Neurocomputing 126 (2014) 141–150

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Online fuzzy medoid based clustering algorithms

Nicolas Labroche [*]

UPMC Univ Paris 06, UMR 7606, LIP6, 4 place Jussieu, 75005 Paris, France

ARTICLE INFO

Article history:
Received 3 January 2012
Received in revised form
19 April 2012
Accepted 29 July 2012
Available online 11 June 2013

Keywords:
Fuzzy clustering
Online
Medoids
Stream
Large dataset

ABSTRACT

This paper describes two new online fuzzy clustering algorithms based on medoids. These algorithms have been developed to deal with either very large datasets that do not fit in main memory or data streams in which data are produced continuously. The innovative aspect of our approach is the combination of fuzzy methods, which are well adapted to outliers and overlapping clusters, with medoids and the introduction of a decay mechanism to adapt more effectively to changes over time in the data streams. The use of medoids instead of means allows to deal with non-numerical data (e.g. sequences…) and improves the interpretability of the cluster centers. Experiments conducted on artificial and real datasets show that our new algorithms are competitive with state-of-the-art clustering algorithms in terms of purity of the partition, F1 score and computation times. Finally, experiments conducted on artificial data streams show the benefit of our decay mechanism in the case of evolving distributions.

1. Introduction

Online learning aims at adapting a model to the continuous fluctuations of the input data, and has been studied in numerous research fields from classification, clustering, system identification to time series prediction [1]. More specifically, online clustering aims at discovering meaningful clusters from a stream of input data. Proposed algorithms have to adapt to limited input data fluctuations (drifts), more abrupt changes in the data distribution (shifts) or clusters fusion or separation. Although several clustering algorithms have already been proposed, they are generally limited to numerical data to simplify the online cluster representation.

To this aim, this paper proposes an extension of the classical fuzzy clustering algorithm [2] in three main areas: (1) our model is designed to deal with nowadays very large datasets, (2) the proposed method incorporates a weighting mechanism to adapt to evolving distribution in data streams and, (3) our algorithm can handle any data types (numerical vectors, sequences …) as the cluster centers are defined as medoids, which are representative points from the dataset. In our model, a very large dataset (or a stream) is considered as a sequence of small data chunks that can be processed in main memory as in [3,4]. Then, two fuzzy clustering algorithms are proposed to aggregate the knowledge extracted from each data chunk and to produce the final cluster centers.

Some preliminary works have been already proposed in [5], but this paper introduces new mechanisms to process online data streams. Our new improved algorithms are compared to state-of-the-art fuzzy clustering algorithms on large artificial and real datasets. Experiments show that our new approaches perform closely if not better than existing clustering algorithms while adding the capability to handle relational data. Finally, experiments conducted on artificial data streams show that our new model can adapt to evolving data distribution over time contrary to existing methods.

This paper is organized as follows: Section 2 describes related works on large scale clustering and stream clustering algorithms. Section 3 introduces the weighted c-medoids algorithm which processes the data chunks and on which rely our new online fuzzy clustering models. Section 4 proposes a comparative evaluation of our new fuzzy approaches against well-known clustering algorithms on artificial and real datasets and a validation of our new algorithms on artificial data streams. Finally Section 5 concludes and discusses the perspectives of this work.

2. Related works

This section presents related research works either in the field of clustering of (very) large data sets or in the domain of stream clustering. In the second case, a distinction is proposed between, on the one hand, fuzzy neural network algorithms or, on the other hand, extensions of classical clustering models to the particular problem of stream analysis.

2.1. Clustering of large data sets

According to a study reported in [6], clustering algorithms based on centers such as k-means [7] are among the most widely used algorithms in data mining. Fuzzy variant introduced in [2]

[*] Tel.: +33 1 44 27 88 08; fax: +33 1 44 27 70 00.
E-mail address: nicolas.labroche@lip6.fr

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.07.057

improves the original model by introducing for each point a membership value to each cluster. This method is more robust to outliers than traditional crisp k-means and can deal with overlapping clusters. However, like other well-known clustering methods such as hierarchical methods [8], these methods cannot handle very large data sets. Thus, several methods have been proposed to overcome this problem. The algorithm BIRCH [9,10] is a hierarchical method that can deal with large datasets, but can only process numerical data, because of the cluster feature tree structure that is used to summarize and index the data in memory. In the particular case of clustering algorithms based on centers, several studies have been proposed to speed up the processing by providing better center initializations than the random method that is generally used [11–16].

Although these improved methods may allow a faster and a better convergence, they cannot handle the size of currently available datasets or their complexity as they are limited to numerical data processing. To overcome this limitation, relational clustering algorithms are based exclusively on the expression of the relation between data objects as a (dis)similarity matrix. In [17–19] the authors derive c-means like approaches (crisp, fuzzy or possibilistic) into relational methods. To achieve this, the methods used to update the centers after each iteration and to compute the distance between points are replaced so that they only depend on the assignment of points to the clusters and the values of the (dis)similarity matrix. The method NERF [19] (Non-Euclidian Relational Fuzzy c means clustering) proposes an extension of these methods in the case where the (dis)similarity matrix is not Euclidean. In [20] the authors propose to construct a sequence of nested samples to find the smallest sample of the dataset that guarantees a certain quality. Once the sample is built, the relational algorithm NERF is applied to the sample to learn the cluster centers. The result is then extended to the whole dataset, which allows the new algorithm to handle extremely large datasets. However, this approach cannot be applied to online data streams processing as the data changes over time do not guarantee the representativeness of the sample used to build the clustering model.

Other methods define the cluster centers as medoids (i.e. points that belong to the dataset) rather than means to allow the processing of all types of data. These methods are more resistant to noise and allow a better interpretation of results. These include algorithms PAM (Partitioning Around medoids), CLARA and CLARANS [21,22]. PAM builds the partition iteratively by exchanging a medoids with a point that is not a center to improve the quality of the partition at each iteration. To reduce the quadratic complexity of PAM, CLARA and CLARANS use sampling methods: CLARA randomly constructs multiple samples and applies PAM on each to build the final cluster centers, while CLARANS only consider a sample of the dataset at each iteration to replace an existing medoid. In [23], the authors use a structure called Slim Tree, similar to a R-Tree or a M-Tree, but adapted to relational data in order to achieve an intelligent sampling of the dataset. As with other sampling methods, this structure is not usable in our context, as all data may not be available at the beginning of the analysis. Other works propose efficient c-medoids algorithms with sub-quadratic complexity as in [24]. Finally, in [25,26], the authors describe an efficient fuzzy c-medoids algorithm with linear complexity, which considers only the points with the highest membership to a cluster as medoid candidates for this cluster. This algorithm performs well but is limited to datasets that can be stored in memory.

2.2. Data stream analysis

Stream mining refers to machine learning algorithms that are specifically designed to process data streams. In this case, the difficulty is not simply to handle very large (or possibly) infinite data sets but to adapt continuously and automatically to the evolutions of the data streams. These evolutions may be smooth moves of the classes in their definition space, also called drifts, or can be more abrupt changes known as shifts. In this paper, we are interested in clustering algorithms that identify models of similar input data over time. This problem has been tackled in many ways in the literature.

Fuzzy neural networks methods are interested in the more general problem of identification of an evolving non-linear system in which the structure and the parameters have ideally to be learned from the online data stream. In this context, two main families of methods have been proposed: either fuzzy rule based system such as [1,27–29] or evolving neural networks [30].

For example, in [1], the author proposes the simpl_eTS+ algorithm that improves previous method presented in [27]. In this framework, the solution is defined as a set of fuzzily connected local models and relies on a clustering algorithm named Simpl_e_Clustering to learn automatically the structure of the system. This online clustering algorithm adapts to the shifts in the input data thanks to a measure of the density derived from the relative position to the mean of the overall data. Finally, and contrary to existing approaches that include rules or neurons pruning mechanisms [31,32,28], simpl_eTS+ integrates two mechanisms to first, select the fuzzy rules based on their utility, and second, select the input attributes. Provided experiments show that the proposed method outperforms the simpl_eTS approach [27]. However, even if simpl_eTS+ is described as being based on prototypes to describe clusters (or fuzzy rules), the method computes a global mean to evaluate the density increment and is thus limited to numerical data streams.

Stream mining has also been studied regarding more specifically the clustering problem. A state of the art can be found in [33] and an overview on scalable fuzzy approaches can be found in [34]. In [3] the authors introduce a streaming algorithm that uses a k-Medoid algorithm called Localsearch. In this approach, the continuous data stream is viewed as a set of data chunks that can be partitioned and represented by their weighted centers. At the end, the set of all weighted centers obtained from all data chunks is clustered with Localsearch to produce the final partition. In [4], the authors introduce an online clustering algorithm that uses an incremental model based on previous centers history to analyze the new chunks and build a fuzzy partition of the dataset. Comparative experiments between the algorithm proposed by [3] and the online fuzzy c-means proposed by [4] show that the second obtains better results on benchmark datasets. However this approach is limited to numerical data.

In [35], the authors introduce the Clustream algorithm that process data streams in two phases. The online phase summarizes the data streams as a set of micro-clusters while the online phase produces the clustering of the micro-clusters. A micro-cluster is represented as a temporal extension of the cluster feature formalism first introduced by [9]. In Clustream micro-clusters are store in a pyramidal time frame structure to mine efficiently the clusters at different time intervals and scales. Similarly, [36] uses a ClusTree structure to maintain stream summaries that adapt efficiently to the speed of the input data. More recently, the authors of Clustream [35] have introduced the HPStream clustering algorithm [37] that improves Clustream by introducing a fading concept and a projection mechanism to handle more efficiently highly dimensional data. In [38] the authors propose the HSWStream algorithm that improves HPStream to deal with sliding windows based on an efficient representation of data points distribution named exponential histogram.

Finally, some works [39,40] describe density-based stream clustering algorithm to deal with arbitrary shaped clusters and