

# A greedy set cover approach for the recomputation of the cluster centres in the FPAC Algorithm

Ivan Feliciano<sup>1</sup> and Edgar Hernandez-Gonzalez<sup>2</sup>

<sup>1</sup> National Institute of Astrophysics, Optics and Electronics  
ivan.felavel@gmail.com

<sup>2</sup> National Institute of Astrophysics, Optics and Electronics  
edgarmoy.28@gmail.com

**Abstract.** En este trabajo presentamos una modificación al algoritmo K-means usando una heurística que permite hacer el recalcado de centroides de una manera diferente al k-means tradicional. El procedimiento se basa en... primero, después, por ultimo.

**Keywords:** First keyword · Second keyword · Another keyword.

## 1 Introduction

The amount of information, specifically text, that is generated everyday is increasing rapidly and dramatically. This necessitates application of effective and efficient content management techniques to fulfill the task of finding groups of similar documents in a collection of documents. This task is also known as clustering. These document clusters can then be useful for a variety of applications, such as document alignment, information retrieval (IR), text classification, etc [1].

We can define the goal in clustering as follows. Given a set of documents,  $D = \{d_1, \dots, d_N\}$ , a desired number of clusters,  $K$ , and an objective function that evaluates the quality of a clustering, we want to compute an assignment  $\gamma : D \rightarrow \{1, \dots, K\}$  that minimizes (or, in other cases, maximizes) the objective function. In most cases, we also demand that  $\gamma$  is surjective, that is, that none of the  $K$  clusters is empty. The objective function is often defined in terms of similarity or distance between documents.

$K$ -means is perhaps the most widely used clustering algorithm because of its simplicity and efficiency. The objective in  $K$ -means clustering is to minimize the average distance between documents and their centroids or, equivalently, to maximize the similarity between documents and their centroids.

$K$ -means is an EM-based algorithm, in which starting with a set of randomly chosen initial centres, each input point is repeatedly assigned to its nearest cluster centre, the E-step. The cluster centres are then recomputed by making use of the current assignments, the M-step.

The  $K$ -means clustering algorithm does not scale well to datasets that are considerably large in size and large in dimensionality, e.g. large document collections where each document is a sparse vector of large dimension (vocabulary size of the collection). In the case of clustering vectors of large dimensionality, the computational overhead of the  $K$ -means algorithm arises from both the E and the M steps, that is when: (a) assigning vectors to one of the centres (interchangeably referred to as ‘cluster centres’); and (b) recomputing the centres. Concretely speaking, the computation required to assign each vector in the collection to one of the centroids is expensive because it involves computing the similarity of this data point with every cluster centre, which is an expensive operation if the collection is very large.

Since the introduction to the  $K$ -means algorithm, research has progressed towards making it more efficient for cases where the number of data points or the dimensionality of the data is large, e.g. the case when the dataset is comprised of a collection of documents.

The Fast Partitional Clustering Algorithm (FPAC) [1] involves the nearest neighbour based heuristic to scale up  $K$ -means clustering for large document collections. The main contribution of the FPAC on the  $K$ -means algorithm is the use of an operation, namely  $TOP(x)$  which gives a list of the top most similar vectors with respect to the current vector. This operation is used in the three fundamental steps of the  $K$ -mean, the selection of the initial cluster centres, the assignment of a cluster to a non-centroid vector  $d$ , and the recomputation of the centre clusters.

Our work is based on the FPAC algorithm and we have focused in the recomputation of the centroids step. In the FPAC algorithm they select as the cluster centroid the vector with the highest number of distinct terms. Despite the efficiency of just take the vector with the maximum number of unique terms, what happen if the set of terms in the vector is disjoint from the other set of terms in the cluster vocabulary. That case inspired us to develop a solution based on the set cover problem.

Our contribution is the extension of the method for the recomputation of the cluster centroids. Instead of using just one centroid that tries to cover most of the vocabulary within a cluster, we proposed the selection of  $M$  centroids such that they cover the maximum number of terms in a cluster vocabulary.

## 2 Related Work

The Fast PARTitional Clustering (FPAC) algorithm makes use of the nearest neighbour based heuristic. This heuristic consists in using a set of the most similar vector from a each cluster vector  $x$ . The operation of getting the set of the top nearest vectors is called  $TOP(x)$  and is efficiently computed by using a inverted index data structure.

In a very general way we can highlight three fundamental steps in the FPAC algorithm: the selection of the initial centroids, the assignment of a cluster to a non-centroid vector and the recomputation of the cluster centres.

To ensure that the initially selected cluster centres are dissimilar to each other, they select the first cluster centre,  $C_1$ , randomly from the whole collection. The next cluster centroid,  $C_2$ , is then selected such that it has a low similarity with the already chosen centroid. More precisely,  $C_2$  it is a randomly chosen vector from the collection that is composed by the vectors that do not occur in the retrieved list of  $C_1$ ,  $TOP(C_1)$ . For selecting  $C_3$ , a vector is chosen at random that does not occur in the union of the retrieved lists for  $C_1$  and  $C_2$ , and the process continues till  $K$  cluster centres are obtained.

To assign a cluster to a non-centroid vector  $d$ , they use the  $K$  centroids as queries. If the cluster centres themselves are dissimilar to each other, it is expected that the ranked lists retrieved for each centre will have a small intersection. If a vector  $d$  it is included it is retrieved in the top set of only a single ranked list,  $TOP(C_k)$ , it is included in the set of vector assigned to the cluster corresponding to  $C_k$ . If a vector  $d$  is retrieved in multiple ranked lists, it is assigned to the cluster for which the normalized similarity score is maximum. The vectors which are not retrieved in the union set of the ranked lists for each centre are assigned randomly to any one of the  $K$  cluster centroids.

For computing the new centroid for a cluster they treat the vector which has the maximum number of unique terms within a cluster (i.e. the one with maximum length) as the updated cluster centre.

The results they obtained from the experiments of the FPAC algorithm compared to the K-means and SK-means algorithms, show an increase in the execution speed attributed to the IR-based approach and the two additional heuristics related to initial centroid selection and centroid recomputation.

The FPAC similarity-based initial centroids selection produces the best results in terms of clustering effectiveness.

### 3 Proposed Solution

- Describir en el paso en el algoritmo en que nos concentramos
- Describir la adaptación que se hizo a los demás pasos para que utilizara varios centroides, `initCentroids()` y `getClosestCluster()`
- Tal vez sea bueno escribir un poco sobre el problema de ser cover y la aproximación greedy
- introducir un poco de lo que se habla en la siguiente subsección

#### 3.1 Recomputation of the cluster centres

- Describir el procedimiento
- Tal vez la complejidad y desventajas

## 4 Experiments

### 4.1 Dataset

qué conjuntos de datos utilizamos

- Por qué no usamos el TREC Microblog Dataset
- 20 news, descripción del dataset
- sentiment140 tweets
- gender tweets

#### 4.2 Clustering evaluation

- Medidas que utilizamos para evaluar el clustering y hablar un poco de las clases que hay de cada conjunto que se utilizó
- Purity
- NMI
- RI

#### 4.3 Implementation

Decir que utilizamos como base lo desarrollado por Ganguly usando Lucene

#### 4.4 Compared approaches

Decir que comparamos esta versión del FPAC M centroids con K means normal, SKMeans y FPAC.

#### 4.5 Parameters

- El valor de K
- el número de iteraciones
- El valor de M

### 5 Results

- Por cada dataset una gráfica del NMI, Purity y RI
- 

### 6 Conclusions and future work

#### References

1. Ganguly, D.: A fast partitional clustering algorithm based on nearest neighbours heuristics. Pattern Recognition Letters **112**, 198–204 (2018). <https://doi.org/10.1016/j.patrec.2018.07.017>
2. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. Cambridge University Press (2009)