**Αναφορά Εργασίας Υλοποίησης Συστήματος Ανάκτησης Πληροφορίας με χρήση Lucene**



Εαρινό Εξάμηνο 2023

Ομάδα :

* Αδαμάντιος Γκαραβέλας ΑΜ: 4007
* Βασίλης Γεωργάκης ΑΜ: 3197

GitHub Link : https://github.com/AkisGk/AnaktisiProject.git

**Σύντομη περιγραφή Εργαστηριακής Άσκησης**

Η εργασία πραγματεύεται το σχεδιασμό και την υλοποίηση ενός συστήματος αναζήτησης τραγουδιών και άλλης πληροφορίας σχετικής με μουσικούς και τραγούδια. Για την υλοποίηση χρησιμοποιήθηκε η βιβλιοθήκη Lucene. Αρχικά γίνεται η συλλογή εγγράφων σχετικά με τραγούδια, συλλογές τραγουδιών ή μουσικούς (scrapping). Στη συνέχεια προστίθενται λειτουργίες ανάλυσης κειμένου (όπως η διόρθωση τυπογραφικών λαθών, κλπ.) και η κατασκευή ευρετηρίου. Η αναζήτηση εγγράφων γίνεται με λέξεις κλειδιά με κάποιες παραπάνω λειτουργίες και τέλος γίνεται παρουσίαση αποτελεσμάτων.

**Συλλογή Εγγράφων (corpus)**

Για την συλλογή των εγγράφων γίνεται από δύο πηγές, Spotify API και Genius API. Αρχικά χρειαζόμαστε τα Spotify Client Id, Spotify Client Secret, και το Spotify access token μέσω της getSpotifyAccessToken, ώστε να συνδεθούμε στο Spotify και χρησιμοποιώντας playlist Ids να βρίσκουμε τις βασικές πληροφορίες για τα τραγούδια της playlist. Πιο συγκεκριμένα, στην κλάση LyricsCollector, χρσησιμοποιώντας την getPlaylistTracks παίρνουμε μια λίστα με τα τραγούδια της Playlist. Στη συνέχεια για κάθε τραγούδι βάζουμε σε ένα Object Song όλες τις πληροφορίες ενός τραγουδιού, δηλαδή τον τίτλο, το όνομα του καλλιτέχνη, το όνομα του άλμπουμ, την ημερομηνία κυκλοφορίας, και τοποθετούμε το Object σε μία λίστα με όλα τα τραγούδια allSongs. Μετά αφού έχουμε πάρει και το Genius access token, χρησιμοποιούμε το Genius API και έχοντας τις πληροφορίες των τραγουδιών παίρνουμε και τους στίχους που τους αντιστοιχούν και κάνει scrape την ιστοσελίδα με Jsoup. Στην collectSongs παίρνουμε τα τραγούδια που θέλουμε μέσα από λίστα με Playlists, κρατώντας υπόψη ότι χρειαζόμαστε τουλάχιστον 500. Με αυτή τη μέθοδο που ακολουθήσαμε παρατηρήσαμε ότι κάθε περίπου 1 ώρα το Spotify Access Token έχανε την ισχύ του. Για το λόγο αυτό φτιάξαμε την startTokenRefresher η οποία ανανέωνε το Spotify Access Token κάθε 3500 δευτερόλεπτα, ώστε να μη διακοπεί η διεργασία. Αφού έχουμε ολοκληρώσει την διεργασία αυτή, έπειτα στην κλάση DatasetCollector και έχουμε συλλέξει τις πληροφορίες που θέλουμε, αποθηκεύουμε τα δεδομένα μας σε ένα text file χρησιμοποιώντας την saveToTextFile, σε ένα serialized file χρησιμοποιώντας την saveToSerFile, και σε μία βάση δεδομένων χρησιμοποιώντας την saveToDatabase. Η βάση δεδομένων έχουμε φροντίσει να δημιουργηθεί στην DatabaseManager, με όνομα songs, και πεδία: id INTEGER PRIMARY KEY, title TEXT NOT NULL, artist TEXT NOT NULL, album TEXT, release\_date TEXT, lyrics TEXT, UNIQUE(title, artist)), έχοντας βάλει Unique τον τίτλο και τον καλλιτέχνη έτσι ώστε η removeDuplicates να μπορεί να αφαιρέσει τις πολλαπλές εμφανίσεις.

**Ανάλυση Κειμένου και Κατασκευή Ευρετηρίου**

Πριν από την ευρετηρίαση του συνόλου δεδομένων, εκτελούνται κάποιες εργασίες προεπεξεργασίας. Ξεκινάμε στην κλάση CustomAnalyzer, όπου δημιουργούμε ένα custom analyzer για την επεξαργασία που θέλουμε. Πιο συγκεκριμένα, φτιάχνουμε ένα TokenStreamComponents object το οποίο συνδυάζει : Tokenization : Διαχωρισμός του κειμένου σε μεμονωμένους όρους, LowerCaseFilter : Μετατροπή όλου του κειμένου σε πεζά γράμματα για τη διασφάλιση της αναζήτησης χωρίς διάκριση πεζών-κεφαλαίων, StopFilter : Αφαίρεση κοινών αγγλικών λέξεων (π.χ. the, and, etc.) που δεν συμβάλλουν στη συνάφεια της αναζήτησης,SynonymGraphFilter όπου χρησιμοποιεί ένα Synonym file που έχουμε δημιουργήσει, το οποίο περιέχει 100 λέξεις και συνώνημες τους. Τέλος μέσω της κλάσης AcronymExpander, δημιουργήσαμε ένα HashMap με ακρώνυμα και το πλήρες νόημά τους έτσι ώστε αν κάποιος αναζητήσει π.χ. YMCA ή Young Men's Christian Association να λάβει τα ίδια αποτελέσματα. Η ενότητα εγγράφων στο σύστημα μας είναι το κάθε μεμονωμένο τραγούδι, και κάθε ενότητα θα έχει τα εξής πεδία:

Title: Ο τίτλος του τραγουδιού

Artist : Το όνομα του καλλιτέχνη

Album : Το όνομα του άλμπουμ στο οποίο ανήκει το τραγούδι

Release Date : Η ημερομηνία κυκλοφορίας του τραγουδιού

Lyrics : Οι στίχοι του τραγουδιού

Το ευρετήριο που κατασκευάσαμε είναι ευρετήριο Lucene και είναι με τα εξής πεδία:

Title : Ένα πεδίο για το τίτλο του τραγουδιού με ευρετήριο, με διακριτικό και αποθηκευμένο πεδίο

Artist : Ένα πεδίο για το όνομα του καλλιτέχνη με ευρετήριο, με διακριτικό και αποθηκευμένο πεδίο

Album : Ένα πεδίο για το όνομα του άλμπουμ με ευρετήριο, με διακριτικό και αποθηκευμένο πεδίο

Release Date : Ένα πεδίο για την ημερομηνία κυκλοφορίας του τραγουδιού, με διακριτικό και αποθηκευμένο πεδίο

Lyrics : Ένα πεδίο για τους στίχους του τραγουδιού με ευρετήριο, με διακριτικό και αποθηκευμένο πεδίο

Αυτή η δομή μας επιτρέπει να υποστηρίζουμε διάφορους τρόπους αναζήτησης, όπως αναζήτηση ανά τίτλο, άλμπουμ, καλλιτέχνη, ημερομηνία κυκλοφορίας ή στίχους.

**Αναζήτηση**

Το σύστημα μας έχει κατασκευαστεί για να κάνει αναζήτηση με λέξεις ή φράσεις κλειδιά, ενώ ταυτόχρονα κρατάει και ιστορικό για τις αναζητήσεις του χρήστη, με χρήση της κλάσης QueryHistory. Πιο συγκεκριμένα όταν αναζητά με λέξεις ή κλειδιά, αναλόγως τα πεδία που θέλει να ψάξει, γίνεται αναζήτηση σε όλα τα έγγραφα για να βρεθούν οι ακριβείς λέξεις ή φράσεις. Επιπρόσθετα με χρήση του Query History του χρήστη αλλά και του Synonym File που έχουμε φτιάξει, μέσω της alternateQuery προτείνονται στο χρήστη Queries τα οποία είναι συνώνυμα με τα Queries που έχει ήδη αναζητήσει. Με τον τρόπο αυτό αν δεν βρει αυτό που αναζητάει με συγκεκριμένες λέξεις ίσως καταφέρει να το βρει με τις συνώνυμες αυτών.

**Παρουσίαση Αποτελεσμάτων**

Για την παρουσίαση των αποτελεσμάτων φτιάξαμε στην κλάση SongSearchApp ένα βασικό GUI, με διαστάσεις 800x600 και τίτλο “Song Lyrics Search”. Στη συνέχεια δημιουργήσαμε ένα Panel για τα παρακάτω στοιχεία : Search Panel, Results Panel και Checkbox Panel. Πιο συγκεκριμένα υπάρχει ένα πεδίο όπου ο χρήστης μπορεί να γράψει τις λέξεις ή φράσεις τις οποίες αναζητά. Από κάτω υπάρχουν τα εξής checkboxes: Title, Artist, Album, Lyrics για το κάθε πεδίο στο οποίο θα ήθελε να γίνει αναζήτηση, ένα checkbox All Fields για να γίνει σε όλα τα πεδία η αναζήτηση καθώς και ένα checkbox για Semantic Search το οποίο αφορά το προαιρετικό ερώτημα. Αφού κάνει τις απαραίτητες ενέργειες σύμφωνα με τα παραπάνω ο χρήστης πατάει το κουμπί Enter. Έπειτα στο Panel για τα αποτελέσματα, θα εμφανιστούν τα τραγούδια και όλες οι απαραίτητες πληροφορίες (τίτλος, άλμπουμ, καλλιτέχνης, στίχοι) που ταιριάζουν στην αναζήτησή του. Τα αποτελέσματα θα είναι σε σελίδες των 10, δηλαδή 10 αποτελέσματα ανά σελίδα, καθώς και οι λέξεις ή φράσεις που αναζήτησε και ταιριάζουν σε αποτελέσματα θα είναι τονισμένες με έντονα γράμματα. Ο χρήστης έχει τη δυνατότητα να πατήσει το κουμπί “Next Page” για πλοήγηση στα αποτελέσματα. Τέλος όλα τα αποτελέσματα θα είναι ομαδοποιημένα ανά καλλιτέχνη, καθώς υποθέσαμε ότι είναι το πιο πρακτικό όταν κάποιος αναζητά πληροφορίες για τραγούδια να είναι ομαδοποιημένα με βάση τους καλλιτέχνες.

**Προαιρετικό Ερώτημα**

Στόχος του ερωτήματος αυτού είναι η επέκταση της μηχανικής αναζήτησης ώστε να υποστηρίζεται η σημασιολογική αναζήτηση. Πιο συγκεκριμένα, να χρησιμοποιήσουμε κάποιο μοντέλο παραγωγής διανυσματικών αναπαραστάσεων (embedding). Οι διανυσματικές αυτές αναπαραστάσεις πρέπει να αποθηκευτούν σε κάποια δομή. Στη συνέχεια η ανάκτηση των εγγράφων θα γινόταν εξετάζοντας την ομοιότητα μεταξύ των διανυσματικών αυτών αναπαραστάσεων. Η υλοποίηση μας ξεκινάει με ένα script σε python, το Word2Vec\_Embedding, το οποίο αντιστοιχούσε σε κάθε λέξη ένα vector και μετά υπολόγιζε ένα vector για κάθε τραγούδι βγάζοντας ένα μέσο όρο από τα vectors των στίχων του και τα αποθηκεύει σε ένα json αρχείο. Έπειτα στην κλάση VectorLoader, πέρναμε τα vectors των τραγουδίων από το αρχείο, και με βάση το id τους και τα vector τους συγκρίναμε τα συνημίτονα τους ώστε να βρούμε την ομοιότητα τους. Στην κλάση Searcher παίρνουμε το Query με βάση το οποίο γίνεται η αναζήτησή μας σε όλα μας τα πεδία πλέον, βρίσκουμε το vector που του αντιστοιχεί, και στη συνέχεια το συγκρίνουμε με τα vectors από τα τραγούδια μας. Έχοντας θέσει ένα minimum similarity που θα πρέπει να έχουν τα vectors μας, βρίσκουμε τα τραγούδια που ταιριάζουν σημασιολογικά. Βέβαια η υλοποίηση μας έχει κάποια προβλήματα καθώς σύμφωνα με κάποια prints δεν είναι όλα τα vectors που αντιστοιχούν στα τραγούδια μας σωστά, επομένως η αναζήτηση δεν είναι πλήρως σωστή. Ωστόσο κάναμε ότι μπορέσαμε και με όσες γνώσεις είχαμε να το υλοποιήσουμε όσο καλύτερα μπορούμε.