Alexandria Engineering Journal (2015) **54**, 1147–1154

HOSTED BY

ELSEVIER

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

ORIGINAL ARTICLE

Efficient incremental density-based algorithm for clustering large datasets

Ahmad M. Bakr [*], Nagia M. Ghanem, Mohamed A. Ismail

Faculty of Engineering, Computer and Systems Engineering Department, Alexandria University, Alexandria, Egypt

Received 4 November 2014; accepted 20 August 2015
Available online 12 October 2015

KEYWORDS

Incremental clustering;
Density-based clustering;
Document clustering;
Information retrieval

Abstract In dynamic information environments such as the web, the amount of information is rapidly increasing. Thus, the need to organize such information in an efficient manner is more important than ever. With such dynamic nature, incremental clustering algorithms are always preferred compared to traditional static algorithms. In this paper, an enhanced version of the incremental DBSCAN algorithm is introduced for incrementally building and updating arbitrary shaped clusters in large datasets. The proposed algorithm enhances the incremental clustering process by limiting the search space to partitions rather than the whole dataset which results in significant improvements in the performance compared to relevant incremental clustering algorithms. Experimental results with datasets of different sizes and dimensions show that the proposed algorithm speeds up the incremental clustering process by factor up to 3.2 compared to existing incremental algorithms.

1. Introduction

Data clustering [1] is a discovery process that groups set of objects in disjoint clusters such that the intra-cluster similarity is maximized and inter-cluster similarity is minimized. The resulted clusters can explain characteristics of the underlying data distribution which can be used as a foundation for other data mining and analysis techniques. Data clustering is used in a wide range of applications. For example, in marketing, clustering is used to find group of customers with similar behaviors [2]. In biology, it is used to find similar plants or animals given their features [3]. In search engines, clustering is used to group similar documents to facilitate user search and topics discovery [4], while in networks, clustering is used in analyzing and classifications of network traffic [5]. Given the number of clustering applications, there are challenges associated with the clustering process. Such challenges include the choice of a suitable clustering algorithm, the choice of best representative features of objects and the choice of appropriate distance/similarity measure [6]. Other challenges include dealing with outliers [7], ability to interpret clustering results (selection of cluster's representative and cluster summarization) [8] and dealing with huge number of dimensions and distributed data [9].

Due to data overwhelming, new challenges have appeared for traditional algorithms that were implemented before. One challenge is the ability to perform incremental update of the

[*] Corresponding author. Tel.: +20 1004102280.
E-mail addresses: ahmad.bakr@alexu.edu.eg (A.M. Bakr), nagia.ghanem@alexu.edu.eg (N.M. Ghanem), maismail@alexu.edu.eg (M.A. Ismail).
Peer review under responsibility of Faculty of Engineering, Alexandria University.

http://dx.doi.org/10.1016/j.aej.2015.08.009
1110-0168 © 2015 Faculty of Engineering, Alexandria University. Production and hosting by Elsevier B.V.

clusters upon any change in the dataset. Traditional algorithms require existence of the whole dataset before running the algorithm. However, in many online applications where the time factor is essential, traditional algorithms are not feasible. As a result, algorithms that can perform incremental updates to the clusters are always preferred in dynamic environments. In such algorithms, objects are processed one at a time and incrementally assigned to their prospective clusters while they progress with minimized overload. Along with incremental process, incremental clustering algorithms face other challenges such as finding the best way to determine the most suitable cluster that next object will be assigned, and also once an object is assigned to a cluster, this assignment may change in the future as new objects are added or removed from the dataset (which is also known as The Insertion Order Problem [10]).

In this paper, an incremental density-based clustering algorithm is introduced for incrementally building and updating clusters in the dataset. The proposed algorithm enhances the clustering process by incrementally partitioning the dataset to reduce the search space of the neighborhood to one partition rather than the whole dataset. For each partition, the proposed algorithm maintains an incremental DBSCAN algorithm to detect and update dense regions at the partition with new objects. Finally, to identify the final natural number of final clusters, the algorithm employs a merging step to merge dense regions in different partitions. Experimental results show that the proposed algorithm speeds up the incremental clustering process with a factor up to 3.2 compared to relevant existing incremental clustering algorithms. The main contribution of this work can be summarized in enhancing the incremental DBSCAN algorithm by limiting the search space to partitions instead of the whole dataset which speeds up the clustering process. The proposed algorithm is proved to perform better than the existing incremental DBSCAN especially in large datasets with higher number of dimensions.

The rest of the paper is organized as follow: Section 2 discusses related work; Section 3 presents the proposed algorithm in detail; Section 4 presents the experimental results with other algorithms as well as the evaluation metrics and the datasets; finally Section 5 concludes a summary of the proposed work with potential future work.

2. Related work

Clustering is unsupervised classification of data into groups or clusters. Existing clustering algorithms can be classified into different classes. Centroid-based algorithms such as K-means [11], PAM [12], BIRCH [13], and CLARANS [14] are simple and fast to converge to local optimum; however they have limitations of predefining the number of clusters and dealing with clusters of different sizes and shapes. Hierarchical-based algorithms include single linkage, CURE [15] and Chameleon [16] can deal with different shapes and sizes of clusters and less susceptible to initialization and outliers; however they suffer from two main limitations: firstly they can never undo what were done (e.g. whenever two clusters are merged, they will always be merged) and secondly their high complexity which makes them slow to converge. Density-based algorithms such as DBSCAN [17], SSN [18] and OPTICS [19] overcome the difficulty of detecting arbitrary shaped clusters by extracting high dense regions (clusters) separated by low dense regions. Density-based algorithms can detect noise objects as they are not reachable from any of the generated clusters and hence noise has less effect of density based algorithms.

In dynamic environments where data are changing frequently overtime, clustering algorithms are required to perform the necessary updates incrementally in an efficient manner. Many incremental clustering algorithms were developed to cope with the dynamic nature of the data. Tzortzis and Likas [20] proposed a kernel K-means as an extension of the standard K-means that incrementally identifies nonlinearly separable clusters based on kernel based clustering. Widyantoro and Ioerger [21] proposed an incremental hierarchical clustering algorithm that works in a bottom-up fashion such that a new instance is placed in the hierarchy and a sequence of hierarchy restructuring is performed only in regions that have been affected by the presence of the new instance. Mary and Kumar [22] proposed an incremental density-based algorithm that can detect clusters with arbitrary shapes and handle updates in an incremental manner. Hammouda and Kamel [23] proposed an incremental similarity-histogram based algorithm where the similarity histogram is a concise statistical representation of the set of pair-wise document similarities distribution in the cluster and objects are added to a cluster only if they enhance its similarity histogram (keeping the distribution of similarities skewed to the right of the histogram). As the incremental DBSCAN algorithm [24] is used at one of the stages of the proposed algorithm, more details are introduced in the next subsection.

2.1. Incremental DBSCAN algorithm

Incremental DBSCAN algorithm is density-based clustering algorithm that can detect arbitrary shaped clusters. While static DBSCAN [17] is applied to static datasets in which the existence of all objects is required before running the algorithm, incremental DBSCAN [24] works by processing objects as they come and update/create clusters as needed. Given two parameters Eps and Minpts:

Definition 1. The Eps neighborhood of an object p is donated by $N_{\mathrm{Eps}}(p)$, is defined by $N_{\mathrm{Eps}}(p) = \{q|\mathrm{dis}(p,q) \leqslant \mathrm{Eps}\}$.

Definition 2. An object p is directly density-reachable from object q if $p \in N_{\mathrm{Eps}}(q)$ and $|N_{\mathrm{Eps}}(q)| \geqslant$ Minpts (i.e. q is a core object).

Definition 3. An object p is density-reachable from an object q if there is a chain of objects $p_1,p_2,\ldots,p_n$ such that p_{i+1} is directly density reachable from p_i and $p_1 = p$ and $p_n = q$.

Due to the density-based nature of the incremental DBSCAN algorithm, the insertion or deletion of an object affects only the objects within a certain neighborhood (Fig. 1). Affected objects are potentially the objects that may change their cluster membership after insertion/deletion of an object p and they are defined as the objects in the $N_{\mathrm{Eps}}(p)$ plus all other objects that are density reachable from objects in $N_{\mathrm{Eps}}(p)$. The cluster memberships of all other objects are not affected.

Assuming that D is the objects space, following the insertion of an object p, defines: