An Iterative Clustering Method for the XML-Mining Task of the INEX 2010*

Mireya Tovar[1,4], Adrián Cruz[2,4], Blanca Vázquez[3,4], David Pinto[1], Darnes Vilariño[1], and Azucena Montes[4]

[1] Benemérita Universidad Autónoma de Puebla, México
[2] Instituto Tecnológico de Cerro Azul, México
[3] Instituto Tecnológico de Tuxtla Gutiérrez, México
[4] Centro Nacional de Investigación y Desarrollo Tecnológico, México
{mtovar,dpinto,darnes}@cs.buap.mx, abadrector@gmail.com, blanca_tec@hotmail.com, amr@cenidet.edu.mx

Abstract. In this paper we propose two iterative clustering methods for grouping Wikipedia documents of a given huge collection into clusters. The recursive method clusters iteratively subsets of the complete collection. In each iteration, we select representative items for each group, which are then used for the next stage of clustering.

The presented approaches are scalable algorithms which may be used with huge collections that in other way (for instance, using the classic clustering methods) would be computationally expensive of being clustered. The obtained results outperformed the random baseline presented in the INEX 2010 clustering task of the XML-Mining track.

1 Introduction

The INEX 2010 clustering task was presented with the purpose of being an evaluation forum for providing a platform for measuring the performance of clustering methods over a real-world and high-volume Wikipedia collection.

Clustering analysis refers to the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait, often proximity, according to some defined distance measure [1,2,3].

Clustering methods are usually classified with respect to their underlying algorithmic approaches. Hierarchical, iterative (or partitional) and density-based are some possible categories belonging to that taxonomy.

In this paper we report the obtained results when two different approaches for clustering the INEX2010 collection were used. The description of both approaches are given in the following section.

2 Description of the Two Approaches

In order to be able to cluster high volumes of data, we have approached two clustering techniques by partitioning the complete document collection.

* This work has been partially supported by the CONACYT project #106625, VIEP #PIAD-ING11-I, as well as by the PROMEP/103.5/09/4213 grant.

S. Geva et al. (Eds.): INEX 2010, LNCS 6932, pp. 377–382, 2011.

The first approach consider two main modules: the K-biX and the K-biN.

The K-biX clustering method receives as input a similarity matrix sorted in a non-increasing order. Thereafter, it brings together all those items whose similarity value is greater than a given threshold (in this particular case, we have used the average of the similarity matrix). The procedure of K-biX is given in Algorithm 1.

Algorithm 1: Algorithm K-biX used for clustering the INEX 2010

Input: A $n \times n$ similarity matrix $\varphi(d_i, d_j)$ sorted in a non-increasing order, a threshold ϵ

Output: A set of clusters $C_1, C_2, ...$

```
1  D = {d_1, d_2, ...};
2  REG = (|D|^2 − |D|)/2;
3  loop = 1;
4  while (loop ≤ REG) and (φ(d_i, d_j) ≥ ε) do
5      if (|d_i| > |d_j|) then
6          tmp = d_i; d_i = d_j; d_j = tmp;
7      if (d_i ∉ rel and d_j ∉ rel) then
8          fusion_{d_i} = {d_i, d_j};
9          rel_{d_i} = {d_i}; rel_{d_j} = {d_i};
10     else if (d_i ∈ rel) and(d_j ∉ rel) then
11         fusion_{rel_{d_i}} = fusion_{rel_{d_i}} ∪ {d_j};
12         rel_{d_j} = {d_i};
13     else if d_j ∈ rel and (d_i ∉ rel) then
14         fusion_{rel_{d_j}} = fusion_{rel_{d_j}} ∪ {d_i};
15         rel_{d_i} = {d_j};
16     loop = loop + 1;
17 cluster = 1;
18 foreach d_x ∈ fusion do
19     first_d_representative(fusion_{d_x});
20     C_cluster = fusion_{d_x};
21     cluster = cluster + 1;
22 foreach d_x ∈ |D| do
23     if d_x ∉ rel then
24         C_cluster = {d_x};
25         cluster = cluster + 1;
26 return C_1, C_2, ...
```

K-biX is unable to find a fixed number of clusters; instead, it tries to discover the optimal number of clusters. Therefore, we have executed an additional clustering method in order to fix the number of clusters to exactly those required in the competition (50, 100, 200, 500 and 1000).

On the other hand, the K-biN clustering method considers the clustering with a fixed number of clusters. This number depends of criteria given in advance, and

Algorithm 2: Algorithm K-biN used for clustering the INEX 2010

Input: A $n \times n$ similarity matrix $\varphi(d_i, d_j)$ sorted in a non-increasing order, n number of clusters

Output: A set of clusters $C_1, C_2, ...C_n$

```
1  D = {d_1, d_2, ...};
2  REG = (|D|^2 − |D|)/2;
3  gs = n + 1;
4  cn = 0;
5  loop = 1;
6  while (loop ≤ REG) and (gs > n) do
7      if (|d_i| > |d_j|) then
8          tmp = d_i; d_i = d_j; d_j = tmp;
9      if (d_i ∉ rel and d_j ∉ rel) then
10         fusion_{d_i} = {d_i, d_j};
11         rel_{d_i} = {d_i}; rel_{d_j} = {d_i};
12         cn = cn + 2;
13     else if (d_i ∈ rel) and(d_j ∉ rel) then
14         fusion_{rel_{d_i}} = fusion_{rel_{d_i}} ∪ {d_j};
15         rel_{d_j} = {d_i};
16         cn = cn + 1;
17     else if d_j ∈ rel and (d_i ∉ rel) then
18         fusion_{rel_{d_j}} = fusion_{rel_{d_j}} ∪ {d_i};
19         rel_{d_i} = {d_j};
20         cn = cn + 1;
21     if |D| − n ≥ cn then
22         f = 0;
23         cd = 0;
24         foreach d_x ∈ fusion do f = f + 1;
25         foreach d_x ∈ |D| do
26             if d_x ∈ rel then cd = cd + 1;
27         gs = f + cd;
28     loop = loop + 1;
29 cluster = 1;
30 foreach d_x ∈ fusion do
31     first_d_representative(fusion_{d_x});
32     C_cluster = fusion_{d_x};
33     cluster = cluster + 1;
34 foreach d_x ∈ D do
35     if d_x ∉ rel then
36         C_cluster = {d_x};
37         cluster = cluster + 1;
38 return C_1, C_2, ..., C_n
```

Algorithm 3: Algorithm used for clustering the INEX 2010 with K-Means

Input: A corpus D, n number of clusters (50, 100, 200, 500 or 1000)
Output: A set of clusters $C_1, C_2, ...C_n$

Represent each document according to TF-IDF;
Split D into m subsets D_i made of $\frac{|D|}{m}$ documents;
foreach $D_i \subset D$ *such as* $D_i = \{d_{i,1}, d_{i,2}, d_{i,3}, ..., d_{i,\frac{|D|}{m}}\}$ **do**
    Calculate the similirity matrix M_i of D_i by using the cosine measure;
    Apply the K-Means clustering method to M_i in order to obtain k clusters;
    $(\{C_{i,1}, C_{i,2}, ..., C_{i,k}\})$;
end
$Loop = 1$;
while $(Loop \leq MAX\_ITERATIONS)$ **do**
    Select a random representative document $d_{i,j}$ for each cluster $C_{i,j}$ obtained;
    Let D' be the set of documents $d_{i,j}$;
    Calculate the similiraty matrix M'_i of D'_i by using the cosine measure;
    Apply the K-Means clustering method to M'_i in order to obtain n clusters $(\{C'_{i,1}, C'_{i,2}, ..., C'_{i,n}\})$;
    Let $C_{i,j} = C_{i,j} \cup C_{i,j'}$, where $d_i, j \in C'_{i,r}$ and $d_{i,j'} \in C'_{i,r}$ with $1 \leq r \leq k$ and $j <> j'$;
    $Loop = Loop + 1$;
end
return $C_1, C_2, ...C_n$

the similarity degree among the documents processed. The Algorithm 2 presents the K-biN method.

2.1 The K-Means Based Clustering Approach

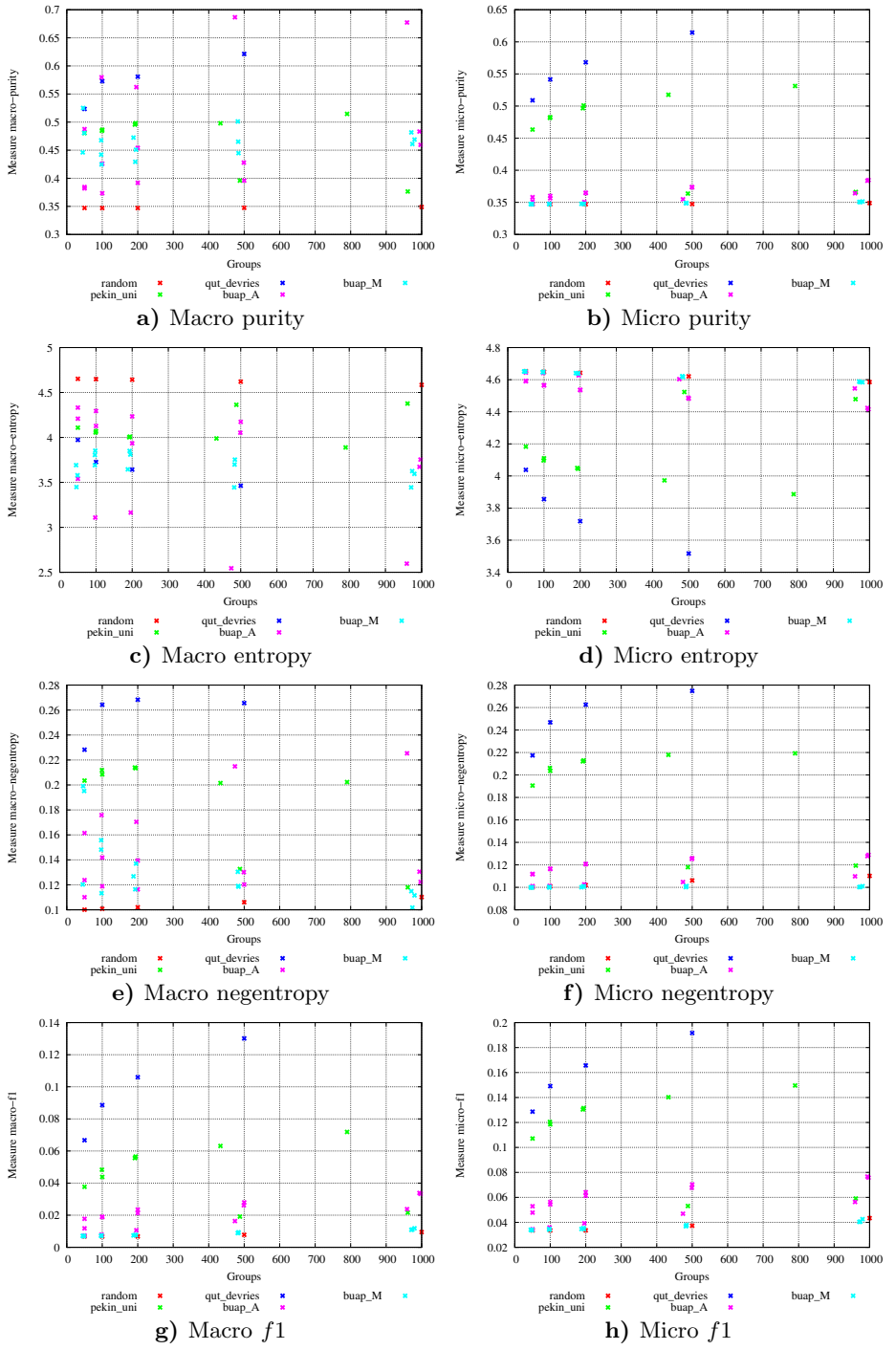
The second approach has used the widely known K-Means algorithm, which assigns each object to the cluster whose center is nearest. The center is the average of all the points of the cluster. The rationale of the K-Means clustering method is shown in ([3]). The main advantage of this algorithm is its simplicity and speed which allows it to run on large datasets. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments. The proposed approach is depicted in Algorithm 3.

The clustering methods aforementioned assume that a matrix that represents the similarity degree among all the documents of the collection is given in advance. For this purpose, we construct this matrix by using the cosine similarity measure over a vectorial representation of each document [4].

The obtained results are presented and discussed in the following section.

3 Experimental Results

The experiments were carried out on the clustering task of the INEX 2010, which evaluated unsupervised machine learning solutions against the ground



a) Macro purity **b)** Micro purity

c) Macro entropy **d)** Micro entropy

e) Macro negentropy **f)** Micro negentropy

g) Macro $f1$ **h)** Micro $f1$

Fig. 1. Evaluations of the runs with Purity, Entropy, negentropy and $f1$ (macro and micro) at the INEX 2010 clustering task

truth categories by using standard evaluation criteria such as Purity, Entropy, Negentropy and F-score ($f1$).

The evaluation is performed by using 146,225 documents that have been preprocessed in order to provide various representations of these documents such as, vector space representation of terms, frequent bi-grams, XML tags, trees, links and named entities. In this paper, we have used unigrams and frequent bigrams (original terms and stemmed terms). The obtained results are shown in Figure 1.

In general, it can be noticed that the presented approaches performs slightly better than the random assignment. Our thought is that we have not sufficiently iterated the algorithm in order to converge to an optimal clustering. We are considering to repeat the experiments with the gold standard in hand in order to analyze these hypotheses.

4 Conclusions

A recursive method based on the K-biX/K-biN and K-Means clustering methods has been proposed in this paper. The aim of the two presented approaches was to allow high scalability of the clustering algorithms. Traditional clustering of huge volumes of data requires to calculate a two dimensional similarity matrix. A process which needs quadratic time complexity with respect to the number of documents. The lower the dimensionality of the similarity matrix, the faster the clustering algorithm will be executed. However, the performance of both approaches were not as expected, because it just slightly improved a baseline made up of a random assignment. We would like to analyze this behavior when the gold standard is released by the task organizers.

References

1. MacKay, D.J.C.: Information Theory, Inference and Learning Algorithms. Cambridge University Press, Cambridge (2003)
2. Mirkin, B.G.: Mathematical Classification and Clustering. Springer, Heidelberg (1996)
3. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297. University of California Press, Berkeley (1967)
4. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)