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

Ivan Feliciano[1] and Edgar Hernandez-Gonzalez[2]

[1] National Institute of Astrophysics, Optics and Electronics
ivan.felavel@gmail.com
[2] 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 recalculo 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 $D = \{d_1, \ldots, d_N\}$, a set of documents, K, a desired number of clusters and an objective objective function that evaluates the quality of a clustering. we want to compute an assignment $\gamma : D \to \{1, \ldots, K\}$ that minimizes (or, in other cases, maximizes) the objective function. In most cases, we also demand that γ 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 cen- tre, 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 mos 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

- Describir los intentos hasta lo que hizo el autor
- Describir el algoritmo del autor
- Describir el problema con el algoritmo del autor

Se han hecho varias aportaciones para mejorar el algoritmo k-means. [la del articulo] desarrollo un algoritmo de partición rápida basado en una heurística de los vecinos mas cercanos. Dado un conjunto de centroides, evitar el calculo de distancia por pares entre vectores para obtener una partición de la colección,

en su lugar se ocupó una asignación basada en el vecino mas cercano de cada centro, para esto se utilizo una lista invertida de vectores dispersos. También se evito el costoso calculo del centroide verdadero de cada grupo, se propuso una heurística para elegir el centroide de manera eficiente. En [9] los autores utilizan la heurística más lejana primero que consiste en seleccionar los centroides iniciales y evitar los cálculos de distancia redundante desde los no centroides a los centros. En [12, 19 y 20] se utilizaron estructuras de datos de partición de espacio como kd-trees. Esto aumenta la eficacia de k-means, pero solo para pocas dimensiones En [23] reasignaciones de clúster ocurren frecuentemente para puntos que no están cerca de los centroides. Identifica estos puntos al agrupar puntos vecinos usando múltiples arboles de partición. En [2 y 18] se utiliza una heurística la cual elije aleatoriamente el primer centroide y utiliza una distribución de probabilidad. En [4] se utilizo k-means escalable en el cual cada centroide se ve como una consulta para recuperar una lista de documentos que luego se asignan a ese grupo sin cálculo de distancia. [2] k-means ++.[2]

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)