פרויקט מנוע – אחזור מידע
חלק ב'

רועי כהן 204702518, רועי ניסן 205567522

הקדמה על התוכנית

בחלק זה של הפרויקט אנו בעצם מחברים את שני החלקים לכדי חלק אחד שלם שמטרתו להפעיל מנוע חיפוש המופעל על קורפוס של מסמכים ורץ על שאילתה או מספר שאילתות מקובץ אשר על פיהם המנוע מאחזר את המסמכים הרלוונטיים לאותה שאילתה.

כל חלק בתוכנית מסתמך על חלק אחר ולכל אחד מהחלקים אחריות אשר ייחודית לו ורק לו.

מחלקת Index

על מחלקת Index הרחבתי בדו"ח של חלק א' ועל כן לא ארחיב ואוסיף מידע כפול.

שדות המחלקה שהוספנו בחלק זה:

```
private HashMap<String,List<String>> fiveMostRelevantEntitys;
```

שדה זה הוא מבנה נתונים אשר הKey שלו הוא הdocument name והValue שלו הוא הרשימה של הישויות שקיימות עבור אותו document name.

שיטות המחלקה שהוספנו בחלק זה:

```
public void createDocumentsFromFile()
```

פונקציה זו היא פונקציה אשר טוענת לתוך מבנה הנתונים documents את כל המידע ממסמך Documents אשר נמצא בתוך תיקיית ה Posting עבור כל Document ויוצרת אובייקט חדש מסוג Doc עבור כל מסמך, בנוסף הפונקציה טוענת לתוכה את חמשת הישויות אשר שמורות עבור כל מסמך.

```
private void writeDocumentsToFile()
```

פונקציה זו היא פונקציה אשר כותבת לתוך קובץ(מסמך טקסט) את כל המידע אשר האינדקס שומר על מסמך כלשהו, את שמו, מספר המילים בו, מספר הmaxFrequency של מילה כלשהי ואת הישויות של אותו מסמך.

הסברים נוספים:

במהלך יצירת האינדקס (Parse and Posting) מצאנו את הישויות של כל מסמך ושמרנו את הישויות במבנה נתונים.

הישויות הרלוונטיות ביותר הן הישויות שמופיעות הכי הרבה פעמים במסמך כך שבמבנה הנתונים שמורות חמשת הישויות אשר מופיעות הכי הרבה במסמך.

אלגוריתם:

S.1→ מבנה נתונים של מסמך המכיל term וfrequecny שלו במסמך

R.2→מבנה נתונים שמכיל String

3. maxValue→0

4.currMaxEntity→empty String

5.עבור בלולאה כל עוד יש ישויות במסמך

5.1. אם termה הנוכחי הוא ישות:

5.1.1 אם termFreq>maxValue

5.1.1.1. term→currMaxEntity

5.1.1.2.termFreq→maxValue

R.5.2→currMaxEntity

R.6→חמשת termה הראשונים בR

אלגוריתם זה מבטיח שאם ישנם ישויות הן יהיו ממוינות בסדר עולה כך שהראשון יהיה הישות הפופולארית ביותר במסמך.

מחלקת ReadFile

על מחלקת ReadFile הרחבתי בדו"ח של חלק א' ועל כן לא ארחיב ואוסיף מידע כפול.

שדות המחלקה שהוספנו בחלק זה:

```
private List<String> queries;
```

זהו מבנה נתונים אשר מכיל בתוכו את כל הQueries מקובץ הqueries אשר נבחר להרצה על המנוע.

שיטות המחלקה שהוספנו בחלק זה:

```
public void readQueries(String path,boolean semantic){
```

זוהי פונקציה אשר מקבלת אליה נתיב לטעינת קובץ queries והאם המשתמש בחר בניתוח סמנטי או לא.

הפונקציה קוראת את כל הqueries לתוך מבנה הנתונים בצורה הבאה:

1. אם המשתמש בחר בניתוח סמנטי אז המידע שישמר בquery יהיה אך ורק המידע שמופיע בתגית <title>.
2. אחרת המידע שישמר בquery יהיה המידע שמופיע בתגית <title> והמידע שמופיע בתגית <desc>.

מחלקת Posting

שדות המחלקה שהוספנו בחלק זה:

```
private List<String> alphaBetL;
```

מבנה נתונים אשר מחזיק את שמות הקבצים לפני ביצוע פעולת המיזוג.

```
private List<String> alphaBetS;
```

מבנה נתונים אשר מחזיק את שמות קבצי הPosting עם stemming.

```
private final int K3 = 10000;
```

משתנה ששומר בתוכו את מספר השורות אשר נרצה לשמור לקובץ בפונקציה combineLines.

שיטות המחלקה שהוספנו בחלק זה:

```
public HashMap<String,Term> combineLines(HashMap<String,Term>
dictionary,boolean stem){
```

פונקציית מעטפת אשר עבור כל קובץ posting קוראת לפונקציית העבודה combineLines עבור כתיבה בקובץ ה Posting.

```
public HashMap<String,Term> combineLines(File oldFile,File
newFile,HashMap<String,Term> dictionary){
```

פונקציית עבודה אשר משלבת את השורות עבור אותו term באותו קובץ Posting.

שילוב השורות מתבצע באופן הבא:

Term,DocId,FrequencyInDoc|DocId,FrequencyInDoc….

```
private void writeNewPostingFile(File file,List<String> newPostingLines){
```

פונקציה אשר כותבת אל תוך קבצי הPosting.

```
public void deleteABCLFiles(){
```

פונקציה אשר מוודאת מחיקה של קבצי Posting ישנים שאינם ממוזגים לקבצי Posting מהצורה שcombineLine יוצרת.

```
public void clearFiles(){
```

פונקציה אשר כל תפקידה הוא לנקות את כל הקבצים אשר מחלקת Posting יוצרת ואין בהם שימוש בסוף תהליך יצירת הקבצים.

מחלקת Ranker

מחלקת Ranker היא מחלקה אשר מקבלת את המידע הנחוץ לה לחישוב הדירוג של כל מסמך בעבור אותו query שאנחנו עובדים עליו בהרצה זו.

מחלקה זו עובדת עם אלגוריתם BM25 אשר למדנו עליו בהרצאות ובתרגולים ותפקידו הוא להחזיר את הIdf עבור כל מסמך לפי משקולות וקבועים שנבחרו מראש.

זוהי מחלקה שכל יעודה הוא דירוג המסמך על פי מידע שנשמר בתהליך יצירת קבצי הIndex ומשתנים שונים.

שדות המחלקה:

```
private boolean semantic;
```

שדה בוליאני אשר בתוכו נשמר האם המשתמש בחר לבצע ניתוח סמנטי או לא.

```
private double k1;
```

משתנה k1 אשר משמש לחישוב הidf באלגוריתם BM25 המתבצע בפונקציית score.

```
private double b;
```

משתנה b אשר משמש לחישוב הidf באלגוריתם BM25 המתבצע בפונקציית score.

```
private int corpusSize;
```

משתנה השומר בתוכו את מספר המסמכים בקורפוס.

```
private List<HashMap<String,Integer>> listOfDocuments;
```

מבנה נתונים השומר בתוכו מבני נתונים נוספים אשר הKey שלהם הוא שם המסמך(docID) והValue שלהם הוא מספר ההופעות של המילה מהquery במסמך.

```
private Index index;
```

מצביע לindex.

```
private HashMap<String,Doc> documentsToRank;
```

מבנה נתונים השומר בתוכו את המסמכים ללא כפילויות בכדי שנוכל להגדיר עבור כל מסמך rank, בKey נשמר docID וב־Value נשמר המסמך כישות(Doc).

```
private HashMap<String,Doc> documentsInIndex;
```

מבנה נתונים המכיל את המסמכים מהIndex.

```
private HashMap<String,Double> documentAndRank;
```

מבנה הנתונים שיכיל בתוכו את הדירוג הסופי עבור כל מסמך כאשר הKey הוא הDocID והValue הוא הrank של אותו הdocument.

שיטות המחלקה:

```
public Ranker(boolean semantic,List<HashMap<String,Integer>>
listOfDocuments,Index index){
```

פונקציה זאת היא פונקציית הConstructor של המחלקה, הוא בעצם מקבלת מהSearcher ערך בוליאני שקובע האם בוצע ניתוח סמנטי או לא, מבנה נתונים המכיל את כל המידע עבור הquery מקבצי הPosting ומצביע לindex.

```
public List<Doc> startRank(){
```

פונקציית העבודה הראשית של המחלקה, הפונקציה בעצם רצה עבור כל מסמך אשר קיבלה ומחשבת עבורו את הRank הסופי ולבסוף מחזירה מבנה נתונים המכיל את ה50 מסמכים הכי רלוונטיים עבור אותה שאילתה.

```
public List<Doc> sortDocumentsByRank(){
```

פונקציה אשר תפקידה הוא לעבור על כך המסמכים שהוחזרו מהשאילתה ובוצעו עליהם חישובי rank ובעצם להחזיר 50 מסמכים ממוינים כאשר המסמך הראשון ברשימה הוא בעל הrank הגבוה ביותר.

```
public void updateDocumentAndRank(String docName,double rank){
```

פונקציה שמקבלת שם מסמך ורank עבורו ומעדכנת את ערך הrank שלו על ידי הוספת ערך הrank שקיבלה לערך הrank של אותו מסמך כרגע ברשימה.

```
public void noDuplicateDocument(){
```

פונקציה שרצה בתחילת פונקציית העבודה ותפקידה הוא לוודא כי במבנה הנתונים documentsToRank אין כפילויות של מסמכים על ידי הוספה שלהם פעם אחת בכדי שנוכל לחשב בעבורם את הrank הסופי.

```
public int calculateAvgLengthDocument(){
```

פונקציה אשר מחשבת את אורך המסמכים הממוצע עבור כל המסמכים אשר קיימים בקורפוס.

הסברים נוספים:

במחלקה זו עשינו שימוש באלגוריתם BM25 אשר תפקידו הוא ליצור פונקציית משקל אשר בה האלגוריתם ישתמש לבניית הIdf.

על מנת לדרג מסמכים רלוונטיים ביותר לשאילתה הנוסחה תקבל 5 פרמטרים והם:

1.Tf- (כמות הפעמים שהמילה מופיעה במסמך)/(מספר ההופעות הגדול ביותר של term במסמך)

2. NumberOfDocs – מספר המסמכים בקורפוס.

3. docLength – אורך המסמך.

4. avgDocLength – אורך מסמך ממוצע בקורפוס.

5. docFrequency – מספר המסמכים שבהם הterm מהשאילתה מופיע.

ישנם שני קבועים k1 ו-b שהם פרמטרים חופשיים כך שכל שינוי בקבועים נותן משקל שונה לכל אחד מהפרמטרים ועל כן הם נבחרים בצורה אמפירית על מנת לקבל תוצאות מיטביות.

הנוסחה:

$$K = k1 \cdot (1 - b) + \frac{b \cdot Length(Doc)}{Avg(Length(Doc))}$$

$$W = \frac{(k1 + 1) \cdot tf}{K + tf}$$

$$idf = W \cdot \log(Count(corpus) - Count(Term, Document) + 0.5)$$

מחלקת Searcher

מחלקת Searcher הוא המחלקה המקבלת את הQuery ובעצם בסוף התהליך מחזירה רשימה ממוינת של 50 מסמכים המדורגים מהגבוה לנמוך על פי רמת הרלוונטיות של כל מסמך שהוחזר.

מחלקה זו אחראית להביא את המידע הרלוונטי עבור כל מילה בquery מקובץ הPosting הרלוונטי.

במהלך התהליך מחלקה זו עובדת עם מחלקת Ranker בכדי לתת ציון עבור כל מסמך ובסופו של דבר מקבלת מהמחלקה רשימה של מסמכים ומחזירה למשתמש את המסמכים להצגתם.

שדות המחלקה:

```
private boolean semantic;
```

שדה בוליאני אשר בתוכו נשמר האם המשתמש בחר לבצע ניתוח סמנטי או לא.

```
private Ranker ranker;
```

מצביע לRanker.

```
private List<Doc> docList;
```

מבנה הנתונים ממוין המחזיק בתוכו Doc אשר מוחזר מהמחלקה Ranker כאשר הDoc הראשון הוא הרלוונטי ביותר.

```
private Index index;
```

מצביע לIndex.

```
private HashMap<String,Term> dictionary;
```

מבנה נתונים המכיל בתוכו את המילון מהIndex כאשר הKey הוא שם הTerm והValue הוא האובייקט של הTerm עצמו.

```
private HashMap<String,String> postingNames;
```

מבנה נתונים המחזיר בתוכו את כל שמות קבצי הPosting כאשר הKey שלו הוא האות הראשונה בשם של מסמך קובץ הPosting והValue הוא השם של קובץ הPosting.

```
private boolean stem;
```

שדה בוליאני אשר בתוכו נשמר האם משתמש בחר לבצע stemming או לא.

שיטות המחלקה:

```
public Searcher(boolean semantic,Index index){
```

זוהי פונקציית הConstructor של המחלקה, הוא מקבלת מצביע אל Index וערך בוליאני האם המשתמש בחר לבצע ניתוח סמנטי או לא.

```
public void initPostingNames(){
```

פונקציה המאתחלת את מבנה הנתונים postingNames בערכים שלו, פונקציה זאת נקראת מהConstructor.

```
private String[] splitRegularQuery(String query){
```

זוהי פונקציה אשר מפרקת את הquery למילים לפי רווחים ומחזירה אותו כמערך של String.

```
private String getLineFromPosting(String postingName,int line){
```

זוהי פונקציה אשר מקבלת מספר שורה ושם של קובץ Posting ואחראית להחזיר את השורה מהקובץ המדובר.

```
private HashMap<String,Integer> getTermFromPosting(String term, int line){
```

זוהי פונקציה אשר מחזירה עבור term ספציפית מהQuery מבנה נתונים המכיל את כל המסמכים הרלוונטיים לאותו term מהPosting ועבור כל מסמך את כמות הפעמים שהterm מופיע בו.

```
public void writeToResultsFile(String queryNum,List<Doc> documents,String
trec_eval_path){
```

זוהי פונקציה שתפקידה הוא לרשום את התוצאות של המסמכים שאוחזרו לתוך קובץ results בפורמט שהתוכנה trec_eval תוכל לקרוא ולהשתמש בו.

פונקציה זו מקבלת נתיב לקובץ בו נרצה לכתוב את התוצאות, את מספר הquery ואת רשימת 50 המסמכים הרלוונטיים ביותר.

```
public List<Doc> startQuerySearch(String queryNum,String query,String
trec_eval_path){
```

פונקציית העבודה הראשית של המחלקה, הפונקציה מקבלת את הquery ואת הנתיב אליו נרצה לרשום את התוצאות שקיבלנו מתהליך האיחזור.

הפונקציה בעצם משתמשת בכל הפונקציות במחלקה בכדי להביא את השורה מהPosting וכל המידע עבור הquery אשר שמרנו ובעצם שולחת את המידע שנאסף אל מחלקת Ranker בכדי לבצע חישוב ודירוג של רלוונטיות המסמכים.

לבסוף הפונקציה מחזירה רשימה של 50 מסמכים ממוינים על פי רמת הרלוונטיות שלהם לquery כאשר המסמך הראשון ברשימה הוא הרלוונטי ביותר.

```
public String[] splittedQueryWithStem(String query){
```

זוהי פונקציה שתפקידה הוא לפרק את הquery במקרה שהמשתמש בחר לבצע stemming, תפקידה הוא לבצע stemming עבור כל מילה בquery.

```
public String getSemanticWords(String[] queryWords){
```

זוהי פונקציה שתפקידה הוא להוסיף מילים בעלות משמעות סמנטית זהה אל הquery, היא מקבלת את המילים מהquery מופרדות ובעצם מחזיקה String אשר אותו נוסיף אל הquery.

הפונקציה עושה שימוש במחלקות אשר הוספנו בעזרת jar אשר נטען אל התוכנית.

הסברים נוספים:

במחלקה זו נעשה שימוש בקוד פתוח אשר הגיע מקובץ Jar שנטען אל תוך המערכת ואנו עושים בו שימוש.

הקובץ יצורף כקובץ zip אל תיקיית הפרויקט.

הקובץ הועבר במחלקה ונעשה בו שימוש מחלקתי נרחב בכדי שנוכל לאפשר מימוש מודל סמנטי עבור שאילתה ללא צורך בחיבור לאינטרנט, כלומר כאשר המערכת פועלת offline.

הקוד הוא קוד פתוח של medallia, המודל מממש את האלגוריתם word2Vec בעזרת שיטת n-gramming, כלומר בהינתן מילה האלגוריתם יפרק את המילה לתתי מילים וימצא את המילים שהכי קרובות סמנטית למילה, עבור כל מילה בquery הוספנו עד 10 מילים דומות.

מחלקת GUI

על מחלקת GUI הרחבתי בדו"ח של חלק א' ועל כן לא ארחיב ואוסיף מידע כפול.

שדות המחלקה שהוספנו בחלק זה:

```
public javafx.scene.control.CheckBox semantic;
```

שדה זה הוא קישור אל קוביית הcheckBox של הבחירה הסמנטית בתוכנית ובתוכו נשמר הערך האם המשתמש בחר לבצע ניתוח סמנטי או לא.

```
public javafx.scene.control.TextField tx_query;
```

שדה זה מקושר באופן ישיר אל תיבת הטקסט של כתיבת השאילתה הבודדת.

```
private String trec_eval_path;
```

שדה זה הוא String המכיל את הכתובת לתיקיית trec_eval בה נרצה לשמור את המידע שנאסף מהמנוע על הקבצים בקובץ result

```
private Searcher searcher;
```

שדה זה הוא מצביע למחלקת Searcher.

```
public javafx.scene.control.TextField tx_trec_eval;
```

שדה זה מקושר באופן ישיר אל תיבת הטקסט המכילה את הכתובת אל תיקיית trec_eval.

```
@FXML
public ListView<String> listView;
```

שדה זה מקושר באופן ישיר אל הlistView הקיים בתוכנית אשר בו מוצגים כל המסמכים שאוחזרו.

שיטות המחלקה שהוספנו בחלק זה:

```
private void resetResults(){
```

פונקציה אשר מאפסת את קובץ הresults בתיקיית הtrec_eval בכל קריאה שלה .

```
public void runOneQuery() {
```

זוהי פונקציה אשר תפקידה הוא לרוץ כאשר מופעלת שאילתה בודדת על המערכת מתיבת הטקסט בה ניתן לכתוב שאילתה.

```
public void getTrecEvalBrowser(){
```

זוהי פונקציה אשר פותחת DirectoryChooser עבור בחירת תיקיית trec_eval.

```
public void showEntities(){
```

זוהי פונקציה אשר מציגה את חמשת הישויות הרלוונטיות ביותר עבור מסמך מסוים כאשר המשתמש מקיש על שם המסמך בטבלה.

```
public void chooseQueriesFile(){
```

זוהי פונקציה אשר תפקידה הוא לרוץ על קובץ שאילתות מהקובץ אותו בחר המשתמש.

פונקציה זו נקראת כאשר המשתמש לוחץ על כפתור Browse queries and run.

הערכה של המנוע בלי Stemming:

#Q	Words(Q)	Precision(Q)	Recall(Q)	Precision@5	Precision@15	Precision@30	Precision@50
351	Falkland petroleum exploration What information is available on petroleum exploration in the South Atlantic near the Falkland Islands	0.46	0.48	0	0.2	0.3667	0.46
352	British Chunnel impact What impact has the Chunnel had on the British economy and/or the life style of the British	0.18	0.036	0.2	0.133	0.2667	0.18
358	blood-alcohol fatalities What role does blood-alcohol level play in automobile accident fatalities	0.44	0.4	0.2	0.4667	0.533	0.44
359	mutual fund predictors Are there reliable and consistent predictors of mutual fund performance	0.06	0.107	0	0	0.1	0.06
362	human smuggling Identify incidents of human smuggling	0.16	0.2	0.2	0.2667	0.1333	0.16
367	piracy	0.26	0.07	0.4	0.33	0.3	0.26

	What modern instances have there been of old fashioned piracy, the boarding or taking control of boats						
373	encryption equipment export Identify documents that discuss the concerns of the United States regarding the export of encryption equipment	0.04	0.125	0	0.133	0.067	0.04
374	Nobel prize winners Identify and provide background information on Nobel prize winners	0.3	0.073	0.6	0.33	0.367	0.3
377	cigar smoking Identify documents that discuss the renewed popularity of cigar smoking	0.22	0.305	0	0.133	0.3	?
380	obesity medical treatment Identify documents that discuss medical treatment of obesity	0.08	0.57	0	0.2	0.1	0.08
384	space station moon Identify documents that	0.16	0.156	0.2	0.2	0.167	0.16

	discuss the building of a space station with the intent of colonizing the moon						
385	hybrid fuel cars Identify documents that discuss the current status of hybrid automobile engines cars fueled by something other than gasoline only	0.34	0.2	0	0	0.1667	0.34
387	radioactive waste Identify documents that discuss effective and safe ways to permanently handle long-lived radioactive wastes	0.24	0.164	0.4	0.133	0.2	0.24
388	organic soil enhancement Identify documents that discuss the use of organic fertilizers composted sludge ash vegetable waste microorganisms as soil enhancers	0.18	0.18	0.2	0.133	0.2	0.18
390	orphan drugs Find documents that discuss issues	0.2	0.082	0.6	0.33	0.3	0.2

	associated with so-called orphan drugs that is, drugs that treat diseases affecting relatively few people						

Average precision (non-interpolated) over all rel docs
0.0606

Queryid (Num): 15

Total number of documents over all queries

Retrieved: 750

Relevant: 1241

Rel_ret: 166

הערכת המנוע עם Stemming:

#Q	Words(Q)	Precision(Q)	Recall(Q)	Precision@5	Precision@15	Precision@30	Precision@50
351	Falkland petroleum exploration What information is available on petroleum exploration in the South Atlantic near the Falkland Islands	0.46	0.48	0.2	0.533	0.433	0.46
352	British Chunnel impact What impact has the Chunnel had on the British economy and/or the life style of the British	0.18	0.036	0.2	0.133	0.2667	0.18
358	blood-alcohol fatalities What role does blood-alcohol level play in automobile accident fatalities	0	0	0	0	0	0
359	mutual fund predictors Are there reliable and consistent predictors of mutual fund performance	0.06	0.107	0	0.06	0.05	0.06
362	human smuggling Identify incidents of human smuggling	0.16	0.205	0.2	0.13	0.13	0.16
367	piracy	0.36	0.097	0	0.2667	0.3	0.36

	What modern instances have there been of old fashioned piracy, the boarding or taking control of boats						
373	encryption equipment export Identify documents that discuss the concerns of the United States regarding the export of encryption equipment	0.12	0.375	0.4	0.33	0.2	0.12
374	Nobel prize winners Identify and provide background information on Nobel prize winners	0.3	0.073	0.6	0.33	0.3	0.3
377	cigar smoking Identify documents that discuss the renewed popularity of cigar smoking	0.24	0.333	0	0.133	0.133	0.24
380	obesity medical treatment Identify documents that discuss medical treatment of obesity	0.08	0.57	0	0.2	0.1	0.08
384	space station moon Identify documents that	0.18	0.176	0.2	0.2	0.2	0.18

	discuss the building of a space station with the intent of colonizing the moon						
385	hybrid fuel cars Identify documents that discuss the current status of hybrid automobile engines cars fueled by something other than gasoline only	0.38	0.22	0	0	0.233	0.38
387	radioactive waste Identify documents that discuss effective and safe ways to permanently handle long-lived radioactive wastes	0.18	0.123	0.2	0.2	0.133	0.18
388	organic soil enhancement Identify documents that discuss the use of organic fertilizers composted sludge ash vegetable waste microorganisms as soil enhancers	0.3	0.3	0.2	0.4	0.433	0.3
390	orphan drugs Find documents that discuss issues	0.26	0.1	0.4	0.3	0.4	0.26

	associated with so-called orphan drugs that is, drugs that treat diseases affecting relatively few people						

Average precision (non-interpolated) over all rel docs

0.0635

Queryid (Num): 15

Total number of documents over all queries

Retrieved: 750

Relevant: 1241

Rel_ret: 163

דוגמא עבור אחזור ישויות, לשני המסמכים ישות אחת: