Cerca i Anàlisi d'Informació Massiva
Clustering de documents amb k-means en MapReduce

Marc Canals Riba Joaquin Figueira Chacón

16 de desembre de 2021

Índex

1 Introducció

Dividim un conjunt de documents en categories, clústers de documents similars, mitjançant l'algorisme K-means. Implementem l'algorisme com un programa de tipus MapReduce, que podem executar paral·lelament en sistemes distribuïts. Analitzem l'efecte de la freqüència de les paraules del vocabulari, la mida del vocabulari, el nombre de clústers, el nombre de processos i la representativitat de les paraules més freqüents dels prototips dels clústers resultants.

2 K-means en MapReduce

Aquest algorisme de *clustering* es basa en l'aplicació reiterada de dues accions: l'assignació de cada document al prototip més semblant —podem calcular la semblança amb l'índex de Jaccard (1)— i el càlcul del prototip de cada clúster a partir dels documents assignats (2). Podem utilitzar qualsevol document com a prototip inicial. L'algorisme acaba quan no hi ha canvis entre dues iteracions consecutives o s'assoleix un nombre determinat d'iteracions.

Representem els documents amb vectors de valors booleans i els prototips amb vectors de probabilitats, tots dos de la mida del vocabulari —cada valor correspon a una paraula del vocabulari—. Podem interpretar l'índex de Jaccard com el pes de la intersecció de dos documents respecte del de la unió dels documents.

$$jaccard(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\|^2 + \|d_2\|^2 - d_1 \cdot d_2} \qquad (1) \qquad\qquad prototype(D) = \frac{1}{|D|} \cdot (\sum_{d \in D} d_1, \sum_{d \in D} d_2, \ \ldots) \qquad (2)$$

Podem implementar-lo com un programa de MapReduce en què la funció *map* correspon a l'assignació (3) i la funció *reduce* al càlcul de prototips (4). D'aquesta manera, podem paral·lelitzar els càlculs.

$$\begin{aligned} &map\ (clusters,\ document) \rightarrow (cluster,\ document) : \\ &\quad (argmax\ \{jaccard\ (prototype_{cluster},\ document) : cluster \in clusters\},\ document) \end{aligned} \qquad (3)$$

$$\begin{aligned} &reduce\ (cluster,\ documents) \rightarrow (cluster,\ prototype) : \\ &\quad prototype\ (documents) \end{aligned} \qquad (4)$$

3 Implementació

Implementem el programa amb la llibreria de Python `MRJob`. La implementació consta de diferents fitxers:

- `ExtractData.py`: generació dels documents.
- `GeneratePrototypes.py`: generació dels prototips inicials.
- `MRKmeansStep.py`: implementació de l'algorisme.
- `MRKmeans.py`: execució de l'algorisme.

El programa de generació de documents accepta parells de freqüències de document normalitzades mínimes i màximes com a arguments. D'aquesta manera, generem múltiples conjunts de documents en una sola execució i no repetim operacions lentes com ara l'obtenció de documents.

Inicialment representem els documents amb llistes de paraules ordenades i els prototips amb llistes de parells paraula-probabilitat ordenats per paraula; a l'hora d'obtenir l'índex de Jaccard, calculem el producte escalar amb l'algorisme de fusió de llistes ordenades. Observem que els documents contenen moltes menys paraules que els prototips, per tant, representem els prototips amb diccionaris paraula-probabilitat i calculem el producte escalar cercant cada paraula del document al prototip; com que l'ordre ja no importa, no ordenem els prototips calculats, tot i que la supressió d'aquesta operació no afecta gaire el temps d'execució —hi ha pocs prototips—. El temps d'execució total es redueix aproximadament a un terç de l'inicial.

També observem que a l'hora de calcular el quadrat de la norma del prototip, és molt més eficient calcular `p * p` que `p ** 2`, on `p` és una probabilitat del prototip.

4 Experimentació

4.1 Freqüència i mida del vocabulari

Durant la generació dels documents, podem indicar la freqüència de document normalitzada mínima m (opció `-m` o `--mindf`) i màxima M (opció `-M` o `--maxdf`) de les paraules dels documents als quals apliquem l'algorisme. En altres paraules, m i M són dos valors reals tals que $0 \leq m \leq M \leq 1$ i $\forall\ w \in W\ m \leq n_w\ /\ max\ \{n_w : w \in W\} \leq M$, on W és el vocabulari i n_w el nombre de documents que contenen la paraula w.

També podem indicar la mida del vocabulari (opció `-n` o `--words`), és a dir, el nombre de paraules diferents que contenen els documents generats, tot i que és possible d'expressar-la en funció de m i M.

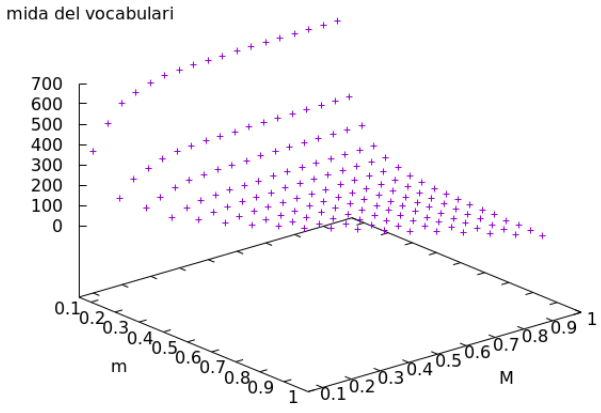

Figura 1: mida del vocabulari en funció de m i M.

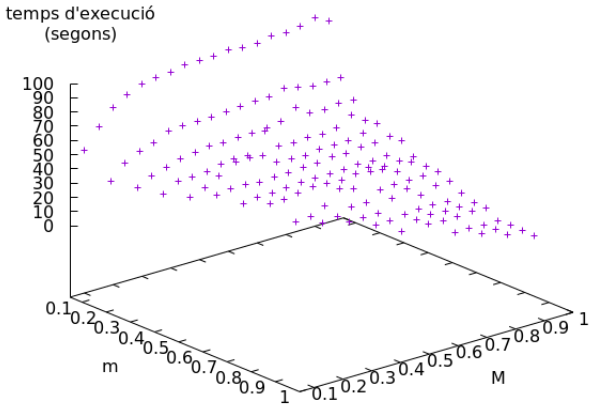

Figura 2: temps d'execució (segons) en funció de m i M (amb 10 clústers inicials i 5 iteracions com a màxim).

Amb una M determinada, quan m disminueix, la mida del vocabulari augmenta a un ritme creixent, perquè incloem paraules menys freqüents, que són més nombroses; en canvi, amb una m determinada, quan M augmenta, augmenta a un ritme decreixent, perquè incloem paraules més freqüents, que són menys nombroses (figura 1).

El temps d'execució sembla proporcional a la mida del vocabulari (figura 2). Observem que el de la primera iteració és menor que els de les següents iteracions, perquè a partir de la segona iteració els prototips contenen totes les paraules dels documents assignats al clúster, mentre que a la primera només contenen les del prototip inicial, en el nostre cas, un document qualsevol. Si després d'una iteració hi ha menys clústers el temps d'execució de les iteracions posteriors també disminueix. Considerem que a partir de $m \approx 0.05$ l'execució és força lenta i que 0.05 és una cota inferior acceptable.

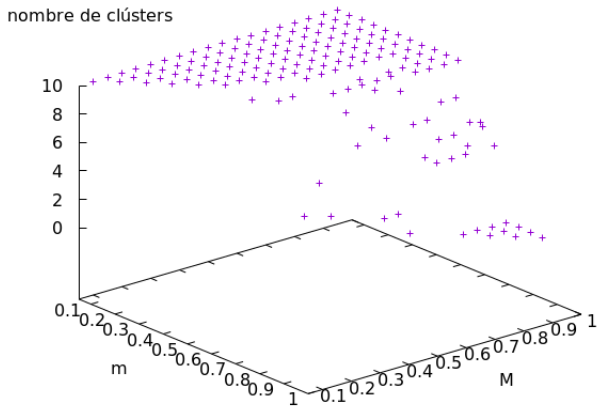

Figura 3: nombre de clústers resultants en funció de m i M (amb 10 clústers inicials i 5 iteracions com a màxim).

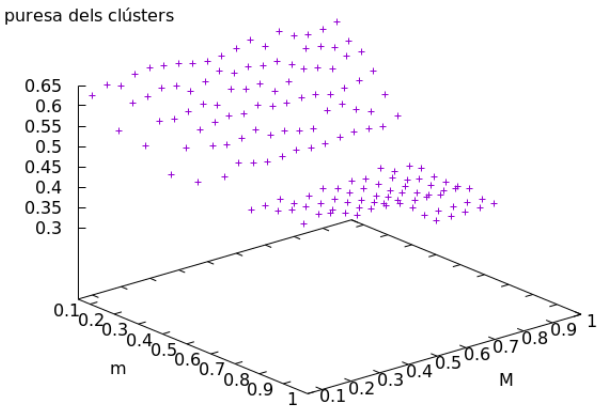

Figura 4: puresa dels clústers resultants en funció de m i M (amb 10 clústers inicials i 5 iteracions com a màxim).

Quan les paraules són freqüents els documents s'assemblen molt i el nombre de clústers disminueix, mentre que quan no ho són els documents s'assemblen poc i el nombre de clústers no varia (figura 3), tot i que si són molt poc freqüents l'assignació pot ser molt aleatòria. Com que a partir de $m \approx 0.3$ els documents s'assemblen tant que el nombre de clústers disminueix, considerem que 0.3 és una cota superior acceptable.

Provem d'avaluar l'assignació mesurant la puresa [1] dels clústers (figura 4): sumem el nombre de documents del grup [1] majoritari de cada clúster, els quals considerem que estan ben assignats, i dividim aquesta suma entre el total de documents. A partir de $m \approx 0.3$ la puresa dels clústers és gairebé $\frac{17970}{55535} \approx 0.32$, la que obtenim amb una assignació uniforme, per tant, mantenim 0.3 com a cota superior.

Havent acotat m i M analitzem les paraules dels prototips finals i considerem que amb $m = 0.05$ i $M = 0.1$ les paraules més freqüents dels prototips permeten d'entreveure de què tracten els documents assignats a cada clúster.

4.2 Nombre de clústers

Durant la generació dels prototips, podem indicar el nombre de clústers inicials (opció `-n` o `--clusters`). Analitzem l'efecte del nombre de clústers amb les 100, 250 i 347 (totes) paraules més freqüents del vocabulari del conjunt de documents generat amb $m = 0.05$ i $M = 0.1$.

El temps d'execució, amb 5 iteracions com a màxim, en tots tres casos és proporcional al nombre de clústers inicials (figura 5). Caldria veure com es comporta si no limitem el nombre d'iteracions.

Provem de determinar el nombre òptim de clústers amb el mètode del colze [2]: calculem la suma dels quadrats (5) i identifiquem el punt on el ritme de creixement disminueix, és a dir, a partir del qual els clústers aporten poca informació addicional [3]. Sembla que aquest punt es troba entre els 5 i els 15 clústers (figura 6), per tant, 10 clústers pot ser una bona elecció.

$$TSS = \sum_{c \in C} \left(\sum_{d \in c} \left(jaccard(prototype_c, d) - \frac{1}{|c|} \cdot \sum_{d \in c} jaccard(prototype_c, d) \right) \right) \tag{5}$$

[1] Els documents són resums d'articles científics d'ArXiv.org i cadascun pertany a una de les següents categories: `astro-ph` (13420), `cond-mat` (5094), `cs` (17970), `hep-ph` (1615), `hep-th` (1467), `math` (7144), `physics` (6981) o `quant-ph` (1844).

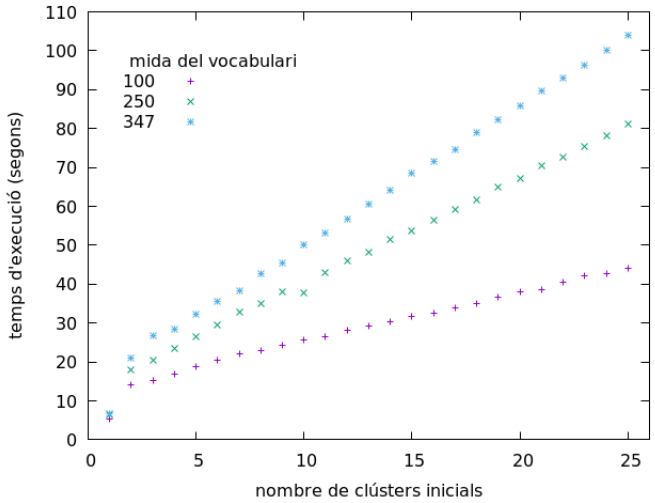

Figura 5: temps d'execució (segons) en funció del nombre de clústers inicials i la mida del vocabulari (amb $m = 0.05$, $M = 0.1$ i 5 iteracions com a màxim).

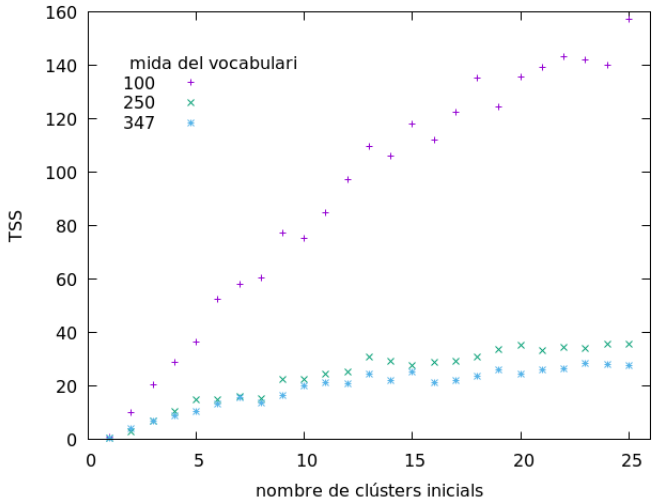

Figura 6: suma dels quadrats en funció del nombre de clústers inicials i la mida del vocabulari (amb $m = 0.05$, $M = 0.1$ i 5 iteracions com a màxim).

4.3 Nombre de processos en paral·lel

Durant l'execució de l'algorisme, podem indicar el nombre de processos en paral·lel (opció `-n` o `--cores`). Analitzem l'efecte del nombre de processos amb les 100, 250 i 347 (totes) paraules més freqüents del vocabulari del conjunt de documents generat amb $m = 0.05$ i $M = 0.1$ i amb 10 clústers inicials.

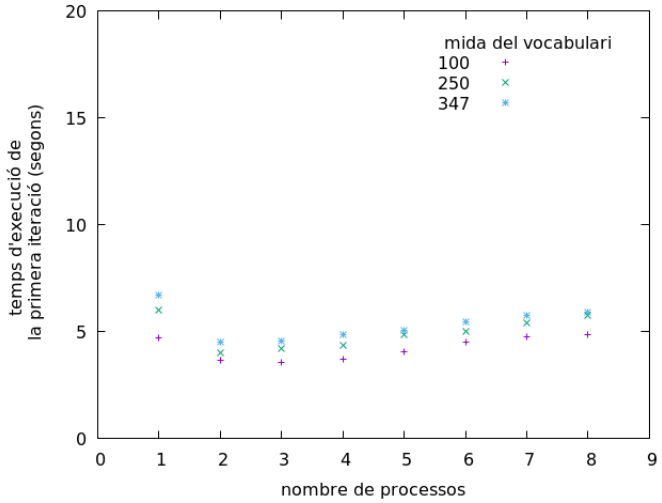

Figura 7: temps d'execució de la primera iteració (segons) en funció del nombre de processos i la mida del vocabulari (amb $m = 0.05$, $M = 0.1$, 5 iteracions com a màxim i 10 clústers inicials).

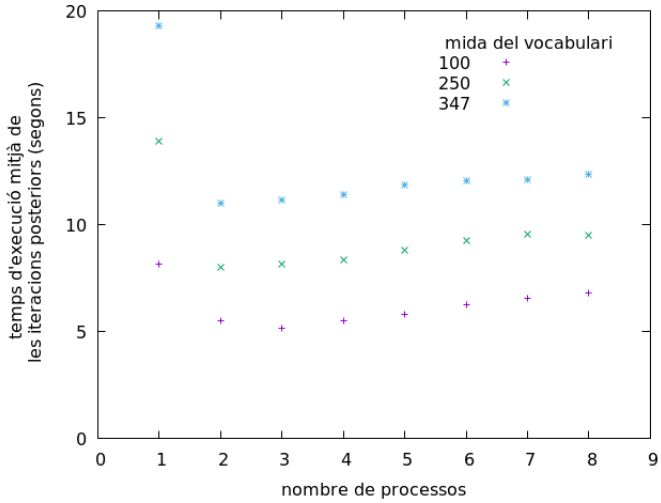

Figura 8: temps d'execució mitjà de les iteracions posteriors (segons) en funció del nombre de processos i la mida del vocabulari (amb $m = 0.05$, $M = 0.1$, 5 iteracions com a màxim i 10 clústers inicials).

El temps d'execució, tant de la primera iteració com de les posteriors, amb 5 iteracions com a màxim, en tots tres casos, quan utilitzem dos processos en lloc de només un procés, es redueix gairebé a la meitat —cal tenir en compte

l'*overhead* de paral·lelització— i augmenta si augmentem el nombre de processos (figures 7 i 8), segurament perquè el processador només té dos nuclis. El temps d'execució de les iteracions posteriors no varia gaire.

4.4 Prototips finals

Analitzem les paraules més freqüents dels prototips resultants de l'execució de l'algorisme amb 20 iteracions, 10 clústers inicials i el conjunt de documents generat amb $m = 0.05$ i $M = 0.1$:

- finit, group, dimens, invari, equival, classic, certain, complet, symmetri, explicit, represent, exact, transform, matrix, satisfi.
- probabl, converg, constant, classic, establish, assumpt, log, matrix, variabl, minim, satisfi, dimens, certain, assum, least.
- communic, user, scheme, channel, cost, access, practic, key, signal, devic, scenario, technolog, maxim, minim, code.
- matter, dark, cosmolog, univers, gravit, background, particl, galaxi, constant, constrain, sigma, futur, signal, redshift, sim.
- electron, magnet, materi, spin, charg, induc, wave, band, metal, particl, optic, symmetri, behavior, reveal, exhibit.
- galaxi, stellar, emiss, gas, sim, spectral, survey, optic, spectra, resolut, veloc, popul, ray, fraction, massiv.
- period, orbit, solar, year, impact, around, after, event, variat, rotat, caus, occur, magnet, frequenc, stellar.
- neural, task, art, machin, research, classif, address, better, input, represent, domain, human, knowledg, world, outperform.
- graph, al, et, edg, log, constant, degre, maximum, best, hard, complet, match, question, input, classic.
- flow, veloc, boundari, particl, forc, finit, diffus, analyt, regim, accur, behavior, stabil, free, spatial, discret.

Podem entreveure de què tracten els documents d'alguns clústers: matemàtiques (dimensió, transformar, matriu), computació (usuari, cost, accedir, dispositiu, codi, neuronal, tasca, màquina, entrada), cosmologia (matèria fosca, cosmologia, univers, gravetat, galàxia), astronomia planetària (orbitar, solar, impactar, rotar) o física (electrònic, magnètic, espín, càrrega, partícula).

Si utilitzem 6 clústers inicials obtenim les següents paraules:

- graph, log, probabl, constant, al, et, edg, maximum, scheme, input, best, communic, maxim, upper, fix.
- finit, classic, converg, dimens, establish, discret, formul, explicit, continu, matrix, certain, satisfi, variabl, complet, equival.
- galaxi, stellar, emiss, sim, survey, gas, veloc, spectral, resolut, optic, popul, spectra, redshift, ray, fit.
- electron, magnet, materi, flow, induc, spin, wave, behavior, regim, exhibit, layer, particl, frequenc, charg, reveal.
- matter, particl, univers, symmetri, constant, cosmolog, dark, perturb, gravit, background, wave, invari, free, light, spectrum.
- task, neural, research, art, machin, address, user, better, practic, cost, world, input, classif, knowledg, domain.

Diferenciem paraules relacionades amb les matemàtiques, l'astronomia, la física, la cosmologia o la computació. Tot i que els prototips inicials són aleatoris els prototips finals s'assemblen força als anteriors.

Cal tenir en compte que el fet que les paraules més freqüents de dos prototips s'assemblin no implica que els documents dels clústers corresponents s'assemblin, perquè els prototips es poden diferenciar per paraules menys freqüents.

Referències

[1] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze. *Introduction to Information Retrieval. Evaluation of Clustering*. Cambridge University Press. 2008.
https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html
(consulta: desembre de 2021).

[2] Towards Data Science. *10 Tips for Choosing the Optimal Number of Clusters*.
https://towardsdatascience.com/10-tips-for-choosing-the-optimal-number-of-clusters-277e93d72d92
(consulta: desembre de 2021).

[3] Wikipedia. *Determining the number of clusters in a data set*.
https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set#Elbow_method
(consulta: desembre de 2021).