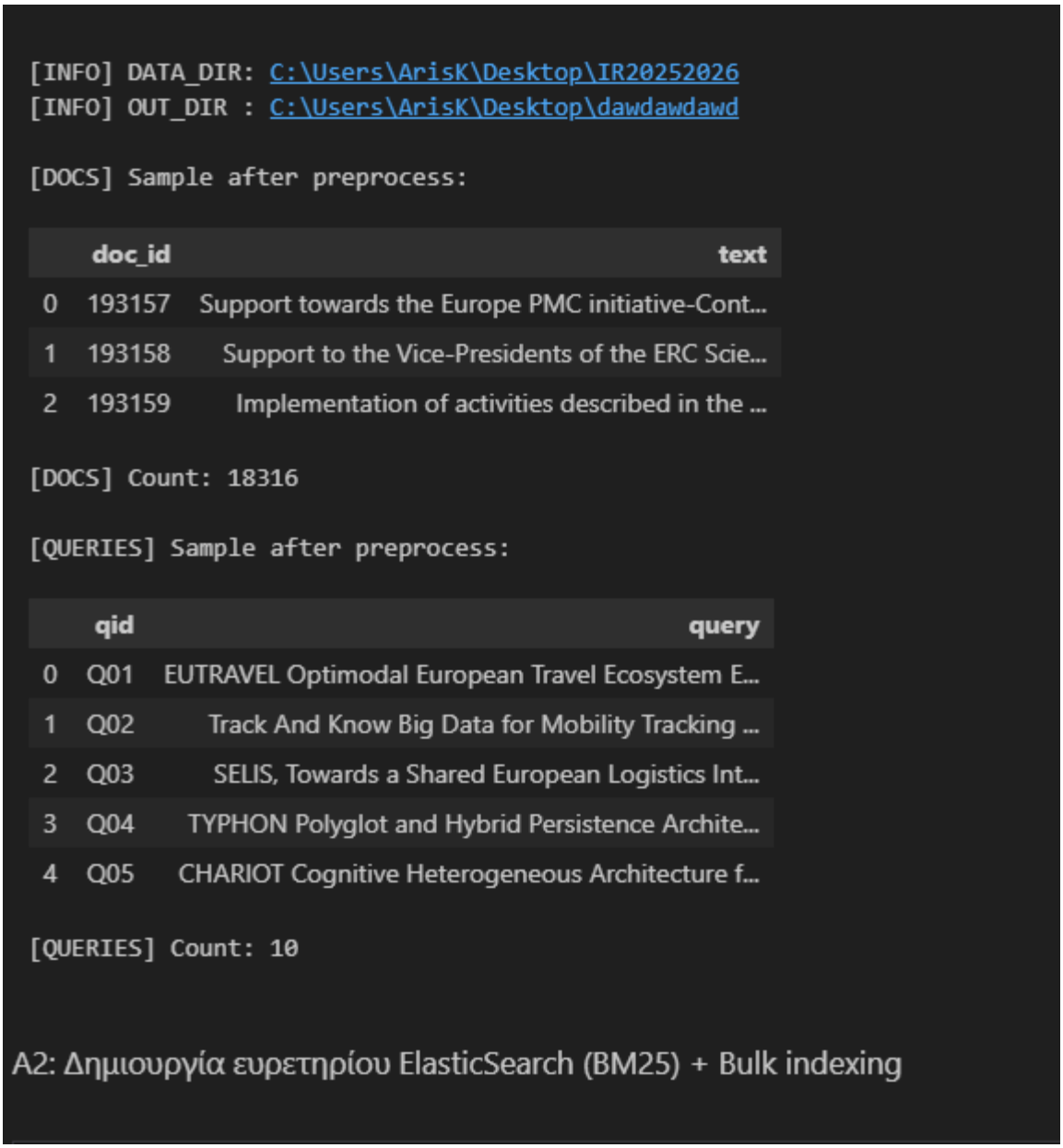
  Similarity καθορίσαμε τον BM25 αλγόριθμο βαθμολόγησης που θα χρησιμοποιήσει η Elasticsearch (αλγόριθμος βαθμολόγησης δίνει υψηλότερη βαθμολογία σε έγγραφα που είναι πιο σχετικά με το query και χαμηλότερη σε αυτά που είναι λιγότερο σχετικά).

  Mapping στο ευρετήριο ορίσαμε σαν keyword το id και text (περιεχόμενο ευρετηρίου) το text που θα τροποποιηθεί με τον "english_analyzer" που ορίσαμε πριν.

- Δημιουργήσαμε την συνάρτηση **generate_actions**, η οποία ορίζει τον τρόπο που θα εισαχθούν τα κείμενα μέσα στο index όταν θα χρησιμοποιήσουμε την συνάρτηση bulk. Δηλαδή η generate_actions δέχεται το documents.csv, διαβάζει μια μια τις γραμμές και ουσιαστικά για κάθε γραμμή/ κείμενο φτιάχνει ένα dict που ορίζει σε ποιο index θα μπει, τι id θα έχει και τι περιεχόμενο, με δομή κατάλληλη για την συνάρτηση bulk (κλειδιά _index, _id, _source και οι αντίστοιχες τιμές τους).

  * το _source περιέχει τα πραγματικά δεδομένα του κάθε κειμένου σε μορφή dictionary. Για το text καλούμε την συνάρτηση process_text για προεπεξεργασία.

- Χρησιμοποιήσαμε την έτοιμη συνάρτηση bulk για το γέμισμα του ευρετηρίου με παραμέτρους την σύνδεση με την Elasticsearch και την generate_actions



Το scresshot δεν φαινόταν καλά αυτό ειναι το print φαίνεται και στον κώδικα.

[OK] Elasticsearch: elasticsearch | 9.2.1

[i] Deleted existing index: ir2025_phase1_bm25

[OK] Created index 'ir2025_phase1_bm25' with BM25(k1=1.2, b=0.75)

[BULK] Indexing documents...

C:\Users\ArisK\AppData\Local\Temp\ipykernel_3308\1684811186.py:61: DeprecationWarning: Passing transport options in the API method is deprecated. Use 'Elasticsearch.options()' instead.

helpers.bulk(es, doc_actions(df_docs, INDEX_NAME), chunk_size=1000, request_timeout=120)

[OK] Indexed docs in ES: 18316

A3

Για το A3 φορτώσαμε τα ερωτήματα από το αρχείο queries.csv χρησιμοποιώντας τη βιβλιοθήκη pandas. Το αρχείο περιέχει για κάθε ερώτημα ένα μοναδικό αναγνωριστικό (ID) και το αντίστοιχο κείμενο (Text). Οι στήλες μετονομάστηκαν σε qid και query αντίστοιχα και εφαρμόστηκε βασική προεπεξεργασία στο κείμενο των ερωτημάτων με τη συνάρτηση clean_text, ώστε να είναι συμβατή με την επεξεργασία των εγγράφων.

Στη συνέχεια, για κάθε ερώτημα εκτελέσαμε αναζήτηση στο ευρετήριο της Elasticsearch, συγκρίνοντας το κείμενο του ερωτήματος με τα κείμενα των εγγράφων του index. Η κατάταξη των εγγράφων πραγματοποιήθηκε με βάση τη βαθμολογία σχετικότητας που υπολογίζεται από τον αλγόριθμο BM25, ο οποίος ορίστηκε στο στάδιο A2.

Για κάθε ερώτημα συλλέξαμε τα top-k αποτελέσματα και δημιουργήσαμε τρία αρχεία run για k = 20, 30 και 50 (run_bm25_k20.txt, run_bm25_k30.txt, run_bm25_k50.txt).

Το scresshot δεν φαινόταν καλά αυτό ειναι το print φαίνεται και στον κώδικα.

Queries k=20: 100%|████████████████| 10/10 [00:00<00:00, 13.06it/s]

[OK] Wrote run_bm25_k20.txt (200 lines)

Queries k=30: 100%|████████████████| 10/10 [00:00<00:00, 34.77it/s]

[OK] Wrote run_bm25_k30.txt (300 lines)

Queries k=50: 100%|████████████████| 10/10 [00:00<00:00, 37.98it/s]

[OK] Wrote run_bm25_k50.txt (500 lines)

A4

Για την αξιολόγηση χρησιμοποιήσαμε το εργαλείο trec_eval, συγκρίνοντας τα run files με το ground truth αρχείο qrels.txt. Τρέξαμε το trec_eval για τα μέτρα:

MAP (Mean Average Precision), που συνοψίζει την ποιότητα κατάταξης σε όλο το ranking,

Precision @k για k = 5, 10, 15, 20, που μετρά πόσα από τα πρώτα k αποτελέσματα είναι συναφή.

Για κάθε run file (k=20,30,50) εκτελέσαμε:

-m map για MAP

-m P.5, -m P.10, -m P.15, -m P.20 για precision στα αντίστοιχα cutoff.

Στη συνέχεια κάναμε parsing της εξόδου σε pandas και δημιουργήσαμε:

πίνακες ανά query (per-query results),

καθώς και συγκεντρωτικό πίνακα "all" που περιέχει τα συνολικά MAP και P@k για κάθε run.

Τέλος, αποθηκεύσαμε τα αποτελέσματα σε CSV ώστε να τα ενσωματώσουμε στην αναφορά και να κάνουμε σύγκριση μεταξύ των διαφορετικών τιμών k.

Συμπέρασμα:

Από τα αποτελέσματα παρατηρούμε ότι το MAP αυξάνεται όσο μεγαλώνει το k, γεγονός που δείχνει ότι το BM25 κατατάσσει σωστά περισσότερα συναφή έγγραφα όταν εξετάζουμε μεγαλύτερο μέρος της λίστας αποτελεσμάτων. Αντίθετα, οι τιμές Precision@5, Precision@10, Precision@15 και Precision@20 παραμένουν ίδιες για όλα τα k, κάτι που σημαίνει ότι τα πιο σχετικά έγγραφα εμφανίζονται σταθερά στις πρώτες θέσεις.

Συνολικά, το BM25 παρουσιάζει σταθερή και καλή απόδοση ως baseline μοντέλο, αποτελώντας μια αξιόπιστη βάση για σύγκριση με πιο προχωρημένες μεθόδους ανάκτησης στις επόμενες φάσεις της εργασίας.

Εκτελείται αξιολόγηση για: run_bm25_k20.txt

Metric	map	P@5	P@10	P@15	P@20
Query					
Q01	0.62280	0.80000	0.80000	0.73330	0.65000
Q02	0.32610	0.60000	0.30000	0.26670	0.30000
Q03	0.70970	0.80000	0.70000	0.66670	0.60000
Q04	0.29750	0.60000	0.40000	0.26670	0.30000
Q05	0.70660	1.00000	0.80000	0.73330	0.65000
Q06	0.69260	1.00000	1.00000	0.80000	0.70000
Q07	0.64130	1.00000	0.80000	0.73330	0.60000
Q08	0.46940	0.80000	0.70000	0.46670	0.45000
Q09	0.67690	1.00000	0.90000	0.86670	0.75000
Q10	0.31980	0.60000	0.30000	0.26670	0.25000
all	0.54630	0.82000	0.67000	0.58000	0.52500

Saved: C:\Users\ArisK\Desktop\dawdawdawd\run_bm25_k20_per_query.csv

Εκτελείται αξιολόγηση για: run_bm25_k30.txt

Metric	map	P@5	P@10	P@15	P@20
Query					
Q01	0.69690	0.80000	0.80000	0.73330	0.65000
Q02	0.32610	0.60000	0.30000	0.26670	0.30000
Q03	0.74070	0.80000	0.70000	0.66670	0.60000
Q04	0.31840	0.60000	0.40000	0.26670	0.30000
Q05	0.73780	1.00000	0.80000	0.73330	0.65000
Q06	0.76510	1.00000	1.00000	0.80000	0.70000
Q07	0.71360	1.00000	0.80000	0.73330	0.60000
Q08	0.57030	0.80000	0.70000	0.46670	0.45000
Q09	0.74260	1.00000	0.90000	0.86670	0.75000
Q10	0.39930	0.60000	0.30000	0.26670	0.25000
all	0.60110	0.82000	0.67000	0.58000	0.52500

💾 Saved: C:\Users\ArisK\Desktop\dawdawdawd\run_bm25_k30_per_query.csv

Εκτελείται αξιολόγηση για: run_bm25_k50.txt

Εκτελείται αξιολόγηση για: run_bm25_k50.txt

Metric	map	P@5	P@10	P@15	P@20
Query					
Q01	0.72910	0.80000	0.80000	0.73330	0.65000
Q02	0.38270	0.60000	0.30000	0.26670	0.30000
Q03	0.74070	0.80000	0.70000	0.66670	0.60000
Q04	0.34880	0.60000	0.40000	0.26670	0.30000
Q05	0.79040	1.00000	0.80000	0.73330	0.65000
Q06	0.78490	1.00000	1.00000	0.80000	0.70000
Q07	0.74290	1.00000	0.80000	0.73330	0.60000
Q08	0.59000	0.80000	0.70000	0.46670	0.45000
Q09	0.84870	1.00000	0.90000	0.86670	0.75000
Q10	0.42660	0.60000	0.30000	0.26670	0.25000
all	0.63850	0.82000	0.67000	0.58000	0.52500

💾 Saved: C:\Users\AriskK\Desktop\dawdawdawd\run_bm25_k50_per_query.csv

=== ΣΥΓΚΕΝΤΡΩΤΙΚΑ (all) για όλα τα runs ===

	map	P@5	P@10	P@15	P@20
run					
run_bm25_k20.txt	0.54630	0.82000	0.67000	0.58000	0.52500
run_bm25_k30.txt	0.60110	0.82000	0.67000	0.58000	0.52500
run_bm25_k50.txt	0.63850	0.82000	0.67000	0.58000	0.52500

💾 Saved: C:\Users\AriskK\Desktop\dawdawdawd\phase1_trec_eval_summary_all.csv