Sessió 3: Rocchio

Elena Ruiz
Miriam Vall

1. Implementació d'un URF mitjançant la regla de Rocchio

L'esquema que finalment s'ha utilitzat ha sigut partint del fitxer original **SearchIndexWeight.py**, on s'ha reutilitzat el codi fins on es cridava l'api d'*ElasticSearch* amb l'objectiu d'agafar els k primers documents.

A part d'aquest codi esmentat, s'ha introduït totes les funcions necessàries per poder realitzar el càlcul dels TFIDF que formaven part del fitxer **TFIDF.py** de la anterior sessió.

Les funcions noves que han sigut implementades per a computar la regla de Rocchio en diverses iteracions són les següents:

- **query2list**: Transforma la consulta en format de string que s'introdueïx a l'hora de cridar el fixter. Retorna la mateixa consulta emmagatzemat en un diccionari amb la forma { 'terme' : 'pes'}
- **list2query:** El mateix que la funció anterior però a la inversa.
- **calculateTFIDFofList**: Crea un diccionari on guarda a cada element la informació de: { 'terme' : sum(d1 + …+dk) } On el segon element és la suma dels TFIFD dels k documents de més rellevància.
- **RocchioRule:** Donat alfa,beta, el pes de que determina l'usuari sobre la paraula, el valor de sum(d1 + …+dk), calculat anteriorment, i el nombre de documents rellevants. S'aplica per a cada terme de la consulta el seu nou pes per a fer la consulta, segons la formula de Rocchio, donada en el pdf de la sessió.

A partir de les funcions anteriors implementades es dur a terme el següent procediment per fer una consulta seguint el USF.

1. Es crida la API per fer la consulta original i agafar els k documents més rellevants
2. Es transforma la consulta en un diccionari cridant a **query2list**
3. Es crida la funció **calculateTFIDFofList** per obtenir tots elsTFIDF dels primers k documents rellevants.
4. Llavors comença el procediment per aplicar la regla de Rocchio iterativament:
    a. Es crida la funció de **RocchioRule,** per recalcular els nous pesos d'importància de la consulta
    b. Es transforma la llista dels nous pesos dels termes a string
    c. Es crida la API per dur a terme una nova consulta
    d. Es recalcula els TFIFD dels nous k documents més rellevants

5. Es fa **nrounds** iteracions, però en la última iteració només es fan els passos ‘a’ i ‘b’ del punt anterior.
6. Treu per pantalla els documents més rellevants.

La dificultat més gran que hem trobat ha sigut entendre completament el codi per saber en quin moment s'havia d'aplicar els mètodes implementats o per implementar. Però sobretot el pensar com estructurar correctament el codi per realitzar el càlcul. Ja que al no partir d’un esquema en concret i haver de reutilitzar codi de la sessió anterior donava una dificultat major. Altre petita dificultat a sigut la necessitat d’utilitzar diccionaris i haver de fer petites modificacions dels codis de les sessions anteriors que potser s’havien implementat en llistes.

2. Experimentació

EXPERIMENT 1: Consulta de l’exemple, quan només hi ha un document rellevant
L’objectiu d’aquest experiment, més que observar quin tipus de documents es donen com a resultats, és per saber com recalcular a cada iteració els nous pesos d’importància de cada terme la regla de Rocchio.
Per aquest experiment s’executa amb els següents parametres:

1. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** toronto nyc
2. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** toronto nyc^2
3. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** toronto^2 nyc

És a dir quan **alpha = 2*beta**.
Les consultes de cada iteració es troben al fitxer exp1.1-i.txt. I es pot observar com a cada iteració els pesos es van incrementant sobre potencies de dos sobre la iteració anterior partint del seu pes inicial. Exemple de la sortida en el cas 2:

- IT: 1 QUERY =['toronto', 'nyc^2']
- IT: 2 QUERY =toronto^2 nyc^4
- IT: 3 QUERY =toronto^4 nyc^8
- IT: 4 QUERY =toronto^8 nyc^16
- IT: 5 QUERY =toronto^16 nyc^32
- IT: 6 QUERY =toronto^32 nyc^64
- IT: 7 QUERY =toronto^64 nyc^128
- IT: 8 QUERY =toronto^128 nyc^256
- IT: 9 QUERY =toronto^256 nyc^512
- IT: 10 QUERY =toronto^512 nyc^1024

Fem el mateix amb **alpha = 3 i beta = 2** i **alpha = 4 i beta = 1 ,** i observem que els pesos es van incrementant en potencies de 3 i de 4 respectivament. Llavors és aquí on s’observa que en aquests experiments el nou pes de rellevància de cada terme bé determinat només pel valor de alfa, possiblement perquè justament en aquesta consulta només tenim com a resultat un únic document rellevant. Per això es farà un segon experiment amb el mateix procediment però amb altre query original.

EXPERIMENT 2: Consulta amb resultats de més d'un document rellevant
Per aquest experiment es fa el mateix, però amb termes que existeixin més resultats als nostres fitxers de cerca.
En el primer experiment, on es guarden als fitxers exp2.1-i.txt, s'ha cridat el fitxer de la següent manera:
Per aquest experiment s'executa amb els següents parametres:

1. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** all the
2. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** all the^2
3. **--index** news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** all^2 the

Tant quan executem aquests paràmetres, com quan també ho fem amb **alpha 3 i beta 2, i alpha 4 i beta 1.** Els pesos d'importància de cada terme donen exactament igual als experiments anteriors. Per això podem afirmar que:

- Els documents resultats no han variat segons els valors de alfa i beta
- Els documents rellevants han variat segon el pes d'importància que inicialment ha donat l'usuari.

I els documents resultats no varien segons els valors dels paràmetres de alfa i beta, sinó, en tot cas, **depenent dels valors inicials d'importància que ha decidit posar l'usuari.**

EXPERIMENT 3: Mateixos paràmetres, diferents resultats a cada iteració

Donat un valor de alfa i beta indiferent, es vol veure si existeix algun canvi de rellevància en els documents a cada iteració. Executant:

--index news **--nhits** 5 **--nrounds** 10 **-R** 10 **--alpha** 2 **--beta** 1 **--query** all the^2

Efectivament, la rellevància dels documents pot canviar a cada iteració. A cada aplicació de la regla de Rocchio, hi ha una major distància a la consulta original, i es poden incorporar nous termes i considerar més documents, fent que cada cop es dona com a resultat documents més relevants segons la puntuació obtinguda.

Això ho podem veure en el nostre fitxer exp3.txt on en la primera iteració el document més rellevant obté una **puntuació de 2.16**, pero finalment el document més rellevant ho és el que ha tingut més puntuació de la iteració 2 a la última, **incrementant el score fins a 65.73.**

3 .Anàlisis i conclusions

Dels experiments realitzats hem pogut obtenir unes quantes conclusions. En primer lloc, la influència de cada paràmetre de la regla de Rocchio sobre la rellevància dels documents obtinguts en les consultes, i per tant en la precisió i el recall daquestes.

Observant la fòrmula donada per calcular una nova consulta a partir d'una altra ja existent aplicant la regla de Rocchio, podem deduïr com influeix la mida dels paràmetres alpha i beta en la creació de la nova consulta. Per exemple, observem que com menor és alpha, menys termes

de la consulta original tindrà la nova consulta (en el cas que alpha = 0, cap terme de la original). Pel que fa a beta, com menor sigui, menys termes nous incorporarà la nova consulta.

En el primer experiment es comprova com en cada iteració de nrounds la proporció d'alpha respecte beta és la mateixa amb la que augmenten els pesos dels termes. En el segon s'han obtingut els mateixos resultats. En aquests experiments, s'ha vist que els documents han variat en rellevància degut als valors d'importància determinats inicialment; per tant, alpha i beta influeixen en la "quantitat" de canvi observat en aquests valors, però els documents rellevants que obtenim finalment dependran fonamentalment de quina importància tenien originalment els termes.

Les variacions en nrounds, el nombre d'aplicacions de la regla de Rocchio, les hem provat en el tercer experiment. Ja que cada aplicació de la regla implica que la modificació respecte la consulta original sigui més gran, el millor seria intentar no sobrepassar el nombre de nrounds en el que es perd massa precisió, encara que també ens interessi millorar el recall. En el tercer experiment veiem que l'ordre de rellevància dels documents pot variar en cada iteració, i si aquesta variació és massa gran, la precisió queda afectada.

Ja que l'objectiu del Pseudorelevance Feedback és obtenir els documents més rellevants sense tenir cap més interacció amb l'usuari que no sigui la original, al generar una nova consulta amb termes que poden ser diferents dels que s'introduien originalment, la precisió es veu afectada per aquest fet, però no obstant el recall millora. Menys precisió implica més termes i documents consultats innecessàriament, però més recall implica que es consultin documents no considerats abans, fent que se'ns escapi més poca cosa.

D'altre banda, pel que fa al paràmetre k (nombre de documents considerats rellevants), un aspecte que es veu afectat per la seva variació és el recall, perquè si k és més petit que el nombre total de documents, n'hi ha algun/s que no tenim en compte.

Finalment, a l'hora de seleccionar R (nombre de termes que es mantenen en la nova consulta, com a rellevants), s'ha de tenir en compte que com més alta és la quantitat de termes considerats «més rellevants», menys rellevància tenen cadascun individualment, i es perd precisió.