Πανεπηστίμιο Κρήτης

Τμήμα Επιστήμης Υπολογιστών

HY463 Συστήματα Ανάκτησης Πληροφοριών

Εξάμηνο: Άνοιξη 2021

Στοιχεία:

Μέλος	1ο
Ονοματεπώνυμο	Νικόλαος Γουνάκης
AM	3932
Email	csd3932@csd.uoc.gr

Πίνακας Περιεχομένων

Εισαγωγή

Το project υλοποιήθηκε σε python και για να τρέξει θα χρειαστεί να εγκαταστήσετε ότι βιβλιοθήκη περιέχει το αρχείο requirements.txt με την εντολή:

```
$ pip install -r requirements.txt .
```

Γενικά υλοποιήθηκε και το απλό indexing το οποίο είναι εξερεπικά γρήγορο αλλά καταναλώνει πολύ μνήμη και το partial indexing το οποίο είναι αργό χρησιμοποιέι λιγότερη μνήμη αλλα μπορεί να ευρετηριάσει μεγαλύτερες συλλογές εγγράφων.

Εφόσον το project έγινε σε διαφορεπική γλώσσα απο την προτεινόμενη (java) χρειάστηκε να υλοποιήσω κάποια απο τα δοθέντα κομμάπια κώδικα απο την αρχή ή να χρησιμοποιήσω βιβλιοθήκες που κάνουν παρόμοια δουλειά.

Διαδικασία Ευρετηρίασης

Για απλό indexing:

```
$ python app.py -index
```

Για partial indexing:

```
$ python app.py -pindex
```

Στην συνέχεια ανοίγει γραφική διεπαφή για την επιλογή φακέλου και είναι η μόνη γραφική διδεπαφη που υπάρχει προς το παρόν. Τα υπόλοιπα γίνονται μέσο του τερματικού.

Διάβασμα Αρχείων

Εφόσον επιλέξουμε φάκελο το πρόγραμμα αρχίζει να διαβάζει αναδρομικά όλους του υπο-φακέλους και διαβάζει μόνο τα αρχεία με κατάληξη `.nxml` . Τα υπόλοιπα τα αγνοεί.

Στο αρχείο `readxml.py` φαίνεται η υλοποίηση , όπου το πρόγραμμα διαβάζει τα απαραίπητα tags. Στην συνέχεια περνάνε απο tokenizer και χρησιμοποιούνται για να φιαχτεί ένα document object (`Document.py`) το οποίο αναπαρηστά ένα έγγραφο.

Αξίζει να αναφερθεί οπι κρατιέται πληροφορία σε ποία tags εμφανίζεται κάθε λέξη και πόσες φορές μέσα σε κάθε document object.

Tokenizing

Η υλοποίηση βρίσκεται στο `tokenizer.py` . Η συνάρτηση που κάνει tokenize παίρνει ως παράμετρο ένα string η έναν πίνακα απο strings και επιστρέφει έναν πίνακα με tokens όπου κάθε token αποτελείται μόνο απο αλφαριθμητικούς χαρακτήρες.

Αφαίρεση Stopwords

Η υλοποίηση βρίσκεται πάλι στο `tokenizer.py` . Διαβάζει τα δοθέντα αρχεία `stopwordsEn.txt` και `stopwordsGr.txt` χρησιμοποιείται απο την συνάρτηση `tokenize(s)` για να τα αφαιρεί.

Stemming

Το stemming γίνεται μέσα στον constructor του document object (`Document.py`) κατα τον υπολογισμό συχνοτήτων των λέξεων.

Για το stemming χρησιμοποιήθηκε η βιβλιοθήκη `nltk` όπου υποστιρίζει αγγλικό stemming

Ανεστραμμένο Ευρετήριο

Και με τις δύο τεχνικές indexing παράγεται το ίδιο ανεστραμμένο ευρετήριο.

DocumentsFile

Έχει τη μορφή:

```
doc_id  path  norm
```

PostingFile

Έχει τη μορφή:

```
doc_id  tf  appearances  pointer to DocumentsFile
```

όπου `appearances` : `{'abstract': [89, 94, 96, 98], 'body': [624, 722]}` ένα dictionary που γράφει σε ποιό σημείο μέσα tag εμφανίζεται ο όρος και απο το length του array μπορούμε να μάθουμε και πόσες φορές.

VocabularyFile

Έχει τη μορφή:

```
term_id  df  pointer to PostingFile
```

όπου `term_id` : το string που αναπαρηστά έναν όρο

Απλό Indexing

1. Το πρόγραμμα ξεκινά να διαβάζει όλα τα documents απο έναν φάκελο.
2. Στην συνέχεια απο όλα τα documents κάνει extract τα terms και δημιουργεί το vocabulary
3. Έπειτα εφόσον έχει όλα τα documents και το vocabulary στην μνήμη ξεκινά να παράγει το inverted file

Partial Indexing

1. Το πρόγραμμα ξεκινά να διαβάζει αρχεία απο έναν φάκελο
2. Όταν η μνήμη φτάσει στο 80% τότε ξεκινά να παράγει ένα partial inverted file
3. Επαναλαμβάνονται τα βήματα 1 και 2 μέχρις ότου να έχουν διαβαστεί όλα τα αρχεία του φακέλου
4. Εφόσον έχουν διαβαστεί όλα τα αρχεία του φακέλου και έχου παραχθεί όλα τα partial inverted files , τότε ξεκινάει η διαδικασία του merging όπως περιγραφεται στο δοσμένο pdf
   `Partial Indexing and Merging`

Αποτιμητής Ερωτήσεων

Για να τρέξει το πρόγραμμα σε query evaluation mode το τρέχουμε χωρίς κανένα argument:

```
$ python app.py
```

Στην συνέχεια φορτώνεται το Vocabulary στην μνήμη και το σύστημα ρωτάει τον χρήστη να επιλέξει ανάμεσα σε:

1. diagnosis
2. test
3. treatment

Εφόσον επιλέξει τότε του δίνεται η ευκαιρία να εισάγει summary ή description.

Έπειτα εφόσον πατήσει `enter` ο χρήστης τότε γίνεται evaluation του query χρησιμοποιόντας το `Διανυσματικό Μοντέλο` και επιστρέφονται όλα τα έγγραφα τα οποία περιέχουν όρους απο το query σε αύξουσα σειρα με βάση το score.

Γενικά , δεν έχει υλοποιηθεί ακόμα κάποιος μηχανισμός για τον διαχωρισμό των εγγράφων στις 3 παραπάνω κατηγορίες διότι παραπάνω έμφαση έδωσα στο να υλοποιήσω και τις 2 τεχνικές indexing. Θα υλοποιηθεί όμως στην επόμενη φάση που είναι πιο χρήσιμο.

Μετρήσεις

Ευρετηρίαση

Method	MiniCollection (4.89 MB)	MedicalCollection/00 (309 MB)	Medical Collection (4.56 GB)
Simple Indexing	6.148648262023926 s	480.54263734817505 s	-
Partial Indexing	6.488290309906006 s (no need for merging)	796.763519525528 s	doc analysis: 7852.338171958923

Αποτίμηση Ερωτήσεων

Single Queries

Query	CollectionIndex size	Time
64-year-old woman with uncontrolled diabetes, now with an oozing, painful skin lesion on her left lower leg.	3.06 MB	0.03461456298828125 s
-	220 MB	1.91103196144104 s
-	-	-

Evaluation with 30 topics

Για την αξιολόγηση χρησιμοποιήθηκαν οι μετρικές `bpref` , `NDCG'` , `AveP'` .

Παρατηρήθηκε στο `bpref` κάποιες φορές το αποτέλεσμα έβγαινε αρνητικό λόγο των περισσότερων irrelevant εγγράφων σε μεγαλύτερο rank απο relevant. Σε αυτήν την περίπτωση έθεσα το bpref ίσο με 0.

Γενικά ο υπολογισμός των μετρικών έγινε με βάση τα judged documents δηλαδή όσα documents ήταν not-judged δεν υπολογιζόντουσαν στο evaluation απο απάντηση του που έδινε το σύστημα.

MedicalCollection/00

```
Topic: 1        bpref: 0.375    AveP: 0.5111111111111111        NDCG: 0.5366657618660127
Topic: 2        bpref: 0.8163265306122449       AveP: 0.8773754023754023        NDCG: 0.79244061144250
Topic: 3        bpref: 0        AveP: 0 NDCG: 0
Topic: 4        bpref: 1.0      AveP: 1.0       NDCG: 1.0
Topic: 5        bpref: 0        AveP: 0.23722943722943723       NDCG: 0.375318572366445
Topic: 6        bpref: 0.7160493827160495       AveP: 0.663888888888889 NDCG: 0.654770443118796
Topic: 7        bpref: 0.4444444444444445       AveP: 0.5333333333333333        NDCG: 0.67973105000376
Topic: 8        bpref: 0.7346938775510204       AveP: 0.6568027210884353        NDCG: 0.73651831949269
Topic: 9        bpref: 0        AveP: 0 NDCG: 0
Topic: 10       bpref: 0        AveP: 0 NDCG: 0
Topic: 11       bpref: 0.609375 AveP: 0.6364853896103896        NDCG: 0.6604722979153912
Topic: 12       bpref: 0        AveP: 0.15476190476190477       NDCG: 0.37007816335371563
Topic: 13       bpref: 0.16326530612244897      AveP: 0.466060606060606 NDCG: 0.5297194146103197
Topic: 14       bpref: 0.6111111111111112       AveP: 0.6763888888888889        NDCG: 0.91028591564852
Topic: 15       bpref: 0.16666666666666674      AveP: 0.42025335775335776       NDCG: 0.60565141335962
Topic: 17       bpref: 0.0      AveP: 0.5       NDCG: 0.6309297535714574
Topic: 18       bpref: 0.5      AveP: 0.65      NDCG: 0.8244017426339619
Topic: 19       bpref: 0.5625   AveP: 0.6041666666666666        NDCG: 0.549806531001946
Topic: 20       bpref: 0.5200000000000001       AveP: 0.5833333333333333        NDCG: 0.58756745115432
Topic: 22       bpref: 0        AveP: 0.5555555555555555        NDCG: 0.5510527863501852
Topic: 23       bpref: 0        AveP: 0.17727272727272728       NDCG: 0.35911066294785976
Topic: 24       bpref: 0.22222222222222218      AveP: 0.7666666666666666        NDCG: 0.69722430413866
Topic: 25       bpref: 0        AveP: 0 NDCG: 0
Topic: 26       bpref: 0.13888888888888887      AveP: 0.4074786324786324        NDCG: 0.63852723898407
Topic: 27       bpref: 0.85     AveP: 1.0       NDCG: 0.8457207304658277
Topic: 28       bpref: 0.75     AveP: 0.8333333333333333        NDCG: 0.6885288809404666
Topic: 29       bpref: 0.653061224489796        AveP: 0.6099162742019885        NDCG: 0.74659547015535
Topic: 30       bpref: 0.75     AveP: 0.8333333333333333        NDCG: 0.9197207891481876
Average score: 0.5504741661036194       Max score: 1.0  Min score: 0.12328401218674027
```

Score with more weight in 'title' and 'abstract'

```
Average score: 0.5132825013622625       Max score: 1.0  Min score: 0.12328401218674027
```

MiniCollection

```
Topic: 1        bpref: 0.890625 AveP: 0.9068813131313131        NDCG: 0.8934861620193163
Topic: 2        bpref: 0        AveP: 0 NDCG: 0
Topic: 3        bpref: 0        AveP: 0 NDCG: 0
Topic: 4        bpref: 0        AveP: 0 NDCG: 0
Topic: 5        bpref: 0.9375   AveP: 0.95      NDCG: 0.7369302591988794
Topic: 6        bpref: 1.0      AveP: 1.0       NDCG: 1.0
Topic: 7        bpref: 0        AveP: 0 NDCG: 0
Topic: 8        bpref: 0        AveP: 0 NDCG: 0
Topic: 9        bpref: 0        AveP: 0 NDCG: 0
Topic: 10       bpref: 0        AveP: 0 NDCG: 0
Topic: 11       bpref: 0.3333333333333333       AveP: 0.4777777777777777        NDCG: 0.52745570886818
Topic: 12       bpref: 0.0      AveP: 0.5       NDCG: 0.6309297535714574
Topic: 13       bpref: 0.75     AveP: 0.8333333333333333        NDCG: 0.9197207891481876
Topic: 14       bpref: 0.84375  AveP: 0.8015241702741702        NDCG: 0.8820718982900894
Topic: 15       bpref: 1.0      AveP: 1.0       NDCG: 1.0
Topic: 16       bpref: 0        AveP: 0 NDCG: 0
Topic: 17       bpref: 0.875    AveP: 0.8875    NDCG: 0.9805704719000117
Topic: 18       bpref: 0        AveP: 0 NDCG: 0
Topic: 19       bpref: 1.0      AveP: 1.0       NDCG: 1.0
Topic: 20       bpref: 1.0      AveP: 1.0       NDCG: 1.0
Topic: 21       bpref: 0        AveP: 0 NDCG: 0
Topic: 22       bpref: 0        AveP: 0 NDCG: 0
Topic: 23       bpref: 0        AveP: 0 NDCG: 0
Topic: 24       bpref: 0        AveP: 0 NDCG: 0
Topic: 25       bpref: 0        AveP: 0 NDCG: 0
Topic: 26       bpref: 0        AveP: 0 NDCG: 0
Topic: 27       bpref: 0.765625 AveP: 0.8430215617715617        NDCG: 0.9452837222996112
Topic: 28       bpref: 0.4375   AveP: 0.6361111111111111        NDCG: 0.8106202559084549
Topic: 29       bpref: 0        AveP: 0 NDCG: 0
Topic: 30       bpref: 0        AveP: 0 NDCG: 0
Average score: 0.8204244005624819       Max score: 1.0  Min score: 0.3769765845238191
```

Score with more weight in 'title' and 'abstract'

```
Average score: 0.7977463012448565       Max score: 1.0  Min score: 0.3769765845238191
```

Επίλογος

Ένα πρόβλημα που άργησα να παρατηρήσω και αποτρέπει το partial indexing να δουλέψει κανονικά είναι οτι η βιβλιοθήκη που χρησιμοποιώ για random access file (linecache) φορτώνει το αρχειό στην μνήμη και χρησιμοποιεί cache για να κάνει fetch γρήγορα lines απο το αρχείο , με αποτέλεσμα να γεμίζει η μνήμη κατα το merging της μεγάλης συλλογής. Οπότε στην επόμενη φάση θα πρέπει να κατασκευάσω άλλη μέδοθο κρατόντας το offset κάθε entry στο inverted file.

Δεν χρειάστηκε να αλλάξω κάπι καθώς πειραματιζόμουν με τις εμφανήσεις σε συγκεκριμένο tag μέσα στο document (πχ. abstract , body , title) και έδινα παραπάνω βάρος σε λέξεις που εμφανιζόντουσαν μέσα σε αυτά τα tags , όμως δεν αποδείχθηκε να αυξάνει το score.

Τέλος το πεδίο type απο τα topics δεν χρησιμοποιήθηκε.

Αναφορές

1. pip
2. nltk
3. linecache