CAIM Laboratory

Session 6: MapReduce and Document clustering

2019-2020 Q1

Pol Renau Larrodé

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

1. Experimentació

Tota l'experimentació s'ha realitzat amb el corpus de dades arxiv_abs, que conté uns 53 mil documents.

1.1 Variació dels paràmetres min freq i max freq

- Per a la següent fase d'experimentació, mantenim el nombre de iteracions 5, fixem el nombre de paraules en 200 i aumentem el valor del nombre de clústers fins a 20, per poder tenir un major angle de comparativa. I variarem la freqüència mínima i màxima amb els valors [0.01, 0.1, 0.2 ,0.3,0.5,0.6] i [0.05,0.3, 0.5,0.4,0.7,0.9] respectivament.

	Experiments					
	1	2	3	4	5	6
	F min | F max	F min | F max	F min | F max	F min | F max	F min | F max	F min | F max
Iteració	0.01 | 0.05	0.1 | 0.3	0.2 | 0.5	0.3 | 0.4	0.5 | 0.7	0.6 | 0.9
1	4.119841 s	6.922094 s	5.020280 s	3.175290 s	2.605821 s	2.456798 s
2	6.903681 s	20.577772 s	10.057895 s	4.000032 s	2.906527 s	2.370702 s
3	10.230542 s	5.270988 s	4.042372 s	2.873244 s	2.526663 s	2.460127 s
4	4.763939 s	-	4.330534 s	4.455336 s	2.599027 s	2.372718 s
5	-	-	4.745323 s	4.248366 s	2.489427 s	2.713000 s
Classes	17, 15 i 9 (3)	2, 4 i 1 (3)	13 i 7 (2)	11 (1)	6 (1)	7 (1)

- Com podem observar si el rang de freqüència és més gran, s'augmenten el nombre de classes obtingudes, això és bastant evident , ja que a major rang més possibles paraules que representin cada classe. No ens ha sorprès que al executar amb freqüències baixes i d'un rang moderat ens hagi fet més classes, que a la resta, ja que aquelles paraules poc freqüents que representen a una classe hi ha poca probabilitat de que estiguin en la resta de docs.
- Per les següents fases d'experimentació fixarem el valor de freqüència mínima en 0.2 i el de màxima en 0.5. Ja que és el rang que considerem més adequat analitzant el corpus sobre el qual treballem.

1.2 Variant el nombre de cores i el nombre de paraules

- Per la primera fase d'aquesta experimentació, variarem el nombre de cores, i s'executarà amb 100 paraules. En la segona fase executarem amb 200 paraules i farem una comparativa dels resultats obtinguts.

Experimentació amb 100 paraules				
	Número de cores			
	1	2	3	4
1	5,512212	5,010363	3,787190	3,530246
2	17,384045	10,786899	8,486829	10,498108
3	8,361328	5,474461	4,535341	4,847224
4	-	-	-	-
5	-	-	-	-
Total	31,257585	21,271723	16,809360	18,875578

Experimentació amb 200 paraules				
	Número de cores			
	1	2	3	4
1	7,986180	4,558936	4,185933	5,227508
2	16,854942	13,216136	9,557642	8,687102
3	8,822157	5,495906	4,492483	6,003151
4	-	-	-	-
5	-	-	-	-
Total	33,663279	23,270978	18,236058	19,917761

- Com podem apreciar, aumentar el número de cores implica majoritàriament reduir el temps d'execució total. No sempre ja que per exemple en els dos casos veiem que és pitjor en quant a temps usar 4 cores que usar 3, això pot estar degut a que la repartició de feina no és necessari que hi hagi una partició en 4 ja que el temps no es pot optimitzar més. I per tant incrementa el temps d'execució, ja que hem de gastar més temps al moment de composar la solució. També apreciem que a major nombre de paraules major es el temps.

1.3 Paraules més rellevants (menor freqüència)

- ➢ Hem executat el MRKmeans.py amb el extract data d'aquelles paraules amb freqüències reduïdes (0.01 i 0.03) aquestes ens han permès identificar aquells grups que es poden formar dins del corpus de dades. Com hem explicat anteriorment hem obtingut un total de 3 clusters. No ens ha sorprès ja que els temes que tracta són temes molt similars, tots els papers del cs, així que vol dir que són documents de recerca en diferents àmbits.
- ➢ Clusters obtinguts:
    - Com podem veure la classe 4 evidentment parla de temes de física, per les paraules que veiem com supernova, relativist, phenomena.

    - En la classe 9 (segon cluster obtingut) veiem que conte les paraules com "spars" ,, "secur" i "mobil". Creiem que aquest va més encaminat cap a la ciberseguretat.

    - Finalment tenim un cluster que ens diu "deploy", "mòbil" i "site", podem entendre que aquest grup té més a veure amb plataformes i coses per l'estil.

2. Implementació i problemes

- ➢ En quan a la implementació de codi, hem realitzat les coses que ens demanava en la documentació, adaptades als problemes que ens havien passat. Per exemple la funció assign_prototype: retornem una tupla que conté el Prototip assignat i un parell que indica el ID del document i la llista de paraules d'aquest.
- ➢ En quant al aggregate_prototype: retornem una tupla que conté la clau del prototip assignat i un pair que conté la llista de paraules del cluster ordenades per ordre alfabètic.
- ➢ Finalment tenim la funció de Jaccard que simplement era aplicar la funció de la forma més òptima que trobada, que ha estat aprofitant que estaven ordenades alfabèticament.
- ➢ Hem generat uns quants scripts que ens han permès fer l'experimentació, també els adjuntarem a la entrega. (runScript.py, runScript2.py)

- ➢ No hi ha hagut molt problema en quant a la implementació un cop entès el funcionament del mapreduce.

3. Conclusions

- ➢ Si volem tenir una agrupació vàlida dels grups haurem de posar freqüències baixes per tal de que les paraules que representin als clusters, siguin específiques de certs temes.
- ➢ Si volem una agrupació més general doncs no haurem de tractar freqüències molt baixes però tampoc molt altes, com les que hem usat al llarg de l'experimentació 0.2 i 0.5 serien uns valors bastant adients.
- ➢ Respecte al nombre de paraules hem vist que si reduïm el nombre de paraules, els clústers canvien, ja que les paraules que representaran el cluster estaran més limitades.