Sinem SAV
CS533-Assignment4

Bilkent University

CS533: Information Retrieval Systems

Assignment – 4

Q1) a) C^2MCM is used for cluster maintenance; where addition or deletion to a document database occurs [1]. The algorithm is explained in [1] and it is actually very similar to C^3M; but less costly because it does not need to run C^3M algorithm in each update to the database. Instead, for new comer documents number of clusters and seed powers are calculated, and newcomers are assigned to the seeds that cover them most (the most suitable ones). In case of having documents not covered by seeds, they are put in *ragbag* cluster: newcomers of the next step. Also, in case of deletion of a seed document or a non-seed becoming new seed, the cluster is falsified and the members are also the newcomers of the next step. As there is no need to run overall C^3M algorithm (calculate the C matrix all over again) and only looking at the c_{ii} values, efficiency is gained.

b) The clustering similarity is explained in [1] to validate the newly introduced approach: the similarity of the clusters generated from C^3M and C^2MCM method is compared to see how well the C^2MCM performs.

c) In a data stream environment like news articles, it is logical to use C^2MCM since there will be lots of update on database and maintainability becomes an issue. Also, the proposed method can be used in data stream environment since it will be dynamic and items always come in a certain order: which makes order dependency ignorable (news come in one order). The definition of old would be the items that are not in specific time window t (discard those items).

Q2) a) In the case of addition or deletion in the database, I would change the single-link algorithm in a way that if a new document Is added, it first adds it into similarity matrix to rearrange the pairs in descending order and then assigns to the cluster which it has the most similarity pair value. For example, in the example that we covered in class for single-link (where pairs are AD, AC, BD, BC, AB and AC in descending order), if the document E is added and final similarity matrix gives AE as biggest, then I would put E into the cluster of A and mark the new level as new similarity value (say 0.8 is the value for AE pair). In case of deletion, I would delete the item from cluster and change the label of the clusters by rearranging the similarity matrix again-> similarity values won't change for the remaining ones but the pairs including the deleted items will be removed and if there is label of that pair, it will be removed too and instead, label it with the closest one.

b) I think it is easier to maintain the clusters using k-means, if one item is deleted from the cluster, new mean for that cluster could be calculated to re-run the k-means starting from that step, including only the closest clusters (since if one cluster will be falsified, it is quite likely that the closest clusters will be affected first). but as only one item won't change the means significantly for a large dataset, it would be less costly then running the k-means for overall dataset. Similar scheme can be used for additions, calculate new mean and re-run for the closest clusters.

Q3) CS1= { {a, d}, {b, c, e}, {f, g}}

CS2= {{a, b}, {c, g}, {d, e, f}}

While calculating the Rand's coefficient, we need the TP, TN, FP, and FN counts. As we have 7 elements in total (a,b,c,d,e,f,g), we need to look at pairwise classification of each element and label it as TP, TN, FP or FN which gives $\binom{7}{2} = 21$ pairs to look at.

For Rand's similarity, we need TP+TN count and it won't change from where we look at (when we use either CS1 or CS2 as ground truth) and it obviously gives 11 which all comes as TN from ac, ae, af, ag, bd, bf, bg, cd, cf, dg, eg.

Rand's coefficient = $\frac{TP+TN}{TP+TN+FP+FN}$ = $\frac{11}{21}$

Pair	ab	ac	ad	ae	af	ag	bc	bd	be	bf	bg	cd	ce	cf	cg	de	df	dg	ef	eg	fg
Decision Type	**FP**	**TN**	**FN**	**TN**	**TN**	**TN**	**FN**	**TN**	**FN**	**TN**	**TN**	**TN**	**FN**	**TN**	**FP**	**FP**	**FP**	**TN**	**FP**	**TN**	**FN**

Q4) When CS1 is used as ground truth, the decision table is as follows:

Recall = $\frac{TP}{TP+FN}$ = 0 (no TP) **Precision =** $\frac{TP}{TP+FP}$ = 0 (no TP) **F-measure** = $\frac{2PR}{P+R}$ = 0

Similarly, when CS2 is used, the decision table becomes:

Pair	ab	ac	ad	ae	af	ag	bc	bd	be	bf	bg	cd	ce	cf	cg	de	df	dg	ef	eg	fg
Decision Type	**FN**	**TN**	**FP**	**TN**	**TN**	**TN**	**FP**	**TN**	**FP**	**TN**	**TN**	**TN**	**FP**	**TN**	**FN**	**FN**	**FN**	**TN**	**FN**	**TN**	**FP**

Recall = 0 (no TP) **Precision =** 0 (no TP) **F-measure** = 0

As it can be seen, no matter which cluster structure is used, the final F-measure won't change. This is because of the formulas of recall and precision. When we change the ground truth, TN and TP labels won't change as they stand for "true" classification. However, FP of one scheme will be FN and FN in one will be FP in the other structure. Thus, precision value in the first one will be recall value in the second and recall value in the first will be precision value in the second. Thus, overall F-measure won't change in any case.

Q5) C1= {x, x, x, y} -> **x=3** y=1 z=0

C2= {y, y, y, x) -> x=1 **y=3** z=0

C3= {z, z, z, z, x, y} -> x=1 y=1 **z=4**

Purity = $\frac{3+3+4}{4+4+6}$ = **0.714**

Q6) Min number of pages to be accessed would be 1(best-case) and max number would be 100 (worst-case).

n = number of records = 100*20 = 2000

m = n_c = number of clusters = 100

p = n/m = 20

k = number of relevant records to be accessed = 4

Then, using Yao's formula to find expected number of blocks to be accessed:

= $\sum_{j=1}^{m} E(I_j) = m * [1 - \prod_{i=1}^{k} \frac{n-\frac{n}{m}+1}{n-i+1}]$ = $100 * [1 - \prod_{i=1}^{4} \frac{1981-i}{2001-i}]$ ≅ 3.9 ≅ 4 pages to be accessed.

Q7) As Cardenas suggests in [3] the formula below can be used to estimate the number of blocks or clusters(pages) in our case accessed for a given query:

$$X_D = m\left(1 - \left(1 - \frac{1}{m}\right)^k\right)$$

m=100 and k=4 in our question, which gives:

$$X_D = 100\left(1 - \left(1 - \frac{1}{100}\right)^4\right) \cong 3.94$$

Q8) A= (a, d, b, c)

B= (a, b, e, d)

C= (c, a, e, f)

D= (b, e, g, f)

a) Reciprocal Rank Method

$r(d_i) = \frac{1}{\sum_j 1/\text{position}(d_{ij})}$ formula which is used to find rank position of each document is adopted from [2].

r(a) = $\frac{1}{1+1+\frac{1}{2}} = \frac{2}{5}$ = 0.4

r(b) = $\frac{1}{\frac{1}{3}+\frac{1}{2}+1} = \frac{6}{11}$ = 0.545

r(c) = $\frac{1}{\frac{1}{4}+1} = \frac{4}{5} = 0.8$

r(d) = $\frac{1}{\frac{1}{2}+\frac{1}{4}} = \frac{4}{3} = 1.33$

r(e) = $\frac{1}{\frac{1}{3}+\frac{1}{3}+\frac{1}{2}} = \frac{6}{7}$ = 0.857

r(f) = $\frac{1}{\frac{1}{4}+\frac{1}{4}}$ = 2

r(g) = $\frac{1}{\frac{1}{3}}$ = 3

Thus; **a > b > c > e > d > f > g.**

b) Borda Count Method

In this method, we calculate borda count for each document according to “votes” from each system [2]. As we have 7 documents n=7 and the highest ranked document in each system will get 7 votes where the next one gets 6 and it goes on.

BC(a) = 7+7+6 = 20

BC(b) = 5+6+7 = 18

BC(c) = 4+7 = 11

BC(d) = 6+4 = 10

BC(e) = 5+5+6 = 16

BC(f) = 4+4 =8

BC(g) = 4

Thus; **a > b > e > c > d > f > g.**

Sinem SAV
CS533-Assignment4

c) Condorcet Method

In this method, as described in [2] we look at pairwise win, lose, tie situations for each document and make a final table for each one of them to show total number of win, lose and ties.

In the below table cell[a,b] shows document a (win,lose,tie) respectively over document b.

	a	**b**	**c**	**d**	**e**	**f**	**g**
a	-	(3,1,0)	(2,1,1)	(3,0,1)	(3,1,0)	(3,1,0)	(3,1,0)
b	(1,3,0)	-	(3,1,0)	(2,1,1)	(3,1,0)	(3,1,0)	(3,0,1)
c	(1,2,1)	(1,3,0)	-	(1,2,1)	(2,2,0)	(2,1,1)	(2,1,1)
d	(0,3,1)	(1,2,1)	(2,1,1)	-	(1,3,0)	(2,2,0)	(2,1,1)
e	(1,3,0)	(1,3,0)	(2,2,0)	(3,1,0)	-	(3,0,1)	(3,0,1)
f	(1,3,0)	(1,3,0)	(1,2,1)	(2,2,0)	(0,3,1)	-	(1,1,2)
g	(1,3,0)	(0,3,1)	(1,2,1)	(1,2,1)	(0,3,1)	(1,1,2)	-

Final win, lose, tie table is by looking at the table above:

	Win	**Lose**	**Tie**
a	6	0	0
b	5	1	0
c	2	3	1
d	2	3	1
e	3	2	1
f	0	4	2
g	0	5	1

Thus; **a > b > e > c = d > f > g.**

References

[1] Can, F. (1993). Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems*, 11(2), pp.143-164.

[2] Nuray, R. and Can, F. (2006). Automatic ranking of information retrieval systems using data fusion. *Information Processing & Management*, 42(3), pp.595-614.

[3] Cárdenas, A. (1975). Analysis and performance of inverted data base structures. *Communications of the ACM*, 18(5), pp.253-263.