

# ΑΝΑΚΤΗΣΗ ΠΛΗΡΟΦΟΡΙΑΣ

# Εισαγωγή

Στα πλαίσια του μαθήματος Ανάκτηση Πληροφορίας, με βάση την εκφώνηση της εργαστηριακής εργασίας, αναπτύχθηκε μια μηχανή αναζήτησης ακαδημαϊκών εργασιών η οποία έχει συγγραφεί σε γλώσσα προγραμματισμού Python.

Το πρόγραμμα αποτελείται από μια γραφική διεπαφή χρήστη(GUI) στην οποία οι χρήστες έχουν την δυνατότητα να αναζητούν και να ανακτούν τα έγγραφα με πολλαπλούς αλγόριθμους ανάκτησης και φίλτρα. Συγκεκριμένα αναπτύχθηκαν οι αλγόριθμοι ανάκτησης Boolean retrieval, Vector Space Model και Okapi BM25(Probabilistic retrieval model). Ως φίλτρα χρησιμοποιήθηκαν η ημερομηνία δημοσίευσης και ο συγγραφέας.

Η συλλογή των δεδομένων γίνεται από έναν σταχυολογητή (web crawler) από το αποθετήριο ακαδημαϊκών εργασιών arXiv. Στη συνέχεια, τα κειμενικά περιεχόμενα(abstract) που προκύπτουν από το προηγούμενο βήμα προεξεργάζονται. Έπειτα, δημιουργούμε μία ανεστραμμένη δομή δεδομένων ευρετηρίου (inverted index) για την αποτελεσματική αντιστοίχιση όρων στα έγγραφα στα οποία εμφανίζονται. Παράλληλα, υλοποιούμε ένα λεξικό(dictionary) για την αποθήκευση της αντιστοίχισης μεταξύ λέξεων και εγγράφων. Επιπροσθέτως, στο επόμενο βήμα, υλοποιούμε μία απλή διεπαφή ιστού και τους 3 αλγόριθμους ανάκτησης που αναφέραμε παραπάνω. Επιπλέον, δημιουργούμε μία συνάρτηση, ώστε να φιλτράρονται τα αποτελέσματα της ανάκτησης, με βάση ημερομηνία δημοσίευσης του άρθρου ή/και συγγραφέα του άρθρου που έχει ανακτηθεί. Αξίζει να τονιστεί το γεγονός ότι οι χρήστες μπορούν να αναζητούν έγγραφα χρησιμοποιώντας μία ή περισσότερες λέξεις(boolean πράξεις AND/OR/NOT). Τέλος, υλοποιούμε τον αλγόριθμο κατάταξης TF-IDF και τον cosine similarity, ώστε να γίνει κατάταξη των αποτελεσμάτων της ανάκτησης με πολλαπλούς τρόπους.

Στη συνέχεια του συγκεκριμένου αρχείου ακολουθεί η αξιολόγηση του συστήματος(Βήμα 5) και η τεκμηρίωση(Βήμα 6).

## Αξιολόγηση συστήματος (Βήμα 5°)

Για την αξιολόγηση του συστήματος υπάρχουν διάφορες μεθοδολογίες αλλά στην προκειμένη περίπτωση, τα σύνολα δεδομένων που έχουμε αποκομίσει από την ιστοσελίδα έχουν ταξινομηθεί από τον κώδικα μας (συνάρτηση **rank documents**), επομένως θα κινηθούμε με την μεθοδολογία της Ταξινομημένης Αξιολόγησης και τις μετρικές της. Μερικά από αυτά τα μέτρα είναι η καμπύλη Ακρίβειας-ανάκλησης, MAP, R-precision, η DCG 11-point precision και P@k.

Ακολουθεί μια σύντομη περιγραφή του κάθε μέτρου:

- **Precision-Recall Curve (Καμπύλη Precision-Recall):** Είναι ένα γράφημα που δείχνει το εύρος των τιμών της Precision και της Recall για διάφορα κατώφλια αποφάσεων ταξινόμησης.
- **MAP:** Μέση ακρίβεια (AP) για ένα ερώτημα  $q$  με σχετικά έγγραφα  $\{d_1, \dots, d_m\}$  είναι ο μέσος όρος των βαθμολογιών ακρίβειας που μετρούνται στις τάξεις των σχετικών εγγράφων, η MAP είναι η AP που υπολογίζεται κατά μέσο όρο στο σύνολο των ερωτημάτων  $Q$ ,

$$\text{MAP} = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} P(R_{jk})$$

- **R-Precision :** Είναι η ακρίβεια στο σημείο όπου ο αριθμός των ανακαλούμενων εγγράφων ισούται με το πλήθος των πραγματικών θετικών εγγράφων.
- **nDCG:** Μετρά την ποιότητα της σειράς των αποτελεσμάτων αναζήτησης, λαμβάνοντας υπόψη τη σημασία των θετικών εγγράφων.
- **11-point precision :** Η ακρίβεια 11 σημείων περιγράφει την απόδοση ενός συστήματος IR μέσω ακρίβειας που μετριέται σε 11 διαφορετικά επίπεδα ανάκλησης.
- **P@k (Precision at k - Ακρίβεια στο k):** Υπολογίζει την ακρίβεια των πρώτων  $k$  αποτελεσμάτων στη σειρά.

Σε διαφορετική περίπτωση, η αξιολόγηση του συστήματος θα μπορούσε να υλοποιηθεί με τις μετρικές Ακρίβεια / Ανάκληση / F(F1 score) με τις οποίες έχουμε ασχοληθεί σε θεωρητικό επίπεδο στο μάθημα της Θεωρίας. Πιο αναλυτικά:

- **Ακρίβεια:** είναι το ποσοστό των ανακτηθέντων εγγράφων που είναι συναφή

$$precision = true\_positives / (true\_positives + false\_positives)$$

- **Ανάκληση:** είναι το ποσοστό των συναφών εγγράφων που έχουν ανακτηθεί

$$recall = true\_positives / (true\_positives + false\_negatives)$$

- **F1 Score:** επιτρέπει την εναλλαγή ακρίβειας – ανάκλησης, δηλαδή είναι ο αρμονικός μέσος των δύο, καθώς η ακρίβεια με την ανάκληση, αλληλοαναιρούνται .

$$f1\_score = 2 * (precision * recall) / (precision + recall)$$

Στην συνέχεια, σε επίπεδο κώδικα για την αξιολόγηση της μηχανής αναζήτησης είναι αναγκαία η χρήση βιβλιοθηκών της Python για την υλοποίηση της αξιολόγησης. Μια τέτοια βιβλιοθήκη είναι η `pytrec_eval` . Για την εγκατάσταση της θα χρησιμοποιήσουμε την εντολή `-pip install pytrec_eval` στο terminal και για την χρήση της στον κώδικα την `import pytrec_eval`.

Συμπερασματικά, η επιλογή σωστής μετρικής εξαρτάται από τον τύπο του προβλήματος, τους στόχους της αξιολόγησης και ποια μορφή απόδοσης της μηχανής αναζήτησης θέλουμε να τονιστεί.

## Αναφορά και Τεκμηρίωση (Βήμα 6°)

Κατά την υλοποίηση του προγράμματος ξεκινήσαμε με την δημιουργία του web crawler, το οποίο είναι μια συνάρτηση(**crawler**) που συλλέγει δεδομένα(τίτλος, συγγραφείς, abstract και ημερομηνία) από το arXiv με τη βοήθεια του BeautifulSoup και τα αποθηκεύει σε ένα .json αρχείο (**arXiv\_results\_raw.json**).

Στην συνέχεια, προχωρήσαμε στην προεπεξεργασία των δεδομένων που συλλέχθηκαν από τον web crawler(συνάρτηση **preprocess\_text**). Πιο αναλυτικά, έγινε η αφαίρεση ειδικών χαρακτήρων, το tokenization, η κανονικοποίηση κειμένου, το Stemming, η αφαίρεση σημείων στίξης και προθημάτων (stop words) και η ενσωμάτωση των λέξεων σε ένα string. Τέλος, αυτά τα προεπεξεργασμένα δεδομένα καταχωρούνται στο arXiv\_results\_preprocessed.json αρχείο.

Έπειτα, έγινε η υλοποίηση του αντεστραμμένου ευρετηρίου(συνάρτηση **create\_inverted\_index**). Αρχικά η συνάρτηση συλλέγει τα έγγραφα και μετατρέπει το καθένα απ' αυτά σε λίστα στοιχείων. Μετά από κάποιους ελέγχους εντάσσονται στο ευρετήριο (**inverted\_index.json**) και αυτό ταξινομείται αλφαβητικά.

Για να προχωρήσουμε στο επόμενο βήμα, πραγματοποιήθηκε μια εκτενής έρευνα αλγορίθμων και πληροφοριών έτσι ώστε να υλοποιηθούν: η Μηχανή Αναζήτησης, η Επεξεργασία ερωτήματος και η Κατάταξη αποτελεσμάτων. Αρχικά, αναπτύχθηκε μια γραφική διεπαφή χρήστη (συνάρτηση **user\_interface**) στην οποία οι χρήστες έχουν την δυνατότητα να ανακτούν τα έγγραφα με τρεις διαφορετικούς αλγορίθμους. Για τη διεπαφή χρησιμοποιήσαμε τη βιβλιοθήκη της Python tkinter. Η διεπαφή αποτελείται από:

1. Πεδίο στο οποίο ο χρήστης εισάγει το ερώτημα του
2. 3 Radio buttons, ώστε να επιλέξει τον αλγόριθμο ανάκτησης της αρεσκείας του
3. 2 φίλτρα αναζήτησης. Τα φίλτρα δεν είναι υποχρεωτικά, δηλαδή κατά την αναζήτηση ακαδημαϊκών εργασιών γίνεται να εφαρμοστούν 1,2 ή και κανένα φίλτρο.
4. Κουμπί «Αναζήτηση», ώστε να ξεκινήσει η αναζήτηση(καλεί την εσωτερική συνάρτηση **search\_and\_display\_results**).

Η εσωτερική συνάρτηση **search\_and\_display\_results**, ασχολείται με την Επεξεργασία του Ερωτήματος του χρήστη. Αναλυτικότερα, λαμβάνει τις επιλογές του χρήστη στο GUI, και επιτελεί boolean πράξεις, στην περίπτωση που το ερώτημα αποτελείται από πάνω από 2 λέξεις. Παράλληλα, οφείλει να τονιστεί το γεγονός ότι για τις παραπάνω πράξεις δημιουργήσαμε τις βοηθητικές συναρτήσεις intersection, union, NOT. Τέλος, η εσωτερική συνάρτηση **search\_and\_display\_results** καλεί τη συνάρτηση **engine** με τα κατάλληλα ορίσματα, η οποία αποτελεί τη σημαντικότερη συνάρτηση του προγράμματος, Αναλυτικότερα, η **engine** καλεί αρχικά τους 3 αλγορίθμους ανάκτησης που δημιουργήσαμε(ανάλογα με την επιλογή του χρήστη στο GUI), οι οποίοι είναι ο Boolean retrieval, ο VSM (Vector Space Model) και ο Okapi BM25. συναρτήσεις **boolean\_retrieval**, **vector\_space\_model** και **okapi\_bm25** αντίστοιχα). Ειδικότερα, ο αλγόριθμος boolean ανάκτησης υλοποιήθηκε με βάση τη Θεωρία του μαθήματος. Επιτελεί τις βασικές boolean πράξεις. Από την άλλη πλευρά, ο VSM και ο Okapi BM25 δημιουργήθηκαν μέσω αναζήτησης σε διάφορες πηγές στο διαδίκτυο. Όσον αφορά τον VSM, αρχικά κάνουμε προεπεξεργασία του ερωτήματος του χρήστη. Έπειτα, υλοποιούμε τον αλγόριθμο κατάταξης TF-IDF για το ερώτημα(έννοιες που έχουμε δει στη Θεωρία). Στη συνέχεια, με στόχο να ταξινομήσουμε τα έγγραφα ως προς την

ομοιότητα(από τον ορισμό του VSM), υπολογίζουμε την ομοιότητα συνημίτονου μεταξύ του ερωτήματος και των εγγράφων. Τέλος, για τον αλγόριθμο Okapi BM25, αν θέλαμε να χρησιμοποιήσουμε ορισμό, είναι μια συνάρτηση κατάταξης που χρησιμοποιείται από τις μηχανές αναζήτησης για την εκτίμηση της συνάφειας των εγγράφων με ένα δεδομένο ερώτημα αναζήτησης. Ακολουθεί το μοντέλο bag-of-words και ως τυπικές τιμές των παραμέτρων του χρησιμοποιούμε τις εξής:  $k_1=1.5$  και  $b=0.75$ . Υπολογίζει το score αναζήτησης με βάση τη συνεισφορά κάθε όρου, λαμβάνοντας υπόψη τη συχνότητα εμφάνισης του όρου στο έγγραφο και το ερώτημα, καθώς και το μήκος του εγγράφου. Τα έγγραφα ταξινομούνται με βάση το score.

Σε αυτό το σημείο να υπενθυμίσουμε ότι συνεχίζεται η ανάλυση της συνάρτησης **engine**. Ανάλογα την επιλογή του χρήστη στο GUI επιλέγεται ο κατάλληλος αλγόριθμος και δημιουργούνται τα πρώτα αποτελέσματα(search\_results). Στη συνέχεια, ανακτούμε τα συναφή με την αναζήτηση έγγραφα, με χρήση του ανεστραμμένου ευρετηρίου, που αναλύσαμε παραπάνω. Παράλληλα, η **engine** καλεί άλλες 3 συναρτήσεις:

1. **rank\_documents**: Αρχικά γίνεται προεπεξεργασία του ερωτήματος. Στη συνέχεια συγχωνεύουμε τα συναφή έγγραφα σε μία συλλογή. Δημιουργήσαμε 2 τρόπους κατάταξης των αποτελεσμάτων. Πρώτα, υπολογίζουμε το TF-IDF(βασικός αλγόριθμος κατάταξης) για το συγκεκριμένο ερώτημα. Στη συνέχεια ταξινομούμε τα έγγραφα ως προς την ομοιότητα συνημίτονου μεταξύ του ερωτήματος και των εγγράφων(πιο προηγμένη τεχνική κατάταξης).
2. **filter\_results**: Σε αυτή την συνάρτηση, έχοντας ως όρισμα τα αποτελέσματα της ανάκτησης, υλοποιούμε(εάν το επιλέξει ο χρήστης στο GUI) φίλτρα με βάση την ημερομηνία δημοσίευσης και το συγγραφέα του άρθρου που αναζητεί ο χρήστης. Γίνεται δηλαδή μία επιπλέον αναζήτηση στα αποτελέσματα, με σκοπό να βρεθούν συγκεκριμένα άρθρα, που συμβαδίζουν με τα παραπάνω φίλτρα.
3. **print\_results**: Τέλος, η συγκεκριμένη συνάρτηση, έχει ως εργασία να εκτυπώσει τα τελικά αποτελέσματα στην οθόνη του χρήστη, σε φιλική μορφή(χρήση f-strings).

### Σημειώσεις:

- 1) Το url της σελίδας που θα γίνει το crawling είναι το εξής:  
<https://arxiv.org/list/cs/new>
- 2) Ο αριθμός των ακαδημαϊκών εργασιών που χρησιμοποιούμε στο σύνολο του προγράμματος είναι 150, αλλά ο κώδικας έχει δημιουργηθεί με δυναμικό τρόπο, ώστε να λειτουργεί για οποιοδήποτε αριθμό εργασιών.

Εν κατακλείδι, θα αναφέρουμε τις συναρτήσεις που καλούνται στη συνάρτηση `__main__`:

- **crawler**
- **save\_to\_json**(Αποθήκευση αρχείου JSON ΠΡΙΝ από το *preprocess*)
- **preprocess\_text**
- **save\_to\_json**(Αποθήκευση αρχείου JSON META από το *preprocess*)
- **create\_inverted\_index**
- **save\_to\_json**(Αποθήκευση του ανεστραμμένου ευρετηρίου)
- **user\_interface**(Καλεί όλες τις υπόλοιπες συναρτήσεις που εξηγήσαμε αναλυτικά παραπάνω)

Ακολουθούν φωτογραφίες που επιδεικνύουν την λειτουργικότητα του προγράμματος:

```
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\marin\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
```

- arXiv\_results\_preprocessed.json
- arXiv\_results\_raw.json
- inverted\_index.json



## arXiv\_results\_raw.json

```
{
  "arXiv_results_raw.json" > ...
  1  [
  2    {
  3      "title": "Title:Transduce: learning transduction grammars for string transformation",
  4      "authors": [
  5        " ",
  6        "Authors:",
  7        " ",
  8        "Francis Frydman",
  9        " ",
  10       "Philippe Mangion",
  11       " "
  12     ],
  13     "abstract": "The synthesis of string transformation programs from input-output examples\nutilizes various techniques, all based on an inductive bias that compr",
  14     "date": "Not Found"
  15   },
  16   {
  17     "title": "Title:RoleCraft-GLM: Advancing Personalized Role-Playing in Large Language Models",
  18     "authors": [
  19       " ",
  20       "Authors:",
  21       " ",
  22       "Meiling Tao",
  23       " ",
  24       "Xuechen Liang",
  25       " ",
  26       "Tianyu Shi",
  27       " ",
  28       "Lei Yu",
  29       " ",
  30       "Yiting Xie",
  31       " "
  32     ],
  33     "abstract": "This study presents RoleCraft-GLM, an innovative framework aimed at enhancing\npersonalized role-playing with Large Language Models (LLMs). RoleCr",
  34     "date": "Not Found"
  35   },
  36   {
  37     "title": "Title:Modeling, Simulation, and Maneuvering Control of a Generic Submarine",
  38     "authors": [
  39       " ",
  40       "Authors:",
  41       " ",
  42       "Gage MacLin",
  43       " ",
  44       "Maxwell Hammond",
  45       " ",
  46       "Venzazio Cichella",
  47       " ",
  48       "J. Ezequiel Martin",
  49       " "
  50     ],
  51     "abstract": "thi work introduc two multilevel control strategi to address the problem of guidanc and control of underwat vehicl an outerloop pathfollow algorith",
  52     "date": "Not Found"
  53   },
  54   {
  55     "title": "Title:Voxceleb-ESP: preliminary experiments detecting Spanish celebrities from their voices",
  56     "authors": [
  57       " ",
  58       "Authors:",
  59       " ",
  60       "Beltr\u00e9n Labrador",
  61       " ",
  62       "Manuel Otero-Gonzalez",
  63       " ",
  64       "Alicia Lozano-Diez",
  65       " ",
  66       "Daniel Ramos",
  67       " ",
  68       "Doroteo T. Toledano",
  69       " ",
  70       "Joaquin Gonzalez-Rodriguez",
  71       " "
  72     ],
  73     "abstract": "thi paper present voxcelebesp a collect of pointer and timestamp to youtub video facilit the creation of a novel speaker recognit dataset voxcelebe",
  74     "date": "Not Found"
  75   },
  76   {
  77     "title": "Title:Object Attribute Matters in Visual Question Answering",
  78     "authors": [
  79       " ",
  80       "Authors:",
  81       " ",
  82       "Peize Li",
  83       " ",
  84       "Qingyi Si",
  85       " ",
  86       "Peng Fu",
  87       " ",
  88       "Zheng Lin",
  89       " ",
  90       "Yan Wang",
  91       " "
  92     ],
  93     "abstract": "visual question answer is a multimod task that requir the joint comprehens of visual and textual inform howev integr visual and textual semant sole",
  94     "date": "Not Found"
  95   },
  96 ]
}
```

## arXiv\_results\_preprocessed.json


```
{
  "arXiv_results_preprocessed.json" > ...
  48   "J. Ezequiel Martin",
  49   " "
  50 ],
  51 "abstract": "thi work introduc two multilevel control strategi to address the problem of guidanc and control of underwat vehicl an outerloop pathfollow algorith",
  52 "date": "Not Found"
  53 },
  54 {
  55   "title": "Title:Voxceleb-ESP: preliminary experiments detecting Spanish celebrities from their voices",
  56   "authors": [
  57     " ",
  58     "Authors:",
  59     " ",
  60     "Beltr\u00e9n Labrador",
  61     " ",
  62     "Manuel Otero-Gonzalez",
  63     " ",
  64     "Alicia Lozano-Diez",
  65     " ",
  66     "Daniel Ramos",
  67     " ",
  68     "Doroteo T. Toledano",
  69     " ",
  70     "Joaquin Gonzalez-Rodriguez",
  71     " "
  72   ],
  73   "abstract": "thi paper present voxcelebesp a collect of pointer and timestamp to youtub video facilit the creation of a novel speaker recognit dataset voxcelebe",
  74   "date": "Not Found"
  75 },
  76 {
  77   "title": "Title:Object Attribute Matters in Visual Question Answering",
  78   "authors": [
  79     " ",
  80     "Authors:",
  81     " ",
  82     "Peize Li",
  83     " ",
  84     "Qingyi Si",
  85     " ",
  86     "Peng Fu",
  87     " ",
  88     "Zheng Lin",
  89     " ",
  90     "Yan Wang",
  91     " "
  92   ],
  93   "abstract": "visual question answer is a multimod task that requir the joint comprehens of visual and textual inform howev integr visual and textual semant sole",
  94   "date": "Not Found"
  95 },
  96 ]
}
```

## inverted\_index.json

```
{
  "inverted_index.json": {
    "abil": [
      37,
      70,
      6,
      136,
      105,
      77,
      144,
      28,
      95
    ],
    "abilitybecau": [
      38
    ],
    "abl": [
      50,
      95
    ],
    "ablat": [
      16
    ],
    "abnorm": [
      144
    ],
    "about": [
      137,
      75,
      11,
      143,
      84,
      150,
      59
    ],
    "abov": [
      70
    ],
    "abruptli": [
      125
    ],
    "absenc": [
      40
    ],
    "absent": [
      92
    ],
    "absolut": [
      25
    ]
  }
}
```

## Διεπαφή χρήστη(GUI) για 2 διαφορετικά ερωτήματα χρηστών

### Παράδειγμα 1(Απλό ερώτημα)

 Διεπαφή χρήστη για την αναζήτηση ακαδημαϊκών εργασιών

Ερώτημα χρήστη

access

Επιλογή αλγόριθμου ανάκτησης

☒ Boolean retrieval

☐ Vector Space Model(VSM)

☐ Probabilistic retrieval model(Okapi BM25)

Φίλτρα αναζήτησης

☐ Ημερομηνία δημοσίευσης

☐ Συγγραφέας

Αναζήτηση

## Παράδειγμα 2(Επίδειξη λειτουργίας φίλτρων)

Διεπαφή χρήστη για την αναζήτηση ακαδημαϊκών εργασιών

Ερώτημα χρήστη  
access AND accept

Επιλογή αλγόριθμου ανάκτησης

☐ Boolean retrieval  
☒ Vector Space Model(VSM)  
☐ Probabilistic retrieval model(Okapi BM25)

Φίλτρα αναζήτησης

☐ Ημερομηνία δημοσίευσης  
☒ Συγγραφέας

Francesco Pasa

Αναζήτηση

## Αποτελέσματα που προκύπτουν από το 2° ερώτημα χρήστη

```
> doc_num = {list: 150} [{'abstract': 'the synthesi of string transform program from inputoutput exampl util variou techniqu all base on an induct bia that compris ...str transdu ... View}
documents = {set: 1} {55}
  01 {int} 55
  01 _len_ = {int} 1
> Protected Attributes
  01 i = {int} 150
> inverted_index = {dict: 3227} {'abil': [37, 70, 6, 136, 105, 77, 144, 28, 95], 'abilitybecau': [38], 'abl': [50, 95], 'ablat': [16], 'abnorm': [144], 'about': [137, 75, 11, 143, 84, 150, 5 ... View
  01 preprocessed_abstract = {str} 'onlin content platform commonli use engagementbas optim when make recommend thi encourag content creator to invest in qualiti but also ... View
> res = {dict: 4} {'abstract': 'onlin content platform commonli use engagementbas optim when make recommend thi encourag content creator to invest in qualiti...aliz engag re... View
> results = {list: 580} [{'abstract': 'the synthesi of string transform program from inputoutput exampl util variou techniqu all base on an induct bia that compris ...str transduc l... View
  01 term = {str} 'zt'
  01 url = {str} 'https://arxiv.org/list/cs/new'
> Special Variables
```