ΣΥΣΤΗΜΑΤΑ ΑΝΑΚΤΗΣΗΣ ΠΛΗΡΟΦΟΡΙΩΝ

ΑΚΑΔΗΜΑΙΚΟ ΕΤΟΣ: 2019

ΙΑΚΩΒΟΣ ΕΥΔΑΙΜΩΝ 3130059

ΦΑΣΗ 2^{Η}

ΟΜΑΔΑ Α

ΔΙΕΥΚΡΙΝΗΣΕΙΣ

Προτού γίνει η ανάλυση της φάσης 2 θα ήθελα να παραθέσω κάποιες διευκρινήσεις για την φάση 1 και για την 2 οι οποίες ήταν εν αγνοία μου και θα ήθελα να τις αναφέρω τώρα καθώς είναι σημαντικό και για την εκτέλεση της φάσης 2. Καθώς, κατά την φάση 1 ήταν μια μεταβατική περίοδος καθώς νέες εκδόσεις της Elastic Search δημοσιεύτηκαν παρόλο που είχα αναφέρει στο report της φάσης 1 ό,τι χρειάζεται να εγκατασταθεί ώστε να τρέξει η εργασία της φάσης 1 είχα παραλείψει ένα ακόμα στοιχείο καθώς ήταν εν αγνοία μου αυτό το γεγονός. Εκτός από τις νέες εκδόσεις της ElasticSearch εκείνη την περίοδο δημοσιεύτηκαν και ήταν διαθέσιμες προς κατέβασμα νέες εκδόσεις της βιβλιοθήκης elasticsearch της python. Με αποτέλεσμα καθώς πληκτρολογούμε την εντολή **pip install elasticsearch** να εγκαθίσταται στον υπολογιστή μας η τελευταία διαθέσιμη έκδοση της βιβλιοθήκης elasticsearch. Οπότε, λογικά όταν εσείς προβήκατε στην διόρθωση της εργασίας και κάνατε εγκατάσταση την βιβλιοθήκη elasticsearch της python κατεβάσατε διαφορετική έκδοση από αυτή που είχα και δούλευα εγώ. Γεγονός που θα δημιούργησε πρόβλημα στην εκτέλεση του προγράμματός μου και θα πέταγε errors καθώς η τελευταίας έκδοσης της βιβλιοθήκης elasticsearch-py της python δεν υποστηρίζει το doc_type σαν παράμετρο της μεθόδου search(), σε αντίθεση με την έκδοση της βιβλιοθήκης που χρησιμοποίησα εγώ και κατά την φάση 1 και την φάση 2. Για αυτό το λόγο κιόλας παραθέτω αυτή την διευκρίνιση καθώς μπορεί και άλλοι συνάδελφοι μου να μην ήξεραν αυτή τη λεπτομέρεια και να είχαν παρόμοιο θέμα. Πάντως, οι λύσεις σε αυτό το θέμα είναι δύο είτε όπου καλείται η μέθοδος search της βιβλιοθήκης της elasticsearch, να διαγράφεται η παράμετρος doc_type είτε

να εγκατασταθεί η εκάστοτε έκδοση της βιβλιοθήκης της elasticsearch που χρησιμοποιήθηκε από τον κάθε φοιτητή. Η εντολή ώστε να εγκατασταθεί η δικιά μου έκδοση της βιβλιοθήκης της elasticsearch είναι η : **pip install elasticsearch==6.3.1** Παρακάτω αναφέρομαι με μεγαλύτερες λεπτομέρειες για το πως γίνεται η εγκατάσταση. Τέλος, καθώς θα προβείτε στο να εγκαταστήσετε μία διαφορετική έκδοση της βιβλιοθήκης της elasticsearch πρέπει να γνωρίζετε ότι διαγράφονται οι ήδη εγκατεστημένες εκδόσεις της βιβλιοθήκης είτε είναι πιο πρόσφατες είτε είναι πιο παλιές από την έκδοση που πρόκειται να εγκαταστήσετε.

ΔΙΑΔΙΚΑΣΤΙΚΑ ΓΙΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ ΤΗΣ ΦΑΣΗΣ 2

Στο παραδοτέο οι φάκελοι Collection_1 και Queries είναι άδεια ώστε να μπορεί να ανεβεί το παραδοτέο στο eclass. Για να τρέξει το script θα πρέπει να εισαχθούν από τον διορθωτή τα κατάλληλα αρχεία στον κατάλληλο φάκελο. Παρακάτω, αναλύεται σε ποιον φάκελο πρέπει να μπουν τα αρχεία.

Για τους σκοπούς αυτής της φάσης της εργασίας χρησιμοποιήθηκε η γλώσσα προγραμματισμού Python και το IDE της PyCharm. Επίσης χρησιμοποιήθηκε η ElasticSearch-6.6.2. Για την διαχείριση του ElasticSearch χρησιμοποιήθηκε Python Client. Για την εκτέλεση του script της Python είναι απαραίτητη η εγκατάσταση 1 package, το package της elasticsearch. Για το package της elasticsearch πρέπει στο command prompt να πληκτρολογηθεί η εντολή **pip install elasticsearch==6.3.1**. Εγκαταστήστε πρώτα το πακέτο και μετά ανοίξτε το με οποιοδήποτε IDE το project. Επίσης, σε ορισμένους IDE χρειάζεται να οριστεί ένας Interpreter, στην περίπτωση αυτή προτείνω την χρήση του Interpreter που είναι εγκαταστημένος στον υπολογιστή σας ώστε να έχει και τα νέα packages που εγκαταστάθηκαν. Το επισημάνω αυτό καθώς υπάρχει η πιθανότητα να σας έχει σαν επιλογή την χρήση του Interpreter που χρησιμοποιήθηκε για το project και που ήταν εγκαταστημένος στον δικό μου υπολογιστή.

Και τα δύο ερωτήματα της φάσης 2 βρίσκονται σε ένα script. Για τους σκοπούς της εργασίας χρησιμοποιείται ο Index της 1ης φάσης ο οποίος λέγεται **collection και έχει doc_type = project.** Κατά το δεύτερο ερώτημα της εργασίας δημιουργούμε έναν νέο Index ώστε να έχει στο mapping του

το term_vector = “yes” ώστε να μπορέσουμε να εκτελέσουμε τα ερωτήματα more_like_this. Ο νέος Index του ερωτήματος β λέγεται **mlt_collection και έχει doc_type = project.** Για την δημιουργία του Index από την 1η φάση δηλαδή του index collection δεν παρέχεται ο κώδικας με την δημιουργία του και η φόρτωση της συλλογής σε αυτόν καθώς αυτός ο κώδικας παραδόθηκε με το παραδοτέο της 1ης φάσης, οπότε η εργασία της 2ης φάσης προϋποθέτει να υπάρχει ήδη ο Index collection από την 1η φάση.

ΥΛΟΠΟΙΗΣΗ ΦΑΣΗΣ 2 ΚΑΙ ΕΠΕΞΗΓΗΣΗ ΚΩΔΙΚΑ

Αρχικά μέσα στο παραδοτέο project υπάρχουν τρεις φάκελοι. Ο ένας φάκελος λέγεται Collection_1, ο άλλος Queries και ο τρίτος λέγεται Results of queries. Στο φάκελο Collection_1 αντιγράφουμε τα text αρχεία που αποτελούν την Collection_1 όπου μας έχει δοθεί και περιέχει τους όρους και τις φράσεις του κάθε κειμένου. Στον φάκελο Queries αντιγράφουμε τα ερωτήματα (δηλαδή το αρχείο testingQueries.txt) από την πρώτη φάση τα οποία θα χρησιμοποιήσουμε κατά το β ερώτημα της φάσης 2. Τέλος, στον φάκελο Results of queries θα δημιουργούνται τα txt αρχεία με τα αποτελέσματα των ερωτημάτων που θα τρέξουμε και κατά το ερώτημα α και β της φάσης 2. Σε αυτό τον φάκελο θα δημιουργούνται επτά txt αρχεία όπου τα τρία θα αποτελούν τα ερωτήματα του Α’ ερωτήματος και το καθένα από αυτά τα τρία θα αφορά το 30%, το 60% και το 90% των όρων φράσεων αντίστοιχα από κάθε αρχείο txt που αντιστοιχεί στα αρχεία από τα ερωτήματα της φάσης 1. Τα άλλα τέσσερα αρχεία που βρίσκουμε στον φάκελο Results of queries αφορούν τα τέσσερα ερωτήματα του more like this, όπου το αρχείο myResults_mlt_1.txt έχει τα αποτελέσματα από το more like this query με τις default παραμέτρους και τα αρχεία myResults_mlt_2.txt, myResults_mlt_3.txt, myResults_mlt_4.txt έχει τα αποτελέσματα με τα more like this queries με τις κατάλληλες παραμέτρους ώστε να σχηματίζονται πιο χαλαρά ή πιο ισχυρά ερωτήματα κάθε φορά. Πριν εκτελέσουμε τον κώδικα του project πρέπει πρώτα να ξεκινήσουμε το elasticsearch εκτελώντας το elasticsearch.bat που βρίσκεται μέσα στο φάκελο bin του elasticSearch. Στο script Phase2.py γίνονται όλες οι διαδικασίες από την επεξεργασία των φράσεων-όρων του Collection_1, την επεξεργασία των ερωτήσεων από την φάση 1 που ξαναχρησιμοποιούμε στο β ερώτημα της φάσης 2 μέχρι την δημιουργία του νέου index mlt_collection που χρησιμοποιούμε για τα ερωτήματα

more like this και την εκτέλεση των ερωτημάτων more like this και αυτών των ερωτημάτων όπου χρησιμοποιούμε το 30%, 60% και 90% των φράσεων από το Collection_1. Αρχικά, το πρόγραμμά μας ξεκινάει με το διάβασμα και των 'καθαρισμό' από special characters των φράσεων-όρων από το Collection_1 όπου αποθηκεύει όλες τις φράσεις για κάθε ένα αρχείο σε μία λίστα και κάθε τέτοια λίστα σε μία άλλη λίστα. Η τελευταία αυτή λίστα με όλες τις άλλες λίστες χρησιμοποιείται από την μέθοδο choose_files() και ώστε να επιλεχθούν οι φράσεις-όροι από τα κατάλληλα αρχεία μόνο, δηλαδή τα αρχεία που χρησιμοποιήθηκαν σαν ερωτήματα κατά την φάση 1. Έτσι, παίρνουμε δέκα τελικές λίστες, δηλαδή μία λίστα από καθένα από τα δέκα ερωτήματα, όπου αυτές οι λίστες δίνονται σαν παράμετρος σε μία συνάρτηση με το όνομα take_percentage(), η οποία παίρνει και μια δεύτερη παράμετρο σαν όρισμα όπου αφορά το ποσοστό των όρων που θα επιλεχθούν από κάθε λίστα των δέκα ερωτημάτων. Αφού, εκτελεστεί αυτή η μέθοδος τότε μας επιστρέφει το αντίστοιχο ποσοστό των όρων-φράσεων που επιλέχθηκαν από το κάθε ερώτημα. Έπειτα, θα συνδεθούμε με το Elastic Search και θα δώσουμε την αντίστοιχη λίστα με τα δέκα ερωτήματα στην αντίστοιχη μέθοδο ανάλογα με το ποσοστό των όρων που έχουμε επιλέξει. Δηλαδή για τα αποτελέσματα του 30% των όρων από τα ερωτήματα θα δοθεί ως παράμετρος στην συνάρτηση search_30(), για το 60% των όρων από τα ερωτήματα θα δοθεί ως παράμετρος στην συνάρτηση search_60() και για το 90% των όρων από τα ερωτήματα θα δοθεί ως παράμετρος στην συνάρτηση search_90(). Οι συγκεκριμένες τρεις συναρτήσεις καλούν την συνάρτηση search() όπου αυτή θα γράψει ουσιαστικά το αρχείο και θα κάνει τα ερωτήματα. Απλά οι συναρτήσεις search_30(), search_60() και search_90() βοηθούν μόνο ώστε να δημιουργείται το κατάλληλο αρχείο με τα αποτελέσματα των ερωτημάτων με διακριτό όνομα από τα άλλα. Πιο συγκεκριμένα η συνάρτηση search_30 δίνει στο αρχείο το όνομα myResults_30.txt, η συνάρτηση search_60 δίνει στο αρχείο το όνομα myResults_60.txt και η συνάρτηση search_90 δίνει στο αρχείο το όνομα myResults_90.txt και το κάθε όνομα αρχείου δίνεται σαν όρισμα μετά στην συνάρτηση search ώστε να το δημιουργήσει να εκτελέσει τα ερωτήματα και να γράψει τα αποτελέσματά τους στο κατάλληλο αρχείο. Η συνάρτηση search() δημιουργεί το αντίστοιχο αρχείο txt με τα αποτελέσματα των ερωτημάτων που αργότερα θα χρησιμοποιηθεί από το trec_eval για την αξιολόγηση των ερωτημάτων. Η συνάρτηση αυτή

χρησιμοποιεί το match query και το bool query για την εκτέλεση των ερωτημάτων. Δηλαδή, κάθε ένας όρος ενός ερωτήματος χρησιμοποιείται από ένα match query και έτσι κάθε ένα από αυτά τα match queries για κάθε όρο ενός ερωτήματος περικλείονται από το bool query χρησιμοποιώντας την should clause που έχει τον ρόλο του OR για την σύνδεση όλων των match queries.

```
# match and query_string have the same results
query = {
    'from': 1, 'size': 20,
    'query': {
        'bool': {
            'should': [{'match': {'project.text': {'query': q[phrs]}}} for phrs in range(len(q))]
        }
    }
}
```

Αφού, δημιουργηθούν τα τρία αυτά αρχεία με τα αποτελέσματα των ερωτημάτων συνεχίζουμε με την αξιολόγηση των αποτελεσμάτων μας από το trec_eval. Στον φάκελο που βρίσκεται το εργαλείο trec_eval αντιγράφουμε το αρχείο myResults_30.txt, myResults_60.txt και myResults_90.txt που δημιουργήσαμε. Επίσης, στον φάκελο που βρίσκεται το εργαλείο trec_eval αντιγράφουμε και το αρχείο qrels.txt που μας δόθηκε και το οποίο περιέχει τις σωστές απαντήσεις σε κάθε κείμενο. Έπειτα, τρέχοντας το command prompt με current directory το path του φακέλου που βρίσκεται το trec_eval, εκτελούμε αρχικά για το αρχείο myResults_30.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_30.txt

```
trec_eval.exe -q qrels.txt myResults_30.txt
```

Για το αρχείο myResults_30.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     91
map                     all     0.4622
gm_map                  all     0.4186
Rprec                   all     0.5284
bpref                   all     0.6456
recip_rank              all     0.8833
iprec_at_recall_0.00    all     0.9644
iprec_at_recall_0.10    all     0.8844
iprec_at_recall_0.20    all     0.8278
iprec_at_recall_0.30    all     0.6945
iprec_at_recall_0.40    all     0.5669
iprec_at_recall_0.50    all     0.5187
iprec_at_recall_0.60    all     0.3938
iprec_at_recall_0.70    all     0.3187
iprec_at_recall_0.80    all     0.2561
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.7000
P_10                    all     0.5600
P_15                    all     0.5067
P_20                    all     0.4550
P_30                    all     0.3033
P_100                   all     0.0910
P_200                   all     0.0455
P_500                   all     0.0182
P_1000                  all     0.0091
```

Έπειτα, εκτελούμε για το αρχείο myResults_60.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_60.txt

```
trec_eval.exe -q qrels.txt myResults_60.txt
```

Για το αρχείο myResults_60.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

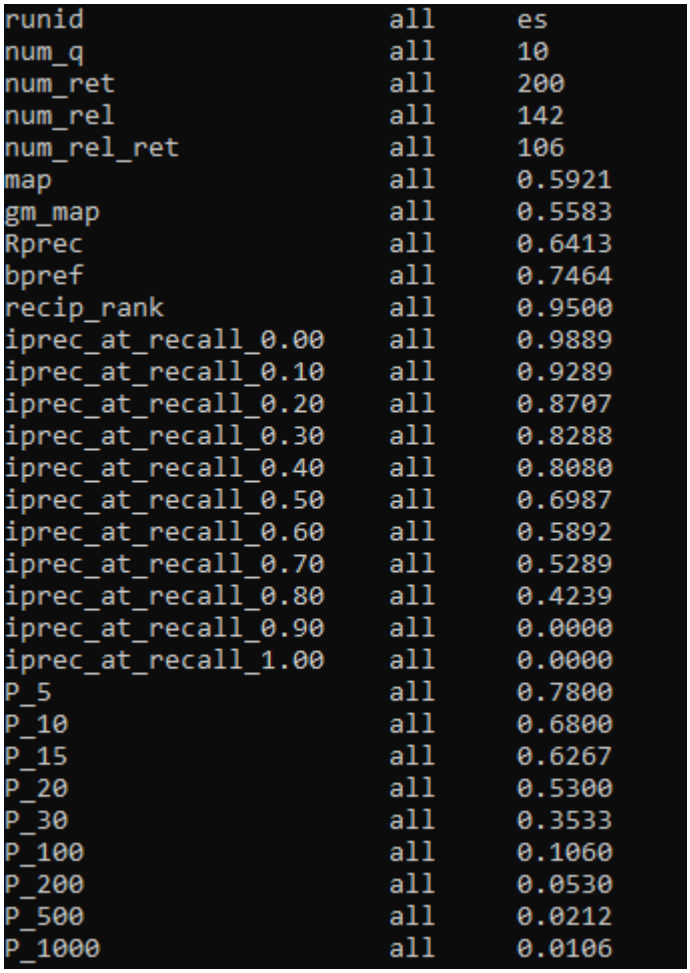
```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     100
map                     all     0.5207
gm_map                  all     0.4857
Rprec                   all     0.5922
bpref                   all     0.6999
recip_rank              all     0.9000
iprec_at_recall_0.00    all     0.9625
iprec_at_recall_0.10    all     0.9425
iprec_at_recall_0.20    all     0.7853
iprec_at_recall_0.30    all     0.7381
iprec_at_recall_0.40    all     0.7043
iprec_at_recall_0.50    all     0.5868
iprec_at_recall_0.60    all     0.5029
iprec_at_recall_0.70    all     0.4239
iprec_at_recall_0.80    all     0.2758
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.7200
P_10                    all     0.6300
P_15                    all     0.5867
P_20                    all     0.5000
P_30                    all     0.3333
P_100                   all     0.1000
P_200                   all     0.0500
P_500                   all     0.0200
P_1000                  all     0.0100
```

Τέλος, εκτελούμε για το αρχείο myResults_90.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_90.txt

```
>trec_eval.exe -q qrels.txt myResults_90.txt
```

Για το αρχείο myResults_90.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:



runid	all	es
num_q	all	10
num_ret	all	200
num_rel	all	142
num_rel_ret	all	106
map	all	0.5921
gm_map	all	0.5583
Rprec	all	0.6413
bpref	all	0.7464
recip_rank	all	0.9500
iprec_at_recall_0.00	all	0.9889
iprec_at_recall_0.10	all	0.9289
iprec_at_recall_0.20	all	0.8707
iprec_at_recall_0.30	all	0.8288
iprec_at_recall_0.40	all	0.8080
iprec_at_recall_0.50	all	0.6987
iprec_at_recall_0.60	all	0.5892
iprec_at_recall_0.70	all	0.5289
iprec_at_recall_0.80	all	0.4239
iprec_at_recall_0.90	all	0.0000
iprec_at_recall_1.00	all	0.0000
P_5	all	0.7800
P_10	all	0.6800
P_15	all	0.6267
P_20	all	0.5300
P_30	all	0.3533
P_100	all	0.1060
P_200	all	0.0530
P_500	all	0.0212
P_1000	all	0.0106

Έτσι, ολοκληρώνουμε το πρώτο ερώτημα της φάσης 2. Κατά το δεύτερο ερώτημα της φάσης 2 δημιουργούμε έναν νέο index με το όνομα mlt_collection με τα ίδια settings με τον index collection αλλά με μία επιπλέον προσθήκη στο mapping ότι κατά τον ορισμό του πεδίου text εισάγουμε και την παράμετρο term_vector: yes που μας είναι απαραίτητο για το more like this query.

```
settings = {
    "settings": {
        "index": {
            "analysis": {
                "analyzer": {
                    "default": {
                        "type": "english"
                    }
                }
            }
        }
    },
    "mappings": {
        "project": {
            "properties": {
                "project": {
                    "properties": {
                        "xmlns": {
                            "type": "text"
                        },
                        "rcn": {
                            "type": "text"
                        },
                        "acronym": {
                            "type": "text"
                        },
                        "text": {
                            "type": "text",
                            "term_vector": "yes"
                        },
                        "identifier": {
                            "type": "text"
                        }
                    }
                }
```

Αφού, δημιουργήσουμε τον νέο index χρησιμοποιούμε την ακόλουθη εντολή:

```
helpers.reindex(client=es, source_index=index_name, target_index=new_index_name)
```

Έτσι ώστε να αντιγράψουμε τη συλλογή από τον index collection στον νέο index mlt_collection. Τέλος, εκτελείται η συνάρτηση search_mlt() η οποία καλεί με την σειρά της την συνάρτηση query_processing() ώστε να διαβαστούν από το αρχείο testingQueries.txt, που βρίσκεται στον φάκελο Queries, τα ερωτήματα που είχαν χρησιμοποιηθεί και κατά την πρώτη φάση τα οποία καθώς τα διαβάζουμε τα επεξεργαζόμαστε κιόλας αφαιρώντας από αυτά τους special characters. Αφού, τα διαβάσουμε και τα προ επεξεργαστούμε τα καθένα από αυτά τα ερωτήματα χρησιμοποιούνται από τα τέσσερα more like this queries που έχουμε. Στο πρώτο more like this query που έχουμε με τις default παραμέτρους τα

αποτελέσματα από τα ερωτήματα γράφονται στο αρχείο myResults_mlt_1.txt.

```
queries = query_processing()
# -------------------DEFAULT EXAMPLE--------------------------------
path = 'Results of queries'
f_name = 'myResults_mlt_1.txt'
fullname = os.path.join(path, f_name)
fh = open(fullname, 'w+')
count = 1

for q in queries:

    query = {
        'from': 1, 'size': 20,
        'query': {
            "more_like_this": {
                "fields": ['project.text'],
                "like": q
            }
        }
    }

    res = els.search(index=index_n, doc_type='project', body=query)
```

Έπειτα, στο δεύτερο more like this query που έχουμε κατά το οποίο οι παράμετροι έχουν επιλεχθεί έτσι ώστε να αποτελεί ένα πιο “χαλαρό” query τα αποτελέσματά του γράφονται στο αρχείο myResults_mlt_2.txt

```
path = 'Results of queries'
f_name = 'myResults_mlt_2.txt'
fullname = os.path.join(path, f_name)
fh = open(fullname, 'w+')
count = 1
for q in queries:
    query = {
        'from': 1, 'size': 20,
        'query': {
            "more_like_this": {
                "fields": ['project.text'],
                "like": q,
                "max_query_terms": 100,
                "min_term_freq": 6,
                "min_doc_freq": 6,
                "max_doc_freq": 12121,
                "minimum_should_match": "30%"
            }
        }
    }
    res = els.search(index=index_n, doc_type='project', body=query)
```

Όπου, έχει επιλεχθεί ένας μεγάλος αριθμός από το μέγιστο όριο των όρων του query δίνοντάς μας θεωρητικά μεγαλύτερη ακρίβεια αλλά μικρότερο map καθώς έχουμε αρκετούς όρους που επιλέγονται οι οποίοι έχουν μικρή "σημασία". Επίσης έχει επιλεχθεί και ένα μεγάλο σχετικά κάτω όριο με την ελάχιστη συχνότητα των όρων σε κάθε κείμενο, ένα μεγάλο σχετικά κάτω όριο με την ελάχιστη συχνότητα της συχνότητας των κειμένων που εμφανίζεται κάθε όρος ώστε να περιορίσουμε τους όρους που εμφανίζονται σε λιγότερα από έξι κείμενα της συλλογής και επιλέχθηκε περίπου το 80% από όλη την συλλογή, αριθμός ο οποίος είναι το μέγιστο όριο των κειμένων όπου θα πρέπει να εμφανίζεται ένας όρος πάνω από τον οποίο δεν θα λαμβάνεται υπόψιν αυτός ο όρος(max_doc_freq), έτσι ώστε να περιορίσουμε να λαμβάνονται υπόψιν τα stopwords.

Στο τρίτο more like this query που έχουμε κατά το οποίο οι παράμετροι έχουν επιλεχθεί έτσι ώστε να αποτελεί ένα πιο ισχυρό query από ότι το παραπάνω και του οποίου τα αποτελέσματά του γράφονται στο αρχείο myResults_mlt_3.txt

```
path = 'Results of queries'
f_name = 'myResults_mlt_3.txt'
fullname = os.path.join(path, f_name)
fh = open(fullname, 'w+')
count = 1
for q in queries:
    query = {
        'from': 1, 'size': 20,
        'query': {
            "more_like_this": {
                "fields": ['project.text'],
                "like": q,
                "max_query_terms": 15,
                "min_term_freq": 5,
                "min_doc_freq": 6,
                "max_doc_freq": 1200,
                "minimum_should_match": "30%"
            }
        }
    }
    res = els.search(index=index_n, doc_type='project', body=query)
```

Όπου, έχει επιλεχθεί ένας μικρός αριθμός από το μέγιστο όριο των όρων του query δίνοντάς μας τους σημαντικότερους όρους, καθώς επίσης έχει επιλεχθεί και ένα μεγάλο σχετικά κάτω όριο με την ελάχιστη συχνότητα των όρων σε κάθε κείμενο. Επιπλέον, έχει επιλεχθεί ένα μεγάλο σχετικά

κάτω όριο με το κατώτατο όριο της συχνότητας εμφάνισης ενός όρου σε όλα τα κείμενα της συλλογής, έτσι ώστε να περιορίσουμε τους όρους που εμφανίζονται σε λιγότερα από έξι κείμενα. Επιπροσθέτως, έχει επιλεχθεί περίπου το 10% από όλη την συλλογή ως max_doc_freq, έτσι ώστε, να περιορίσουμε κατά πολύ την πιθανότητα να λαμβάνονται υπόψιν τα stopwords.

Τέλος, στο τέταρτο και τελευταίο more like this query που έχουμε κατά το οποίο οι παράμετροι έχουν επιλεχθεί έτσι ώστε να αποτελεί ένα ισχυρό query και τα αποτελέσματά του να γράφονται στο αρχείο myResults_mlt_4.txt

```
path = 'Results of queries'
f_name = 'myResults_mlt_4.txt'
fullname = os.path.join(path, f_name)
fh = open(fullname, 'w+')
count = 1
for q in queries:
    query = {
        'from': 1, 'size': 20,
        'query': {
            "more_like_this": {
                "fields": ['project.text'],
                "like": q,
                "max_query_terms": 15,
                "min_term_freq": 2,
                "min_doc_freq": 1,
                "max_doc_freq": 18316,
                "minimum_should_match": "30%"
            }
        }
    }
    res = els.search(index=index_n, doc_type='project', body=query)
```

Όπου, έχει επιλεχθεί ένας μικρός αριθμός από το μέγιστο όριο των όρων του query δίνοντάς μας τους 15 πιο σημαντικούς όρους, καθώς επίσης έχει επιλεχθεί και ένα μικρό κάτω όριο με την ελάχιστη συχνότητα των όρων σε κάθε κείμενο, ένα μικρό κάτω όριο με το σε πόσα κείμενα από την συλλογή πρέπει να εμφανίζεται ένας συγκεκριμένος όρος ώστε να λαμβάνουμε υπόψιν. Επίσης, ορίζεται ότι το ανώτατο όριο και κάτω, όπου ένας όρος θα λαμβάνεται υπόψιν, είναι ακόμα και άμα ένας όρος εμφανίζεται σε όλα τα κείμενα της συλλογής, με αποτέλεσμα να μην περιορίζουμε τα stopwords.

Αφού, δημιουργηθούν τα τέσσερα αυτά αρχεία με τα αποτελέσματα των ερωτημάτων συνεχίζουμε με την αξιολόγηση των αποτελεσμάτων μας από το trec_eval. Στον φάκελο που βρίσκεται το εργαλείο trec_eval αντιγράφουμε το αρχείο myResults_mlt_1.txt, myResults_mlt_2. txt και myResults_mlt_3.txt και myResults_mlt_4.txt που δημιουργήσαμε. Επίσης, στον φάκελο που βρίσκεται το εργαλείο trec_eval αντιγράφουμε και το αρχείο qrels.txt που μας δόθηκε και το οποίο περιέχει τις σωστές απαντήσεις σε κάθε κείμενο. Έπειτα, τρέχοντας το command prompt με current directory το path του φακέλου που βρίσκεται το trec_eval, εκτελούμε αρχικά για το αρχείο myResults_mlt_1.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_mlt_1.txt

```
trec_eval.exe -q qrels.txt myResults_mlt_1.txt
```

Για το αρχείο myResults_mlt_1.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     67
map                     all     0.3299
gm_map                  all     0.2911
Rprec                   all     0.3745
bpref                   all     0.4657
recip_rank              all     1.0000
iprec_at_recall_0.00    all     1.0000
iprec_at_recall_0.10    all     0.8267
iprec_at_recall_0.20    all     0.7025
iprec_at_recall_0.30    all     0.5603
iprec_at_recall_0.40    all     0.3570
iprec_at_recall_0.50    all     0.2589
iprec_at_recall_0.60    all     0.1291
iprec_at_recall_0.70    all     0.0700
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.6400
P_10                    all     0.4800
P_15                    all     0.3667
P_20                    all     0.3350
P_30                    all     0.2233
P_100                   all     0.0670
P_200                   all     0.0335
P_500                   all     0.0134
P_1000                  all     0.0067
```

Έπειτα, εκτελούμε για το αρχείο myResults_mlt_2.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_mlt_2.txt

```
trec_eval.exe -q qrels.txt myResults_mlt_2.txt
```

Για το αρχείο myResults_mlt_2.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     62
map                     all     0.2843
gm_map                  all     0.1011
Rprec                   all     0.3656
bpref                   all     0.4208
recip_rank              all     0.8500
iprec_at_recall_0.00    all     0.8500
iprec_at_recall_0.10    all     0.6830
iprec_at_recall_0.20    all     0.6386
iprec_at_recall_0.30    all     0.5488
iprec_at_recall_0.40    all     0.2925
iprec_at_recall_0.50    all     0.2352
iprec_at_recall_0.60    all     0.1050
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.5200
P_10                    all     0.4000
P_15                    all     0.3733
P_20                    all     0.3100
P_30                    all     0.2067
P_100                   all     0.0620
P_200                   all     0.0310
P_500                   all     0.0124
P_1000                  all     0.0062
```

Μετά, εκτελούμε για το αρχείο myResults_mlt_3.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_mlt_3.txt

```
trec_eval.exe -q qrels.txt myResults_mlt_3.txt
```

Για το αρχείο myResults_mlt_3.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     71
map                     all     0.3212
gm_map                  all     0.1153
Rprec                   all     0.4166
bpref                   all     0.4795
recip_rank              all     0.7083
iprec_at_recall_0.00    all     0.7636
iprec_at_recall_0.10    all     0.7136
iprec_at_recall_0.20    all     0.6692
iprec_at_recall_0.30    all     0.6436
iprec_at_recall_0.40    all     0.4150
iprec_at_recall_0.50    all     0.2617
iprec_at_recall_0.60    all     0.2057
iprec_at_recall_0.70    all     0.1402
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.5200
P_10                    all     0.4700
P_15                    all     0.4333
P_20                    all     0.3550
P_30                    all     0.2367
P_100                   all     0.0710
P_200                   all     0.0355
P_500                   all     0.0142
P_1000                  all     0.0071
```

Τέλος, εκτελούμε για το αρχείο myResults_mlt_4.txt την εντολή:

trec_eval.exe -q qrels.txt myResults_mlt_4.txt

```
trec_eval.exe -q qrels.txt myResults_mlt_4.txt
```

Για το αρχείο myResults_mlt_4.txt το trec_eval μας έδωσε τα ακόλουθα αποτελέσματα:

```
runid                   all     es
num_q                   all     10
num_ret                 all     200
num_rel                 all     142
num_rel_ret             all     78
map                     all     0.4115
gm_map                  all     0.3525
Rprec                   all     0.4915
bpref                   all     0.5293
recip_rank              all     0.8833
iprec_at_recall_0.00    all     0.9100
iprec_at_recall_0.10    all     0.8367
iprec_at_recall_0.20    all     0.7836
iprec_at_recall_0.30    all     0.6459
iprec_at_recall_0.40    all     0.5752
iprec_at_recall_0.50    all     0.5002
iprec_at_recall_0.60    all     0.3868
iprec_at_recall_0.70    all     0.1647
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.7200
P_10                    all     0.5800
P_15                    all     0.4800
P_20                    all     0.3900
P_30                    all     0.2600
P_100                   all     0.0780
P_200                   all     0.0390
P_500                   all     0.0156
P_1000                  all     0.0078
```