Applied Soft Computing 23 (2014) 61–75

Contents lists available at ScienceDirect

Applied Soft Computing

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

Herd Clustering: A synergistic data clustering approach using collective intelligence

Ka-Chun Wong[a,b,*], Chengbin Peng[c], Yue Li[a,b], Tak-Ming Chan[d]

[a] Department of Computer Science, University of Toronto, Toronto, Ontario, Canada
[b] Terrence Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Toronto, Ontario, Canada
[c] CEMSE Division, King Abdullah University of Science and Technology, Thuwal, Jeddah, K.S.A.
[d] Department of Integrative Biology and Physiology, University of California Los Angeles, Los Angeles, California, U.S.A.

ARTICLE INFO

Article history:
Received 27 September 2013
Received in revised form 13 April 2014
Accepted 22 May 2014
Available online 13 June 2014

Keywords:
Heuristic
Natural computing
Herd behavior
Collective intelligence

ABSTRACT

Traditional data mining methods emphasize on analytical abilities to decipher data, assuming that data are static during a mining process. We challenge this assumption, arguing that we can improve the analysis by vitalizing data. In this paper, this principle is used to develop a new clustering algorithm.

Inspired by herd behavior, the clustering method is a synergistic approach using collective intelligence called Herd Clustering (HC). The novel part is laid in its first stage where data instances are represented by moving particles. Particles attract each other locally and form clusters by themselves as shown in the case studies reported. To demonstrate its effectiveness, the performance of HC is compared to other state-of-the art clustering methods on more than thirty datasets using four performance metrics. An application for DNA motif discovery is also conducted. The results support the effectiveness of HC and thus the underlying philosophy.

1. Introduction

Nowadays, with the support of science and technology, large amounts of data have been, and will continue to be, accumulated. For example, a single human genome accounts for about four gigabytes data space [62,63,65] and the transaction logs in financial markets are measured in billions each day [17]. Such a large amount of data is overwhelming and prevents us from applying traditional analysis techniques. Large-scale methods need to be devised to handle it. As one of the main analysis tools, cluster analysis methods have been proposed to separate the large amount of data into clusters. The data clustering methods are unsupervised which means there is not any label for model training; we do not even know the exact number of clusters beforehand. Given a set of data, a clustering method is expected to divide the data into several clusters by itself. Formally speaking, given a set of data instances, a data clustering method is expected to divide the set of data instances into the subsets which maximize the intra-subset similarity and inter-subset dissimilarity, where a similarity measure is defined beforehand.

To tackle the problem, we propose and describe a novel clustering method in this study. It novelties lie in two aspects: (1) The proposed method is inspired from the nature, herd behavior, which is a commonly seen phenomenon in the real world including human mobility patterns [46]. Thus it is very intuitive and easy to be understood for its good performance. (2) The proposed method demonstrates that cluster analysis can be done in a non-traditional way by making data *alive*. We have also applied the proposed method to DNA motif discovery, demonstrating its real-world applicability. In addition, this study gives a comprehensive and fair comparison for different methods because each method has been well-tuned for each dataset tested fairly.

The rest of paper is organized as follows: First, the related works are reviewed and discussed. After that, the new clustering algorithm is described in detail in the subsequent sections. For the sake of clarity, case studies are implemented and used to help us grasp the intuition of the proposed method. To observe its overall performance, it is compared with the other methods on more than thirty datasets. An application on DNA motif discovery is reported. Based on the results, we discussed the pros and cons of the method proposed at the end.

2. Background

Since most data clustering problems have been shown to be NP-hard [26], many methods have been proposed in the

[*] Corresponding author at: Department of Computer Science, University of Toronto, Toronto, Ontario, Canada. Tel.: +1 416 978 6025.
E-mail address: wkc@cs.toronto.edu (K.-C. Wong).

http://dx.doi.org/10.1016/j.asoc.2014.05.034
1568-4946/© 2014 Elsevier B.V. All rights reserved.

past. In general, those methods can be categorized into different paradigms: partitional clustering, hierarchical clustering, density-based clustering, grid-based clustering, correlation clustering, spectral clustering, gravitational clustering, and others.

The most well-known clustering method should be k-means [53] method. It belongs to partitioning clustering paradigm in which data are divided into non-overlapping subsets iteratively. A variant called k-means++ [2] has also been proposed to improve the k-means seeding stage. In contrast, clusters are formed by either a bottom-up approach or a top-down approach in hierarchical clustering paradigm. For example, single-linkage clustering [68] is a classic bottom-up approach in which data points are gradually agglomerated together to form clusters. To model data dynamically, a special hierarchical clustering method called Chameleon has also been proposed [37]. It makes use of the inter-connectivity and closeness concept to merge and divide clusters.

Apart from the well-known clustering methods, there are different clustering paradigms. In density-based clustering, data is clustered based on some connectivity and density functions. For example, DBscan [19] uses density-based notions to define clusters. Two connectivity functions *density-reachable* and *density-connected* have been proposed to define each data point as either a core point or a border point. DBscan visits points arbitrarily until all points have been visited. In grid-based clustering, data space is divided into multiple portions (grids) at different granularity levels to be clustered individually. For example, CLIQUE [1] can automatically find subspaces with high density clusters. No data distribution assumption has been made. Correlation clustering [4] was motivated from a document clustering problem in which one has a pair-wise similarity function *f* learned from past data. The goal is to partition the current set of documents in a way that correlates with *f* as much as possible. In contrast, spectral clustering [40,45,50] is a relatively promising approach for clustering based on the leading eigenvectors of the matrix derived from a distance matrix. The main idea is to make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction for k-means clustering in fewer dimensions. The seminal work [45] is implemented and compared in this work.

Distinct from the works we have mentioned, gravitational clustering is considered as a rather unique method. It was first proposed by Wright [66]. In the method, each data instance is considered as a particle within feature space. A physical model is applied to simulate the movements of the particles. As described in [25], Jonatan et al. proposed a new gravitational clustering method using Newton laws of motion. A simplified version of gravitational clustering was proposed by Long et al. [39]. Wang et al. proposed a local shrinking method to move data toward the medians of their k nearest neighbors [60]. Blekas and Lagaris [11] proposed a similar method called Newtonian Clustering in which Newton's equations of motion are applied to shrink and separate data, followed by an Expectation Maximization (EM) algorithm for building a Gaussian mixture model. Molecular dynamics-like mechanism was also applied for clustering by Junlin et al. [36].

There are lots of other clustering methods proposed in the past. For instance, Maulik et al. applied a genetic algorithm to search for cluster centers [43]. A globally incremental approach to k-means has been reported in [38]. Celeux et al. have proposed a novel method called Gaussian parsimonious clustering models [12]. Different distance measures have been incorporated into an objective function to cluster arbitrary number of clusters [23]. A hierarchical agglomerative clustering methodology using symbolic objects has been described in [27]. Tsao et al. used a fuzzy Kohonen network for clustering [56]. A fuzzy c-means algorithm has been developed as described in [67,73]. An alternative pruning approach to reduce the noise effect has also been proposed for the fuzzy c-means algorithm [70]. In recent years, several kernel methods have been developed for clustering [20]. A fuzzy-rough set application to microarray data has also been reported in [41]. Hu et al. have applied a hierarchical clustering method for active learning [35]. Interestingly, Corsini et al. have trained a neural network to define dissimilarity measures which are subsequently used in the relational clustering [14]. Gullo et al. have also proposed clustering methods on uncertain data [29–31]. There are many other works; comprehensive survey can be found in [5,64,68].

3. Proposed method

3.1. Main idea

In this work, a clustering method called Herd Clustering (HC) is proposed. This method differs from the traditional ones. Instead of trying hard to analyze data alone, it also spends effort on moving data. Two stages are proposed in HC. Inspired by the herd behavior [3], an attraction model is used to guide data movements in the first stage. A new clustering approach is then taken to cluster data in the second stage. At the first glance, HC is similar to Gravitation Clustering (GC) [66]: data instances are moved according to a model. Nonetheless, their details are totally different. For instance, the model in GC is a physical model following Newton Laws of motion, while that in HC is an artificial model which is designed from the empirical observations in clustering efficiency. The particle acceleration decreases as the inter-particle distance increases in GC while they are independent in HC. Calculus is involved in GC whereas only computationally efficient operations are allowed in HC. The second stage in HC is also a new algorithm which can be used as a fast clustering method alone. For the sake of clarity, it is outlined in Algorithm 1.

Algorithm 1. Herd Clustering

```
$\vec{d}_i$: $i$th Data Vector of size $n$, $\vec{v}_i$: Velocity of $i$th Data Vector, $m$: Number of Data
  Vectors
procedure HC$data$, $threshold$
  Scale each dimension into [0, 1] using a feature transformation technique.
  // Stage 1: Moving data like herd behavior
  $iterations = 0$;
  $max = 100$;
  $terminalSpeed = threshold/2$;
  while $iterations < max$ do
    for $i = 1$ to $m$ do
      $\vec{a} = \vec{0}$;
      $total = 0$;
      for $j = 1$ to $m$ do
        if $i \neq j$ then
          if $distance(\vec{d}_i, \vec{d}_j) < threshold$ then //If the distance is less than the
  threshold
            $\vec{\delta} = \vec{d}_j - \vec{d}_i$; //$\vec{d}_i$ and $\vec{d}_j$ attract each other.
            $\vec{a} = \vec{a} + \hat{\delta}$; //Accumulate the attraction for $\vec{d}_i$.
            $total = total + 1$; //Count the number of attractions for $\vec{d}_i$.
          end if
        end if
      end for
      if $total \neq 0$ then //Implementation Trick: Division is expensive in
  computation.
        $\vec{v}_i = \vec{v}_i + \frac{\vec{a}}{total}$; //We accumulate the net attraction and do it once only
  in the last iteration.
      end if
      if $||\vec{v}_i|| > terminalSpeed$ then //Terminal speed implementation
        $\vec{v}_i = terminalSpeed * \hat{v}_i$; //$\hat{v}_i$ is the unit vector of $\vec{v}_i$.
      end if
    end for
    for $i = 1$ to $m$ do //Update the net attraction velocity for all the vectors.
      $\vec{d}_i = \vec{d}_i + \vec{v}_i$;
    end for
    $iterations = iterations + 1$;
  end while
```