

Ανάκτηση Πληροφοριών

1^η εργασία

Αλβανάκη Παρασκευή

AM 57286

A ερωτημα

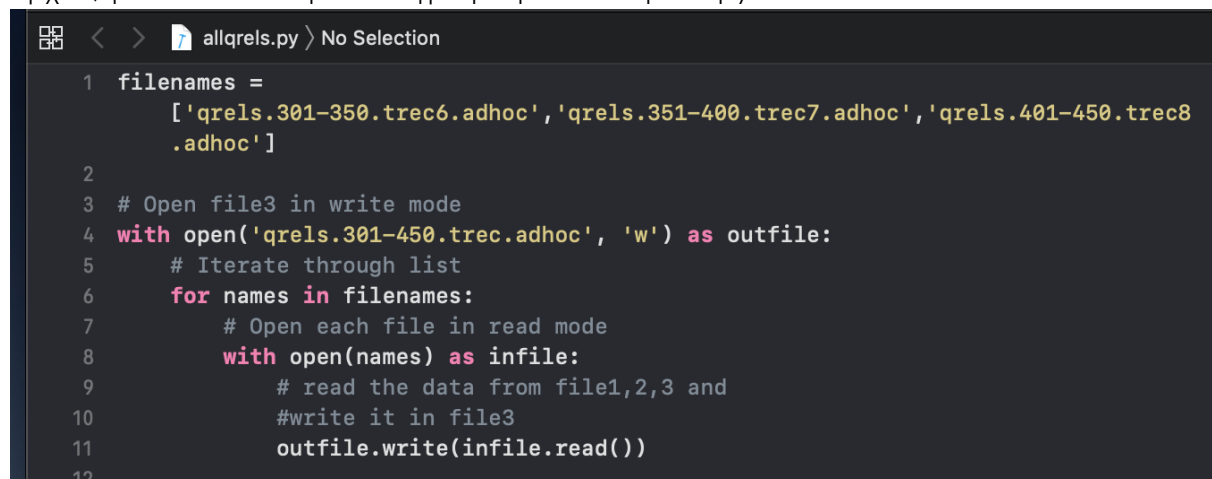
A1. Σχηματίστε και τρέξτε 150 ερωτήσεις (queries), με τρεις διαφορετικούς τρόπους χρησιμοποιώντας τα πεδία title, title+desc, title+desc+narr

Με βάση το tutorial ακολουθήσαμε την παρακάτω διαδικασία :

Αρχικά κατασκευάζουμε το index, αξιοποιώντας την εντολή IndriBuildIndex

IndriBuildIndex.parameter.file.EXAMPLE. Στη συνέχεια δημιουργούμε με χρήση της εντολής dumpIndex ένα αναλυτικό ευρετήριο. Αναζητούμε τις ερωτήσεις με το αρχείο που υπάρχει ως παράδειγμα μετά την παραμετροποίηση του με τη χρήση της εντολής IndriRunQuery IndriRunQuery.queries.file.301-450-titles-only.EXAMPLE

A1.1 Για να τρέξουμε τις 150 ερωτήσεις, αρχικά ομαδοποιούμε τα 150 αρχεία σε ένα ενιαίο αρχείο, για να το κάνουμε αυτό γράφουμε ένα script σε python:



```
1 filenames =  
    ['qrels.301-350.trec6.adhoc', 'qrels.351-400.trec7.adhoc', 'qrels.401-450.trec8  
    .adhoc']  
2  
3 # Open file3 in write mode  
4 with open('qrels.301-450.trec.adhoc', 'w') as outfile:  
5     # Iterate through list  
6     for names in filenames:  
7         # Open each file in read mode  
8         with open(names) as infile:  
9             # read the data from file1,2,3 and  
10             #write it in file3  
11             outfile.write(infile.read())  
12
```

όπου αρχικά εισάγουμε τα αρχεία που θέλουμε να ομαδοποιήσουμε και έπειτα με την εντολή with open δηλώνουμε πού θα γραφούν ομαδοποιημένα όλα τα αρχεία. Σε αυτό το νέο αρχείο εμπεριέχονται από όλα τα αρχεία το title, το description και το narrative.

A1.2 Στη συνέχεια δημιουργούμε scripts που αντιστοιχα απομονώνουν και παράγουν μόνο το title ή title+desc ή title+desc+narr

Δημιουργούμε το αρχείο που ενώνει τα υπάρχοντα αρχεία έτσι ώστε απο 301-450 να είναι όλα μαζί τα descriptions,title και narrative

```
all_trecs (7).py > No Selection
1 filenames = ['topics.301-350.trec6','topics.351-400.trec7','topics.401-450.trec8']
2
3 # Open file3 in write mode
4 with open('topics.301-450.trec', 'w') as outfile:
5     # Iterate through list
6     for names in filenames:
7         # Open each file in read mode
8         with open(names) as infile:
9             # read the data from file1,2,3 and
10             # write it in file3
11             outfile.write(infile.read())
12
13         # Add '\n' to enter data of file2
14         # from next line
15         outfile.write("\n")
16
```

A1.2.1 title+desc

i. Αρχικά, δημιουργούμε ένα αρχείο που περιέχει μόνο τα σημεία του κειμένου που αφορούν τα titles και τα descriptions: με βάση το αρχείο topics.301-450. trec επιτρέπουμε την αντιγραφή στο νέο αρχείο titles_descriptions.trec μόνο του title και του description

```
enosi_arxeiwn_topic_qreals (7).py > No Selection
1 import re
2
3 filename = 'topics.301-450.trec'
4 # Open file3 in write mode
5 # with open('mydocument.trec', 'w') as outfile:
6 with open(filename) as infile, open('titles_descriptions.trec', 'w') as outfile:
7     copy = False
8     for line in infile:
9
10         if bool(re.search("<narr>", line)):
11             copy = False
12         if bool(re.search("<title>", line)):
13             outfile.write('<top> ')
14             copy = True
15         if copy:
16             outfile.write(line)
```

ii. Δημιουργούμε ένα νέο έγγραφο το my.trec που αφαιρούμε τα κενά μεταξύ των λέξεων του εγγράφου titles_descriptions.trec

```
removing_empty_strings (4).py > No Selection
1 import re
2 fin = "titles_descriptions.trec"
3 fout = open("my.trec", "wt")
4
5 with open(fin, "r") as f:
6     for line in f:
7         cleanedLine = line.strip()
8         if cleanedLine: # is not empty
9             fout.write(cleanedLine)
10             fout.write(" ")
```

iii. Για το title+desc δε χρειάζεται να αφαιρέσουμε μέρη του κειμένου, παρά μόνο τους διαχωρισμούς με τις λέξεις "<title>","<desc>" αντίστοιχα. Αυτή η διαδικασία πραγματοποιείται αυτοματοποιημένα κατά αυτόν τον τρόπο:

```

removing_title_desc_ (2).py } No Selection
1 import re
2 fin = "my.trec"
3 fout = open("my2.trec", "wt")
4 filterlist=["<title>", "<desc>"]
5 words=[]
6 with open(fin, "r") as f:
7     for line in f:
8         wordlist1 = line.split()
9         for word in wordlist1:
10             if word.lower() not in filterlist:
11                 words.append(word)
12 s = " "
13 s=s.join(words)
14 fout.write(s)
15

```

όπως φαίνεται και στο script αντιγράφεται το περιεχόμενο του my.trec στο my2.trec έχοντας διαχωρίσει τις σειρές και τις λέξεις μεταξύ τους.

iii. Τελικά δημιουργούμε το αρχείο που θα εκτελεστεί με την εντολή IndriRunQuery και φροντίζουμε να αφαιρεθούν τα σημεία στίξης και να μετατραπούν όλες οι λέξεις με κεφαλαία γράμματα σε μικρά:

```

a_ervtima_title_desc (2).py } No Selection
1 fin = "my2.trec"
2 fout = open("IndriRunQuery.queries.file.301-450-titles-desc-only.EXAMPLE", "wt")
3 filterlist=["<top>"]
4
5 fout.write('<parameters> \n')
6 fout
7     .write
8     ('<index>/home/evi/Downloads/IR-2019-2020-Project-1/indices/example</index>\n'
9     )
10 fout.write('<rule>method:dirichlet,mu:1000</rule> \n')
11 fout.write('<count>1000</count> \n')
12 fout.write('<trecFormat>true</trecFormat> \n')
13 fout.write('<query> <type>indri</type> <number>301</number> <text>')
14 i=300
15 with open('my2.trec', "r") as f:
16     for line in f:
17         line = line.replace(',', ' ')
18         line = line.replace('.', ' ')
19         line = line.replace(':', ' ')
20         line = line.replace('?', ' ')
21         line = line.replace('(', ' ')
22         line = line.replace(')', ' ')
23         wordlist1 = line.split()
24         for word in wordlist1:
25             words = []
26             if word.lower() not in filterlist:
27                 words.append(word)
28                 s = " "
29                 s = s.join(words)
30                 fout.write(s)
31                 fout.write(" ")
32             else:
33                 fout.write('</text> </query> \n')
34                 i+=1
35                 fout.write('<query> <type>indri</type> <number>{i}</number>
36                     <text>'.format(i))
37 fout.write('</text> </query> \n')
38 fout.write('</parameters> \n')
39 print(type(fout))
40

```

1.2.2 title+desc+narr

ii. Αρχικά, δημιουργούμε ένα αρχείο που περιέχει μόνο τα σημεία του κειμένου που αφορούν τα titles και τα descriptions: με βάση το αρχείο topics.301-450.trec και επιτρέπουμε την αντιγραφή στο νέο αρχείο titles_descriptions.trec του title, του description και του narr

```
import re

filename = 'topics.301-450.trec'
# Open file3 in write mode
# with open('mydocument.trec', 'w') as outfile:
with open(filename) as infile, open('titles_descriptions.trec', 'w') as outfile:
    copy = False
    for line in infile:

        if bool(re.search("</top>", line)):
            copy = False
        if bool(re.search("<title>", line)):
            outfile.write('<top> ')
            copy = True
        if copy:
            outfile.write(line)
```

iii. Για το title+desc δε χρειάζεται να αφαιρέσουμε μέρη του κειμένου, παρά μόνο τους διαχωρισμούς με τις λέξεις "<title>", "<desc>", "<narr>" αντίστοιχα. Αυτή η διαδικασία πραγματοποιείται αυτοματοποιημένα κατά αυτόν τον τρόπο:

```
> removing_title_desc_narr_ (6).py > No Selection

import re
fin = "my.trec"
fout = open("my2.trec", "wt")
filterlist=["<title>", "<desc>", "<narr>"]
words=[]
with open(fin, "r") as f:
    for line in f:
        wordlist1 = line.split()
        for word in wordlist1:
            if word.lower() not in filterlist:
                words.append(word)
s = " "
s=s.join(words)
fout.write(s)
```

iv. Τελικά δημιουργούμε το αρχείο που θα εκτελεστεί με την εντολή IndriRunQuery και φροντίζουμε να αφαιρεθούν τα σημεία στίξης και να μετατραπούν όλες οι λέξεις με κεφαλαία γράμματα σε μικρά

```

finalquery-titles-desc-narr (8).py — Edited
finalquery-titles-desc-narr (8).py No Selection
1 fin = "my2.trec"
2 fout = open("IndriRunQuery.queries.file.301-450-titles-desc-narr-only.EXAMPLE",
3 "wt")
4 filterlist=["<top>"]
5 fout.write('<parameters> \n')
6
7 .write
8 ('<index>/home/evi/Downloads/IR-2019-2020-Project-1/indices/example</index>\n'
9 )
10
11 fout.write('<rule>method:dirichlet,mu:1000</rule> \n')
12 fout.write('<count>1000</count> \n')
13 fout.write('<trecFormat>true</trecFormat> \n')
14 fout.write('<query> <type>indri</type> <number>301</number> <text>')
15 i=300
16 with open('my2.trec', "r") as f:
17     for line in f:
18         line = line.replace(',', ' ')
19         line = line.replace('.', ' ')
20         line = line.replace(':', ' ')
21         line = line.replace('?', ' ')
22         line = line.replace('(', ' ')
23         line = line.replace(')', ' ')
24         line = line.replace("'", ' ')
25         line = line.replace('"', ' ')
26         line = line.replace("\n", ' ')
27         line = line.replace("; ", ' ')
28         line = line.replace("/", ' ')
29         wordlist1 = line.split()
30         for word in wordlist1:
31             words = []
32             if word.lower() not in filterlist:
33                 words.append(word.lower())
34                 s = " "
35                 s = s.join(words)
36                 fout.write(s)
37                 fout.write(" ")
38             else:
39                 fout.write('</text> </query> \n')
40                 i+=1
41                 fout.write('<query> <type>indri</type> <number>{}</number>
42                             <text>'.format(i))
43         fout.write('</text> </query> \n')
44     fout.write('</parameters> \n')
45     print(type(fout))

```

1.2.3. titles

Όσον αφορά τα results δε χρειάζεται να κάνουμε κάποια αλλαγή στα αρχεία για να παράξουμε το αρχείο IndriRunQuery, αφού δίνεται ήδη, όπως και το αντίστοιχο trec αρχείο. Για αυτό το μέρος της εργασίας προχωράμε κατευθείαν στις μετρήσεις.

A2. Στα 3 αρχεία qrels.XXX-XXX.trecX.adhoc δίνονται οι κρίσεις συνάφειας (relevance judgments) των παραπάνω ερωτήσεων αναφορικά με τις συλλογές. Αξιολογήστε την ποιότητα ανάκτησης με τους αριθμητικούς μέσους όρους των μετρικών Precision@10, R-Precision, και Average Precision (AP) χρησιμοποιώντας το πρόγραμμα/εργαλείο trec_eval ή το online εργαλείο "The TREC Files". Σχολιάστε τα αποτελέσματα και καταλήξτε σε συμπεράσματα.

Συνολικά, παράγουμε 9 μετρήσεις, αφού έχουμε ενώσει όλα τα qrels και πρέπει να ελέγξουμε για τα qrels τους τρεις διαφορετικούς όρους μετρικών σε σχέση με τα τρία είδη εγγράφων trec που έχουν δημιουργηθεί απο τα scripts.

Εδώ φαίνονται οι μετρήσεις που πραγματοποιήσαμε και παρακάτω αναλύονται οι μεθοδολογίες που ακολουθήσαμε καθώς και τα αποτελέσματα τους:

Αρχείο/Εντολή	(AP)map	map at P.10	rprec
301-450			
results_desc_narr [titles+desc+narr]	O.2143	O.4146	O.2590
results_desc [title+desc]	O.2211	O.4134	O.2683
results [titles]	O.2217	O.4273	O.2691

Παρακάτω θα αναλύσουμε τις μετρήσεις και θα εμβαθύνουμε στα πιθανά αίτια τους:

- για να υπολογίσουμε τις μετρικές χρησιμοποιούμε τη συγκεκριμένη εντολή : [./trec eval /home/User/IR-2019-2020-Project-1/qrels.301-450.trec.adhoc /home/User/Downloads/IR-2019-2020-Project-1/results.trec](#)

Average Precision

- Ουσιαστικά πρόκειται για την εντολή που δείχνει τον μέσο όρο η τιμή ακριβείας που λαμβάνεται μετά από ανάκτηση κάθε σχετικού εγγράφου
- Όταν δεν εμφανίζεται ένα σχετικό document η τιμή της μετρικής είναι 0.
- **Συγκριση ως προς τα αρχαia results**: Η τιμή της μέτρησής είναι μεγαλύτερη για τα αποτελέσματα που έχουν μόνο τους τίτλους και όσο τα results εμπλουτίζονται με παραπάνω στοιχεία (titles+desc ή titles+desc+narr) η τιμή της μέτρησης μειώνεται.

Precision@10

- Με τη χρήση αυτής της μετρικής εξετάζω τη συνάφεια για ένα μονό μικρό αριθμό των ανακτημένων αποτελεσμάτων, εδώ για 10 documents.
- Ως μετρική θεωρείται πως δεν είναι σταθερή και ούτε αξιόπιστη, γιατί ο συνολικός αριθμός των relevant documents επηρεάζει αισθητά το precision.
- **Συγκριση ως προς τα αρχαia results**: Η τιμή στις μετρήσεις δεν αυξάνεται ή μειώνεται ανάλογα με τον εμπλουτισμό των results. Παρατηρείται μια αστάθεια στη μεταβολή των αποτελεσμάτων των μετρήσεων και αυτό οφείλεται στο γεγονός πως η αναζήτηση περιορίζεται σε 10 documents.

Rprec

- Προκειται για μια μορφοποιημένη μορφή του R-Precision που το R-Precision είναι το ίδιο με Precision@X όπου το X είναι ο συνολικός αριθμός των relevant documents της συλλογής. Το ιδανικό σύστημα θα είχε τιμή 1 για την αναζήτηση κάθε query.
- Δηλαδή αν έχουμε |Rel| relevant documents για ένα query εξετάζουμε τα top |Rel| αποτελέσματα του συστήματος και βρίσκω πως r είναι relevant. Έτσι, ορίζω ως $R\text{-precision} = r / |Rel|$.
- **Συγκριση ως προς τα αρχαia results**: Η τιμή της μέτρησής είναι μεγαλύτερη για τα αποτελέσματα που έχουν μόνο τους τίτλους και όσο τα results εμπλουτίζονται με παραπάνω στοιχεία (titles+desc ή titles+desc+narr) η τιμή της μέτρησης μειώνεται.

B.

Κατεύθυνση Β.2 – Επέκταση Ερωτήματος με Φράσεις Βρείτε έναν αριθμό από καλές στατιστικές φράσεις (πχ bi-words) επεξεργάζοντας τη συλλογή, και εμπλουτίστε τα ερωτήματα. Τρέξτε τις εμπλουτισμένες ερωτήσεις και αξιολογήστε τις μεθόδους σας με τις μετρικές του Μέρους Α. Παρουσιάστε τα αποτελέσματα σε μορφή πίνακα, συγκρίνετε και σχολιάστε τα καταλήγοντας σε συμπεράσματα.

B1. Επεξεργασία Συλλογής

Πριν καταλήξουμε στην επιλογή των bi-words θα πρέπει να επεξεργαστούμε τα αρχεία της συλλογής έτσι ώστε να εμφανίζονται οι λέξεις όπως και σε ένα λεξικό: χωρίς σημεία στίξης, με μικρά γράμματα, κρατάμε και αναγνωρίζουμε τη βασική λέξη χωρίς να επηρεάζεται το αποτέλεσμα από το χρόνο ή την κλίση της. Φυσικά, θα πρέπει να αφαιρέσουμε τις πιο συχνά χρησιμοποιούμενες λέξεις όπως τα άρθρα.

B1.1 Αρχικά, πριν καταλήξουμε στην επιλογή των bi-words θα πρέπει να επεξεργαστούμε τα αρχεία της συλλογής έτσι ώστε να εμφανίζονται οι λέξεις όπως και σε ένα λεξικό: χωρίς σημεία στίξης, με μικρά γράμματα, κρατάμε και αναγνωρίζουμε τη βασική λέξη χωρίς να επηρεάζεται το αποτέλεσμα από το χρόνο ή την κλίση της. Φυσικά, θα πρέπει να αφαιρέσουμε τις πιο συχνά χρησιμοποιούμενες λέξεις της αγγλικής γλώσσας, αυτό γίνεται αυτοματοποιημένα από μια εντολή που περιλαμβάνεται στις βιβλιοθήκες που αξιοποιούμε.

```
removing_regular_expressions (3).py > No Selection
1 import io
2 import os
3 import re
4
5 import nltk
6 from nltk.corpus import stopwords
7 from nltk.stem import porter
8 from nltk.tokenize import word_tokenize
9 from nltk.tokenize import sent_tokenize, word_tokenize
10 from nltk.stem import PorterStemmer
11 # nltk.download('stopwords')
12 myPorterStemmer = nltk.stem.porter.PorterStemmer()
13 stop_words = set(stopwords.words('english'))
14 train_folders = ['fbis','latimes']
15 new_folders = 'final'
16 appendFile = open(new_folders, 'wt')
17 i=0
18 j=0
19 for folder in train_folders:
20     files = os.listdir(folder)
21     print(j)
22     j=j+1
23     for file in files:
24         path1 = os.path.join(folder, file)
25         file1 = open(path1)
26         line = file1.read() # Use this to read file content as a stream:
27         result = re.sub("\<[^\>]*\>", "", line, 0, re.IGNORECASE | re.MULTILINE)
28         result2 = re.sub("[^\w\s]", "", result, 0, re.IGNORECASE | re.MULTILINE)
29         result3=result2.lower()
30         words = result3.split()
31         print(i)
32         i=i+1
33         for r in words:
34             if not r in stop_words:
35
36                 appendFile.write(" " + r)
37
```

```

train_folders2 = [ 'fr94', 'ft', ]

j=0
for folder in train_folders2:
    subfolders = os.listdir(folder)
    print(j)
    j = j + 1
    for subfolder in subfolders:

        path = os.path.join(folder, subfolder)
        files=os.listdir(path)
        for file in files:
            path1 = os.path.join(folder,subfolder, file)
            print(i)
            i=i+1
            file1 = open(path1, encoding="ISO-8859-1")
            line = file1.read() # Use this to read file content as a stream:
            result = re.sub("<[^\>]*\>", "", line, 0, re.IGNORECASE |
                re.MULTILINE)
            result2 = re.sub("[^\w\s]", "", result, 0, re.IGNORECASE |
                re.MULTILINE)
            result3 = result2.lower()
            words = result3.split()
            for r in words:
                if not r in stop_words:

                    appendFile.write(" " + r)

appendFile.close()

```

B1. Δημιουργήσαμε το συγκεκριμένο script όπου τελικά αξιοποιήσαμε μόνο αυτά που αποθηκεύσαμε στα f1 και f2. Τα f1 και f2 λειτουργούν ως θέσεις αποθήκευσης των λιστών που καταλήγουν να αποθηκευτούν στα αρχεία final και biwords. Τα f1 και f2 έχουν όλα τα bi-words και τα 30000 πιο συνηθισμένα αντίστοιχα.


```

< > ertima_b_bi-words (1).py > No Selection
1 import re
2 from itertools import islice
3 import pickle
4 import nltk
5 import regex
6 from nltk import BigramCollocationFinder
7 from nltk.util import ngrams
8 from nltk.internals import Counter
9 f1 = open('store.allbi-words', 'wb')
10 f2= open('store.300000', 'wb')
11
12 i=0
13 print(i)
14 f = open('final')
15 appendFile = open('bi-words_', "a")
16 raw = f.read()
17 i=i+1
18 print(i)
19 tokens = nltk.word_tokenize(raw)
20 i=i+1
21 print(i)
22 bgs = nltk.bigrams(tokens)
23 i=i+1
24 print(i)
25
26 fdist = nltk.FreqDist(bgs)
27
28 appendFile2 = open('bi-words_all2', "a")
29 l = fdist.most_common()
30 pickle.dump(l, f1)
31 f1.close()
32
33 print(l, file=appendFile2)
34 k = fdist.most_common(30000)
35 pickle.dump(k, f2)
36 f2.close()
37 print(k, file=appendFile)
38 i=i+1
39 print(i)
40 appendFile.close()
41 f.close()

```

B1.3 Τελικά, αξιοποιώντας τη “θέση αποθήκευσης” που αντιστοιχούσε στο προηγούμενο script στο f1, παίρνουμε ένα προς ένα τα στοιχεία(κ[]) της λίστας που έχει δημιουργηθεί και κάνω τους απαραίτητους ελέγχους για να ενταχθεί τελικά το στοιχείο στο νέο indrironquery της fout , ώστε να οδηγηθούμε στις νέες μετρήσεις

```

> final_step.py > No Selection
import pickle
import re
f = open('store.allbi-words', 'rb')
obj = pickle.load(f)
f.close()

j=0
m=0
filterlist=["<text>","</text>"]
fout=open('IndriRunQuery.queries.file.301-450-titles-only.EXAMPLE_final_higher_than_4000','a')
with open('IndriRunQuery.queries.file.301-450-titles-only.EXAMPLE', "r") as f:

    for line in f:
        m=m+1
        print(m,j)
        i = False

        result = re.sub('<text>', '<text> ', line)
        result2 = re.sub('</text>', ' </text>', result)
        wordlist1 = result2.split()
        word_number=0
        for word in wordlist1:
            words = []
            check_if_bi_word_already_replaced=False
            k=00000
            if i==True:
                while obj[k][1]>4000:
                    if word.lower()==obj[k][0][0] or word.lower()==obj[k][0][1]:
                        j=j+1
                        print(j)
                        if wordlist1[word_number-1].lower()==obj[k][0][0] or wordlist1[word_number-1].lower()==obj[k][0][1]:
                            check_if_bi_word_already_replaced=True
                            j=j-1
                            print(j)
                            break
                        words.append(obj[k][0][0])
                        words.append(obj[k][0][1])
                        s = " "
                        s = s.join(words)
                        fout.write(s)
                        fout.write(" ")
                        break
                    k = k + 1

            k = k + 1
            if(words==[] and check_if_bi_word_already_replaced==False):
                words.append(word)
                s = " "
                s = s.join(words)
                fout.write(s)
                fout.write(" ")

            k=k+1
        else:
            words.append(word)
            s = " "
            s = s.join(words)
            fout.write(s)

        if word.lower() in filterlist:
            i=not i
            word_number = word_number + 1

        fout.write("\n")

```

```

            k = k + 1
            if(words==[] and check_if_bi_word_already_replaced==False):
                words.append(word)
                s = " "
                s = s.join(words)
                fout.write(s)
                fout.write(" ")

            k=k+1
        else:
            words.append(word)
            s = " "
            s = s.join(words)
            fout.write(s)

        if word.lower() in filterlist:
            i=not i
            word_number = word_number + 1

        fout.write("\n")

```

B2. Μετρήσεις

B2.1 Titles

Εδώ φαίνονται οι μετρήσεις που αφορούν τα biwords για το αρχείο που εμπεριέχει μόνο τους titles

συχνότητα εμφάνισης biwords/Εντολή	(AP)map	map at P.10	rprec
301-450			
2 biwords	0.1359	0.2707	0.1774
50 biwords	0.1409	0.2767	0.1839
200 biwords	0.1591	0.3000	0.2058
500 biwords	0.1676	0.3207	0.2106
1000 biwords	0.1762	0.3367	0.2222
2000 biwords	0.1920	0.3620	0.2372
3000 biwords	0.1983	0.3720	0.2417
4000 biwords	0.2047	0.3847	0.2499
8000 biwords	0.2147	0.4013	0.2609
12000 biwords	0.2146	0.4053	0.2610

Παρατηρούμε πως όσο αυξάνεται η συχνότητα εμφάνισης των biwords, αυξάνεται και η απόδοση για τις μετρικές, πιθανών ο εμπλουτισμός με παραπάνω στοιχεία οδηγεί σε αυτή την αλλαγή.

-Μέτρηση του A:

results [titles]	0.2217	0.4273	0.2691
------------------	--------	--------	--------

Συγκρίνοντας με τα αποτελέσματα των μετρήσεων του A, βλέπουμε πως τα αποτελέσματα για συχνότητα εμφάνισης 8000 biwords και παραπάνω, πλησιάζουν αρκετά αυτά του A, αλλά είναι χειρότερα.

Επιπλέον, βλέπουμε πως οι μετρήσεις για average precision παούν να αυξάνονται ραγδαία μετά τη συχνότητα 2000 biwords. Αντίστοιχα, για average precision των 10 documents η ραγδαία αύξηση σταματάει στη συχνότητα 8000 biwords, ενώ για το rprec η μεταβολή τιμών εκμηδενίζεται σχεδόν στη συχνότητα 8000 biwords.

B2.2 Titles+description

Εδώ φαίνονται οι μετρήσεις που αφορούν τα biwords για το αρχείο που εμπεριέχει μόνο τους titles+ description

συχνότητα εμφάνισης biwords/Εντολή	(AP)map	map at P.10	rprec
301-450			
20000 biwords	O.2087	O.4073	O.2559
30000 biwords	O.2094	O.4089	O.2570

-Μέτρηση του A:

results_desc [title+desc]	O.2211	O.4134	O.2683
--	---------------	---------------	---------------

Συγκρίνοντας με τα αποτελέσματα των μετρήσεων του A, βλέπουμε πως όσο η συχνότητα αυξάνεται τα αποτελέσματα για μεγαλύτερες συχνότητες εμφάνισης, πλησιάζουν αρκετά αυτά του A, αλλά είναι χειρότερα.