An efficient fuzzy c-means approach based on canonical polyadic decomposition for clustering big data in IoT

Fanyu Bu

PII: S0167-739X(18)30647-2
DOI: https://doi.org/10.1016/j.future.2018.04.045
Reference: FUTURE 4123

To appear in: *Future Generation Computer Systems*

Received date : 22 March 2018
Revised date : 5 April 2018
Accepted date : 14 April 2018

An Efficient Fuzzy C-means Approach Based on Canonical Polyadic Decomposition for Clustering Big Data in IoT

Fanyu Bu[a,*]

[a]*Department of Biomedical Informatics, Inner Mongolia University of Finance and Economics, Hohhot, China.*

Abstract

Mining smart data from the collected big data in Internet of Things which attempts to better human life by integrating physical devices into the information space. As one of the most important clustering techniques for drilling smart data, the fuzzy c-means algorithm (FCM) assigns each object to multiple groups by calculating a membership matrix. However, each big data object has a large number of attributes, posing an remarkable challenge on FCM for IoT big data real-time clustering. In this paper, we propose an efficient fuzzy c-means approach based on the tensor canonical polyadic decomposition for clustering big data in Internet of Things. In the presented scheme, the traditional fuzzy c-means algorithm is converted to the high-order tensor fuzzy c-means algorithm (HOFCM) via a bijection function. Furthermore, the tensor canonical polyadic decomposition is utilized to reduce the attributes of every objects for enhancing the clustering efficiency. Finally, the extensive experiments are conducted to compare the developed scheme with the traditional fuzzy c-means algorithm on two large IoT datasets including sWSN and eGSAD regarding clustering accuracy and clustering efficiency. The results argue that the developed scheme achieves a significantly higher clustering efficiency with a slight clustering accuracy drop compared with the traditional algorithm, indicating the potential of the developed scheme for drilling smart data from IoT big data.

Keywords: Big data, Internet of Things, Smart data, Fuzzy c-means algorithm, Canonical polyadic decomposition

1. Introduction

Recent years has witnessed the broad application of Internet of Things (IoT) with the goal of enhancing the human life by integrating physical devices into the information space [1,2]. Specially, the representatively successful application domains of IoT include smart city management, industrial manufacturing, intrusion prevention system and so on. The typical architecture of an IoT system consists of three layers from bottom to up, namely physical layer, network layer and application layer, presented in Figure 1.

In the physical layer, a large number of physical devices for instance sensors, global positioning system (GPS) and cameras are utilized to collect raw data. For example, some sensors are utilized to collect the parameters such as temperature and pressure of various

*Corresponding author: bufanyu@imufe.edu.cn.

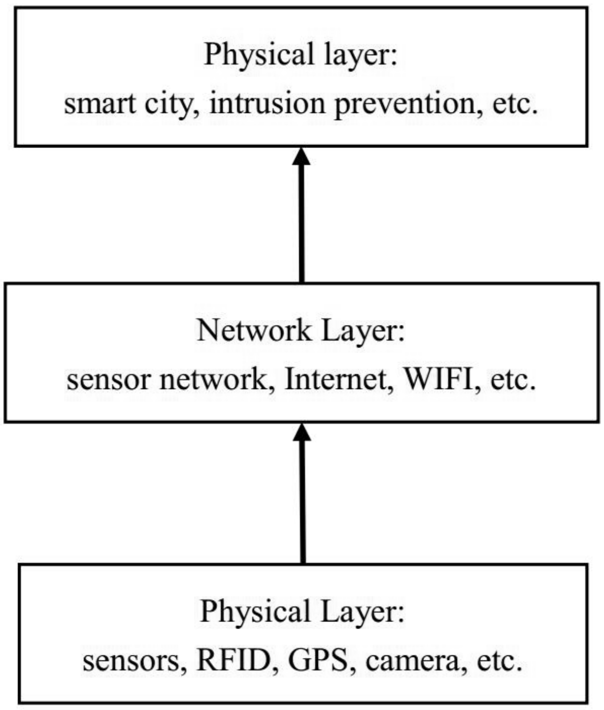
Physical layer:
smart city, intrusion prevention, etc.
Network Layer:
sensor network, Internet, WIFI, etc.
Physical Layer:
sensors, RFID, GPS, camera, etc.

Figure 1: Architecture of an IoT system with three layers.

physical devices which work in the industrial manufacturing. In particular, the volume of the raw data collected from Internet of Things is so large that it is usually called IoT big data. The network layer aims to transfer the IoT big data from the physical layer to the application layer by various networks such as wireless sensor network and Internet. Specially, the collected parameters of the physical devices in the industrial manufacturing will be transferred to the data center. The application layer mines drills smart data from IoT big data to provide all kinds of services such as intelligent decisions. For example, once the anomaly state of the physical devices in the industrial manufacturing is detected by analyzing the collected parameters in the data center, we can take immediate actions to prevent any possible disaster and destruction. According to the architecture of an IoT system, the key is to drill smart data from big data for Internet of Things to offer various intelligent services [3,4]. Commonly utilized techniques include data compression, machine learning, correlation analysis and clustering for data processing and analytics in IoT [3,5].

As one of the most leading big data mining approaches for drilling smart data, clustering attempts to divide the raw objects into multiple different groups depending on a specific similarity metric such that the objects clustered into the same group have as many similar features as possible [6]. One representative clustering approach is the fuzzy c-means algorithm (FCM) which assigns every object to multiple clusters by computing a membership matrix [7,8]. Specially, FCM can achieve an accurate clustering result and reflect the clustering model of data objectively, so it has obtained a lot of successful applications in weather forecasting and agriculture natural disaster analysis. However, it is difficult for the fuzzy c-means algorithm to satisfy the real-time requirements of IoT big data clustering because each object has a large number of attributes. Specially, a large number of attributes will significantly increase the computational cost of the fuzzy c-means algorithm. In addition, the whole dataset with all the objects should be loaded into the memory for the fuzzy c-means algorithm to yield an accurate clustering result. So, a high-performance server is required to execute the fuzzy c-means algorithm for IoT big data clustering since it has the enough memory space for IoT big data storage and a powerful computing unit to satisfy the real-time requirements of IoT big data clustering. In other words, the fuzzy c-means algorithm is hard to be utilized for big data clustering in the IoT systems with low-end computing devices due to their limited memory and computing power.

In this paper, we present an efficient fuzzy c-means approach based on the tensor canonical polyadic decomposition scheme for IoT big data clustering. In the presented method, each object is converted into its tensor format from the vector space via a bijection function. Furthermore, the tensor canonical polyadic decomposition is utilized to reduce the attributes of every object greatly. In all the tensor decomposition schemes, the canonical polyadic decomposition achieves the highest compression rate so it can reduce the volume of the raw data to the greatest extent [9,10]. Specially, assuming that every object is represented by a N-order tensor, the volume of the attributes increases exponentially with N in the raw tensor format. the volume of the attributes in the canonical polyadic decomposition format increases linearly regarding N. Therefore, after using the canonical polyadic decomposition to compress every object, the whole volume of the dataset will be reduced greatly, which makes the fuzzy c-means approach potential to cluster big data in the IoT systems with low-end devices. Another reason for the use of the canonical polyadic decomposition is that the canonical polyadic decomposition is easily implemented by existing decomposition methods for instance alternating least squares [10,11]. Finally, the traditional fuzzy c-means approach is extended to a high-order fuzzy c-means approach for clustering the compressed objects in the tensor space. Since the attributes of every object are reduced significantly by the canonical polyadic decomposition, the high-order fuzzy c-means approach can cluster IoT big data efficiently. Two typical IoT large datasets including eGSAD and sWSN are employed to carry out the extensive experiments in which the developed approach is compared with the traditional fuzzy c-means approach regarding clustering accuracy and clustering efficiency. Based on the experimental results, a higher clustering efficiency is achieved by the developed approach while it yields almost the same clustering accuracy with the traditional fuzzy c-means approach, which indicates the potential of the developed approach to cluster big data for drilling smart data in Internet of Things.

To sum up, the paper presents the following contributions:

- The raw data collected from Internet of Things is so large that the fuzzy c-means algorithm is difficult to be executed in the IoT systems with low-end computing devices. Aiming at this issue, we propose to utilize the tensor canonical polyadic decomposition to reduce the attributes of every object greatly before loading the dataset into the memory.

- We convert every object to the tensor format via a bijection so that the canonical polyadic decomposition can compress the attributes. Furthermore, we extend the fuzzy c-means approach to a high-order fuzzy c-means method to make the clustering operations executed on the compressed objects in the tensor space.

- Two typical large IoT datasets including eGSAD and sWSN are employed to conduce the experiments in which the presented efficient fuzzy c-means approach is compared with the traditional fuzzy c-means method regarding clustering efficiency and clustering accuracy. Furthermore, we discuss the experimental results.

To present the developed efficient fuzzy c-means approach clearly, we will provide the preliminaries about the fuzzy c-means method and the tensor canonical polyadic decomposition in Section 2. Afterwards, the details about the developed scheme will be provided in Section 3. Section 4 compares the experimental results between the developed approach and the traditional fuzzy c-means method and the last section concludes the paper.

2. Preliminaries

The preliminaries, including the traditional fuzzy c-means approach (FCM) and the tensor canonical polyadic decomposition, utilized for the proposed scheme are presented in this section. FCM is introduced firstly, followed by the tensor polyadic decomposition.

2.1. Fuzzy c-Means Approach (FCM)

The standard fuzzy c-means approach was firstly introduced by Beadek et al. [7]. Assuming that $X=\{x_1, x_2, \ldots, x_N\}$ is the dataset to be clustered and every object has m attributes, the standard fuzzy c-means approach assigns every object $x_i (1 \le i \le N)$ to c clusters with a $c \times N$ membership matrix $U = \{u_{ij}\}$ in which u_{ij} denotes the membership of the j-th object belonging to the i-th group. In particular, the standard fuzzy c-means approach is defined by the following equations:

$$U \in R^{c\times N} | u_{ij} \in [0, 1], \tag{1}$$

$$0 < \sum\nolimits_{j=1}^{N} u_{ij} < N, \tag{2}$$

$$\sum\nolimits_{i=1}^{c} u_{ij} = 1. \tag{3}$$

Thus, the goal of the standard fuzzy c-mans approach is to compute the membership matrix and to compute the cluster centers $V = \{v_1, v_2, \ldots, v_c\}$ given the dataset X. To this end, the standard fuzzy c-means approach randomly initializes the membership matrix and then updates the membership matrix and the cluster centers by minimizing the objective J_m as follows:

$$J_m(U, V) = \sum_{i=1}^{c}\sum_{j=1}^{N} u_{ik}^{m} d_{ik}, \tag{4}$$

where m is a fuzziness constant and d_{ik} denotes the distance between x_k and v_i, typically the Euclidean distance:

$$d_{ik} = ||x_k - v_i||^2. \tag{5}$$

To minimize the objective J_m, the updating equations for the membership matrix and the cluster centers are as follows:

$$u_{ik} = [\sum_{j=1}^{c} (\frac{d_{ik}}{d_{jk}})^{\frac{1}{m-1}}]^{-1}, \tag{6}$$

$$v_i = \frac{\sum_{k=1}^{N} (u_{ik})^m x_k}{\sum_{k=1}^{N} (u_{ik})^m}. \tag{7}$$

Overall, the standard fuzzy c-means approach is described in Algorithm 1.

Algorithm 1: The Standard Fuzzy c-Means Approach.

```
  Input: $X=\{x_1, x_2, \ldots, x_N\}$, $c$, $m$, $maxiter$
  Output: $U = \{u_{ij}\}, V = \{v_i\}$
1 Initialize $U = \{u_{ij}\}$ randomly. for $iteration = 1, 2, ..., maxiter$ do
2     for $i = 1, 2, ..., c$ do
3         Utilize Eq.(7) to update the cluster center $v_i$;
4     for $i = 1, 2, ..., c$ do
5         for $k = 1, 2, ..., N$ do
6             Utilize Eq.(6) to update the membership $u_{ik}$;
```

From Algorithm 1, the major steps of the standard fuzzy c-means approach is updating of the cluster centers with a computation complexity of $o(c)$ and updating of the membership matrix with a computational complexity of $o(cN)$. So, the total computational complexity is approximately $o(kcN)$ with k iterations.

Some enhanced fuzzy c-means approaches were introduced to cluster big data in the past decades [8,12]. For instance, the enhance fuzzy c-means approach based on random sample (rseFCM) samples a subset of objects from the raw whole dataset and then utilizes the standard fuzzy c-means approach to calculate the centers of the sampled subset [13]. Afterwards, the centers are employed to cluster the whole dataset. Overall, the rseFCM algorithm can be described in Algorithm 2.

Algorithm 2: The Enhanced Fuzzy c-Means Approach Based on Random Sample.

```
    Input: $X = \{x_1, x_2, \ldots, x_N\}$, $c$, $m$, $maxiter$
    Output: $U = \{u_{ij}\}, V = \{v_i\}$
1   Sample a subset of objects from $X$ randomly, called $X_p$. Initialize $U_p$ for $X_p$ randomly. for $iteration = 1, 2, ..., maxiter$ do
2       for $i = 1, 2, ..., c$ do
3           Utilize Eq.(7) to update the cluster center $v_i$;
4       for $i = 1, 2, ..., c$ do
5           for $k = 1, 2, ..., N$ do
6               Utilize Eq.(6) to update the membership $u_{(p)ik}$;
7       Utilize Eq.(6) to extend the partial result $(U_p, V)$ for $X$.
```

In the standard fuzzy c-means approach, every object has the same importance. Some weighted fuzzy c-means approaches were introduced recently [14]. Specially, a typical weighted fuzzy c-means approach was described by assigning a weight w_j to each object x_j, in which the weight w_j defines the importance of x_j, leading to the objective $J_{mw}(U, V)$:

$$J_{mw}(U, V) = \sum_{i=1}^{c} \sum_{k=1}^{N} w_k u_{ik} d_{ik}. \tag{8}$$

Minimizing the objective $J_{mw}(U, V)$ results in an updating function for the cluster centers:

$$v_i = \frac{\sum_{k=1}^{N} w_k (u_{ik})^m x_k}{\sum_{k=1}^{N} w_k (u_{ik})^m}. \tag{9}$$

The weighted fuzzy c-means approach is described in Algorithm 3.

Algorithm 3: The Weighted Fuzzy c-Means Approach.

```
    Input: $X = \{x_1, x_2, \ldots, x_N\}$, $c$, $m$, $maxiter$
    Output: $U = \{u_{ij}\}, V = \{v_i\}$
1   Initialize $U = \{u_{ij}\}$ randomly. for $iteration = 1, 2, ..., maxiter$ do
2       for $i = 1, 2, ..., c$ do
3           Utilize Eq.(7) to update the cluster center $v_i$;
4       for $i = 1, 2, ..., c$ do
5           for $k = 1, 2, ..., N$ do
6               Utilize Eq.(9) to update the membership $u_{ik}$;
```

To cluster high-dimensional data, the kernel fuzzy c-means approaches [15,16] project each object to the kernel space via: $\phi : x \rightarrow \phi(x)$. Most commonly used kernel functions include polynomial kernel and radial basis kernel:

$$k(x_i, x_j) = (x_i^T x_j + 1)^p, \tag{10}$$

$$k(x_i, x_j) = \exp(\sigma ||x_i - x_j||^2). \tag{11}$$

The kernel fuzzy c-means approach updates the membership matrix as:

$$u_{ij} = [\sum_{k=1}^{c} (\frac{d_k(x_i, v_j)}{d_k(x_i, v_k)})^{\frac{1}{m-1}}]^{-1}, \tag{12}$$

where $d_k(x_i, v_j)$ denotes the kernel distance between x_i and v_j.

The cluster centers are updated via:

$$\phi(v_i) = \frac{\sum_{k=1}^{N} (u_{ik})^m \phi(x_k)}{\sum_{k=1}^{N} (u_{ik})^m}. \tag{13}$$

Furthermore, Havens et al. [12] presented a weighted kernel fuzzy c-means approach to cluster very large data by combining the weighted fuzzy c-means approach and the kernel fuzzy c-means approach. The weighted kernel fuzzy c-means approach updates the membership matrix and cluster centers via the following two equations.

$$\phi(v_i) = \frac{\sum_{k=1}^{N} w_k (u_{ik})^m \phi(x_k)}{\sum_{k=1}^{N} w_k (u_{ik})^m}, \tag{14}$$

$$u_{ij} = [\sum_{k=1}^{c} (\frac{d_k^w(x_i, v_j)}{d_k^w(x_i, v_k)})^{\frac{1}{m-1}}]^{-1}. \tag{15}$$

The weighted kernel fuzzy c-means approach can be described in Algorithm 4.

More recently, Zhang et al. [17] introduced a distributed fuzzy c-means approach based on MapReduce to cluster big data by defining two variables for the Map function:

$$\xi_i^{(t)} = \sum_{k=1}^{N/p} u_{ik}^m x_k, \tag{16}$$

$$\lambda_i^{(t)} = \sum_{k=1}^{N/p} u_{ik}^m, \tag{17}$$

Algorithm 4: The Weighted Kernel Fuzzy c-Means Approach.

```
  Input: $X = \{x_1, x_2, \ldots, x_N\}$, $c$, $m$,
         $maxiter$
  Output: $U = \{u_{ij}\}, V = \{v_i\}$
1 Initialize $U = \{u_{ij}\}$ randomly. for
  $iteration = 1, 2, ..., maxiter$ do
2     for $i = 1, 2, ..., c$ do
3         Utilize Eq.(14) to update the
          cluster center $v_i$;
4     for $i = 1, 2, ..., c$ do
5         for $k = 1, 2, ..., N$ do
6             Utilize Eq.(15) to update the
              membership $u_{ik}$;
```

where p is the number of the computers in the cloud computing platform.

The Reduce function aims to compute the centers via:

$$v_i = \frac{\sum_{t=1}^{p} \xi_i^{(t)}}{\sum_{t=1}^{p} \lambda_i^{(t)}}. \tag{18}$$

To cluster heterogeneous data, Li et al. [18] developed a high-order fuzzy c-means approach based on tensors. Other representative enhanced fuzzy c-means approaches include single-pass fuzzy c-means approach [19] and online fuzzy c-means approach [20] which are based on the incremental manner for big data clustering.

2.2. Tensor Canonical Polyadic Decomposition Scheme

From a mathematical point of view, a vector and a matrix can be viewed as a 1-order tensor and 2-order tensor, respectively, while a 0-order tensor is a scalar [21]. Figure 3 presents an example of a 3-order tensor.

Specially, tensors can represent the heterogeneous data. For instance, an objec-

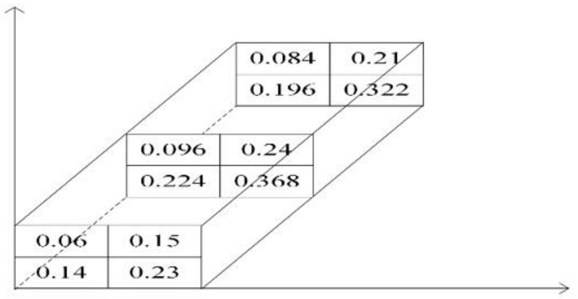

Figure 2: Example of a 3-order tensor.

t with m attributes can be represented by a m-dimensional vector while a gray image can be denoted by a matrix $R^{h\times w}$ with h and w are the height and weight, respectively. Furthermore, an image in the RGB color space is usually represented as a 3-order tensor $R^{h\times w\times c}$ where c denotes the color channel.

Tensors have been widely employed in big data analytics and processing in the past few years. For example, Zhang et al. [22] presented a tensor deep learning model for big data feature learning based on the tensor distance. To enhance the training efficiency, they used tensor decomposition schemes to reduce the parameters greatly [11,23]. A tensor decomposition approach attempts to factorize tensor to one core tensor and a set of component matrices [5]. The core tensor is used to link the component matrices over different modes. In particular, the canonical polyadic decomposition factorizes an N-order tensor to the sum of R rank-1 tensors via:

$$\begin{aligned} X &= \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ \cdots \circ a_r^{(N)} \\ &= [[A^{(1)}, A^{(2)}, \ldots, A^{(N)}]] \end{aligned}, \quad (19)$$

where $\circ$ is the outer product.

A 4-dimensional vector b and a 3-dimensional vector a will yield a 4×3 matrix c via the outer product:

$$\begin{aligned} c = b \circ a &= \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} [a_1, a_2, a_3] \\ &= \begin{bmatrix} a_1b_1 & a_2b_1 & a_3b_1 \\ a_1b_2 & a_2b_2 & a_3b_2 \\ a_1b_3 & a_2b_3 & a_3b_3 \\ a_1b_4 & a_2b_4 & a_3b_4 \end{bmatrix} \end{aligned} . \quad (20)$$

By introducing a core tensor, the standard canonical polyadic decomposition can be computed via:

$$\begin{aligned} X &= \sum_{r=1}^{R} \lambda_r b_r^{(1)} \circ b_r^{(2)} \circ \cdots \circ b_r^{(N)} \\ &= \Lambda \times_1 B^{(1)} \times_2 B^{(2)} \cdots \times_N B^{(N)} \\ &= [[\Lambda; B^{(1)}, B^{(2)}, \ldots, B^{(N)}]] \end{aligned} . \quad (21)$$

Figure 3 presents an example of the standard canonical polyadic decomposition.

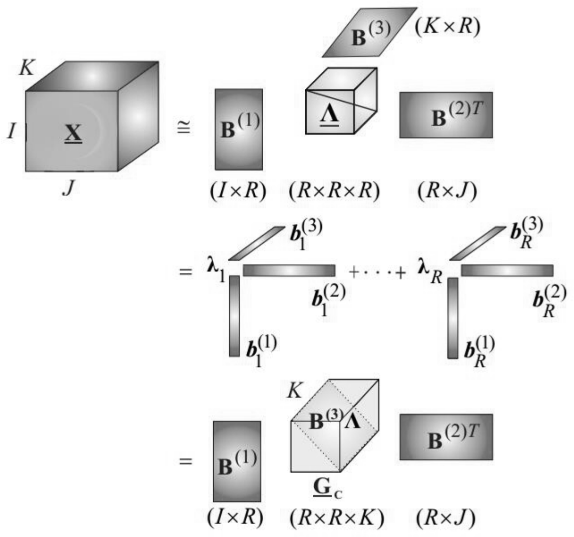

Figure 3: Example of canonical polyadic decomposition.

Given an N-order tensor X, the approach for the canonical polyadic decomposition should calculate the matrices $A^{(n)}(n =$

$1, 2, \ldots, N$). A representative example is the Alternating Least Squares algorithm that calculates $A^{(n)}$ by solving an optimization problem: $\min : ||\underline{X} - \hat{\underline{X}}|| = ||\underline{X} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ \cdots \circ a_r^{(N)}|| = ||\underline{X} - [\lambda; A^{(1)}, A^{(2)}, \ldots, A^{(N)}]||$.

Other two commonly employed tensor decompositions include the Tucker decomposition and the tensor-train network, presented in Figure 4 and Figure 5, respectively [10].

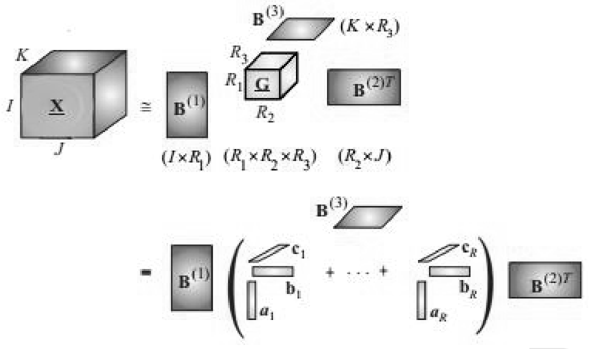

Figure 4: Example of Tucker decomposition.

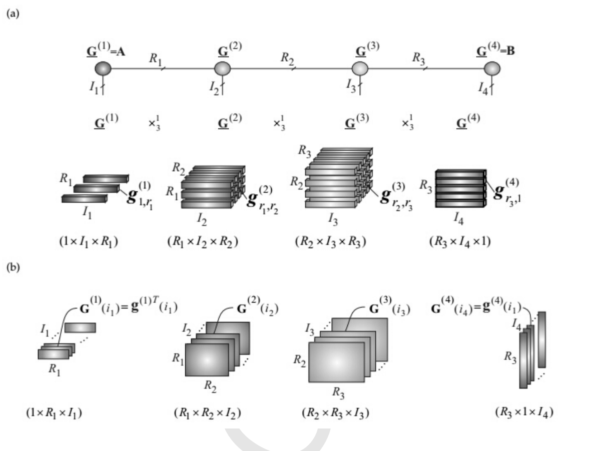

Figure 5: Example of tensor-train network.

Compared with other tensor decompositions, the canonical polyadic decomposition has a highest compression rate so it is obtaining some applications in data mining and deep learning [11].

3. The Efficient Fuzzy c-Means Approach Based on Tensor Canonical Polyadic Decomposition Scheme

In this section, the efficient fuzzy c-means approach based on tensor canonical polyadic decomposition scheme for big data clustering is described in detail. To utilize the tensor polyadic decomposition scheme to compress the dataset, each object to be clustered needs to be converted into the tensor format.

Assuming that each object o_i has M attributes, i.e., $o_i \in R^M$, we utilize the bijection f defined in [24] to convert it into the tensor format $O_i \in R^{G_1 \times G_2 \times \cdots \times G_T}$. In particular, f defines the bijection between the coordinate $g \in \{1, 2, \ldots, M\}$ of o_i and the tensor $f(g) = (f_1(g), f_2(g), \ldots, f_T(g))$ of O_i, in which $f_t(g) \in \{1, 2, \ldots, G_t\}$. So, the item $O_{ig_1g_2\ldots g_T}$ of the tensor O_i via the bijection:

$$O_{ig_1g_2\ldots g_T} = O_{if(g)} = o_{ig}. \quad (22)$$

Thus, each object is converted into a T-order tensor.

After converting every object into the T-order tensor format, a tensor fuzzy c-means approach can be obtained via the following equations for updating the membership matrix and the cluster centers.

$$u_{ik} = [\sum_{j=1}^{c} (\frac{d_{(T)ik}}{d_{(T)jk}})^{\frac{1}{m-1}}]^{-1}, \quad (23)$$

$$v_{ig_1g_2\ldots g_T} = \frac{\sum_{k=1}^{N} (u_{ik})^m O_{kg_1g_2\ldots g_T}}{\sum_{k=1}^{N} (u_{ik})^m}, \quad (24)$$

where $d^2_{(T)ij}$ is the distance between X_j and V_i via:

$$d^2_{(T)ij} = \sum_{g_1\cdots g_T}^{G_1\cdots G_T} (X_{jg_1\cdots g_T} - V_{ig_1\cdots g_T})^2. \quad (25)$$

Algorithm 5 describes the tensor fuzzy c-means approach.

Algorithm 5: The Tensor Fuzzy c-Means Approach.

```
  Input: $X = \{x_1, x_2, \ldots, x_N\}$, $c$, $m$, $maxiter$
  Output: $U = \{u_{ij}\}, V = \{v_i\}$
1 Initialize $U = \{u_{ij}\}$ randomly. for $iteration = 1, 2, ..., maxiter$ do
2     for $i = 1, 2, ..., c$ do
3         Utilize Eq.(24) to update the cluster center $v_i$;
4     for $i = 1, 2, ..., c$ do
5         for $k = 1, 2, ..., N$ do
6             Utilize Eq.(23) to update the membership $u_{ik}$;
```

However, the tensor fuzzy c-means approach cannot decrease the computational cost or save the memory space for the standard fuzzy c-means approach. Therefore, we utilize the canonical polyadic decomposition to factorize each object $O_i \in R^{G_1 \times G_2 \times \cdots \times G_T}$ via:

$$\begin{aligned} O_i &= \sum_{r=1}^{R} a_{ir}^{(1)} \circ a_{ir}^{(2)} \circ \cdots \circ a_{ir}^{(T)} \\ &= [[A_i^{(1)}, A_i^{(2)}, \ldots, A_i^{(T)}]] \end{aligned}, \tag{26}$$

Alternatively, every item in O_i can be denoted by the following format according to the canonical polyadic decomposition:

$$O_{ig_1 \ldots g_T} = \sum_{r=1}^{R} A_{i(g_1,r)}^{(1)} \cdots A_{i(g_T,r)}^{(T)}. \tag{27}$$

Let $G = \max\{G_1, G_2, \ldots, G_T\}$, every object requires the storage space of $O(G^T)$ in the raw format. However, the storage space required by every object is reduced to $O(GTR)$ in the format of the canonical polyadic decomposition.

Generally speaking, the number of cluster centers is significantly smaller than the number of the objects in the big dataset. We do not need to convert the cluster centers into the format of the canonical polyadic decomposition since the converting could not reduce the storage cost greatly. More importantly, using the canonical polyadic decomposition of the cluster centers yields much computation overhead in the clustering process. Therefore, the cluster centers are not converted into the canonical polyadic decomposition format in the presented efficient fuzzy c-means approach.

By applying Eq.(27) to Eq.(4), the objective of the efficient fuzzy c-means approach can be obtained as follows.

$$\begin{aligned} J_m(U,V) = &\sum_{i=1}^{c} \sum_{j=1}^{N} u_{ij}^m \sum_{i_1 \cdots i_N}^{I_1 \cdots I_N} \\ &(\sum_{r=1}^{R} A_{j(g_1,r)}^{(1)} \cdot A_{j(g_2,r)}^{(2)} \cdots A_{j(g_T,r)}^{(T)} - V_{ig_1 g_2 \ldots g_T})^2 \end{aligned}. \tag{28}$$

By minimizing the objective of the efficient fuzzy c-means approach, we can obtain the functions for updating the membership matrix and the cluster centers as follows.

$$u_{ik} = [\sum_{j=1}^{c} (\frac{d_{Tik}}{d_{Tjk}})^{\frac{1}{m-1}}]^{-1}, \tag{29}$$

$$V_{ig_1 \ldots g_T} = \frac{\sum_{k=1}^{N} u_{ik}^m \sum_{r=1}^{R} A_{k(g_1,r)}^{(1)} \cdots A_{k(g_T,r)}^{(T)}}{\sum_{k=1}^{N} u_{ik}^m}, \tag{30}$$

where d_{Tik} can be computed via:

$$d_{Tik} = \sum_{g_1 \cdots g_T}^{G_1 \cdots G_T} (\sum_{r=1}^{R} A_{k(g_1,r)}^{(1)} \cdots A_{k(g_T,r)}^{(T)} - V_{ig_1 \cdots g_T})^2. \tag{31}$$

The efficient fuzzy c-means approach is described in Algorithm 6.

Algorithm 6: The Efficient Fuzzy c-Means Approach Based on Canonical Polyadic Decomposition.

```
   Input: $X = \{x_1, x_2, \ldots, x_N\}$, $c$, $m$, $maxiter$
   Output: $U = \{u_{ij}\}, V = \{v_i\}$
1  Initialize $U = \{u_{ij}\}$ randomly. for $k = 1, 2, ..., N$ do
2      Utilize Eq.(27) to decompose $X_k$;
3  for $iteration = 1, 2, ..., maxiter$ do
4      for $i = 1, 2, ..., c$ do
5          Utilize Eq.(30) to update the cluster center $v_i$;
6      for $i = 1, 2, ..., c$ do
7          for $k = 1, 2, ..., N$ do
8              Utilize Eq.(29) to update the membership $u_{ik}$;
```

4. Experiments

Two typical IoT datasets including eGSAD and sWSN [17] are employed to compare the performance between the standard fuzzy c-means approach and the efficient fuzzy c-means approach in the experiments. Two metrics are employed to judge the performance of the efficient fuzzy c-means approach, i.e., Adjusted Rand Index (ARI) and $E*$. ARI is employed to evaluate the clustering accuracy while $E*$ is employed to evaluate the cluster centers. Specially, let U and $U*$ denote the actual labels of the objects to be clustered and the cluster yielded by the specific clustering approach $*$, respectively. $ARI(U, U*)$ is widely employed to measure the agreement between the actual labels and a specific fuzzy clustering approach. In particular, the higher $ARI(U, U*)$ implies a more accurate clustering result yielded by the clustering approach. $E*$ judges the cluster centers yielded by the approach $*$ via:

$$E_* = \sqrt{\sum_{i=1}^{c} ||v_{ideal}^i - v_*^i||^2}, \qquad (32)$$

where v_{ideal} is the actual center set and $v*$ denotes the cluster center set produced by the specific clustering algorithm $*$.

According to the extensive studies, $m = 2$ can lead to good clustering results in most cases. So, the paper sets $m = 2$ in the experiments.

4.1. Experiments on eGSAD

In general, the clustering results are significantly affected by the rank R. So, we judge the performance of the efficient fuzzy c-means approach with different ranks regarding running time. Figure 6 presents the experimental results.

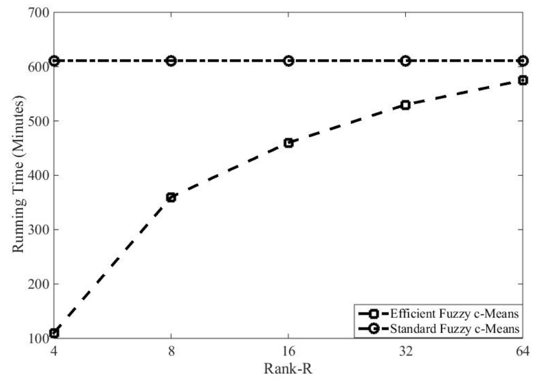

Figure 6: Running time on eGSAD.

Based on the results shown in Figure 6, as the rank R increases from 4 to 64, the running time of the efficient fuzzy c-means approach grows gradually. As the rank increases, the compression rate of the attributes is reduced, so the running time grows. However, the running time of the efficient fuzzy c-means approach is less than the standard fuzzy c-means. For example, when $R = 16$, the running time of the efficient fuzzy c-means approach and the standard fuzzy c-means approach is 460 minutes and 611 minutes, respectively. Therefore, the presented approach is more efficient than the standard fuzzy c-means approach for clustering the eGSAD dataset.

The experimental results regarding clustering accuracy with different ranks are described in Table 1 and Table 2.

The clustering accuracy in terms of $E*$ and ARI is presented in Table 1 and Table 2.

From Table 1 and Table 2, as the rank R increases from 4 to 64, the clustering accuracy of the efficient fuzzy c-means approach grows gradually regarding $E*$ and ARI. When R is small, a large number of attributes are compressed, so the efficient fuzzy c-means approach produces the significantly lower clustering accuracy than the standard fuzzy c-means approach. For example, when $R = 8$, the efficient fuzzy c-means approach yields $E*$ and ARI of 26.94 and 0.71, respectively. The standard fuzzy c-means approach produces $E*$ and ARI of 12.98 and 0.89, respectively. However, when $R \geq 32$, the efficient fuzzy c-means approach produces the considerable clustering accuracy compared with the standard fuzzy c-means approach. In particular, when $R = 64$, the efficient fuzzy c-means approach yields the same clustering accuracy with the standard fuzzy c-means approach regarding ARI. However, such large rank R is not desirable since the clustering efficiency can not be enhanced greatly in this case. According to the results from Figure 6, Table 1 and Table 2, $R = 32$ is large enough for the eGSAD dataset. In this case, the efficient fuzzy c-means approach yields $E*$ and ARI of 13.69 and 0.87 while the clustering efficiency is enhanced significantly. Such the results imply that the attributes of the raw objects are redundant.

4.2. Experiments on sWSN

In this part, we judge the performance of the efficient fuzzy c-means approach with different ranks regarding running time on the sWSN dataset. Figure 7 presents the experimental results.

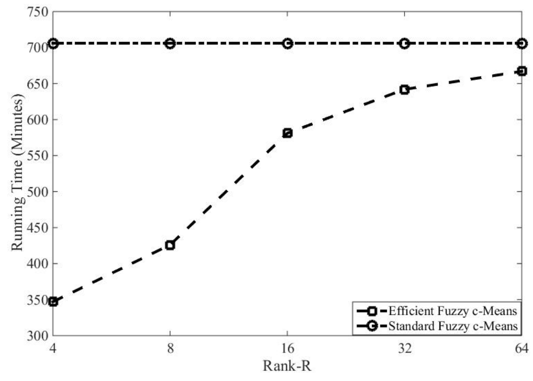

Figure 7: Running time on sWSN.

The clustering results regarding $E*$ and ARI is described in the following two tables.

From the above experimental results, we can draw three conclusions. First, the growth of the rank R increases the running time of the efficient fuzzy c-means approach for clustering the sWSN dataset. For example, when the rank grows from 8 to 32, the running time of the efficient fuzzy c-means approach increases from 462 minutes to 643 minutes for clustering the sWSN dataset. However, the efficient fuzzy c-means approach takes significantly less time than the standard fuzzy c-means approach to cluster the sWSN dataset when R is small. Even when $R = 64$, the running time of the efficient fuzzy c-means approach is less than the standard fuzzy c-means approach. In particular, in this case, the running time of the efficient fuzzy c-means approach and the standard

Table 1: Clustering accuracy regarding $E*$ on eGSAD

Algorithm/Rank	4	8	16	32	64
Standard Approach	12.98	12.98	12.98	12.98	12.98
Efficient Approach	31.07	26.94	14.21	13.69	13.22

Table 2: Clustering accuracy regarding ARI on eGSAE

Algorithm/Rank	4	8	16	32	64
Standard Approach	0.89	0.89	0.89	0.89	0.89
Efficient Approach	0.64	0.71	0.86	0.87	0.89

Table 3: Clustering accuracy regarding $E*$ on sWSN

Algorithm/Rank	4	8	16	32	64
Standard Approach	0.59	0.59	0.59	0.59	0.59
Efficient Approach	1.26	1.07	0.66	0.62	0.61

Table 4: Clustering accuracy regarding ARI on sWSN

Algorithm/Rank	4	8	16	32	64
Standard Approach	0.91	0.91	0.91	0.91	0.91
Efficient Approach	0.73	0.78	0.89	0.89	0.90

fuzzy c-means approach is around 660 minutes and 700 minutes, respectively. Therefore, the presented approach achieves more efficiency than the standard fuzzy c-means approach for clustering the sWSN dataset. Second, the growth of the rank R improves the clustering accuracy of the efficient fuzzy c-means approach regarding $E*$ and ARI. For example, when R grows from 4 to 64, the efficient fuzzy c-means approach reduces $E*$ from 1.26 to 0.61 while it increases ARI from 0.73 to 0.90. However, $R = 64$ is not required since the clustering efficiency can not be improved greatly in this case even though the clustering accuracy is considerable compared with the standard fuzzy c-means approach. When $R = 32$, the efficient fuzzy c-means approach can obtain the considerable clustering accuracy with the standard fuzzy c-means approach. In this case, the presented approach obtains $E*$ and ARI of 0.62 and 0.89 while the standard obtains $E*$ and ARI of 0.59 and 0.91, respectively. This result concludes that the attributes of the raw dataset is highly redundant. Finally, the efficient fuzzy c-means approach is more efficient than the standard fuzzy c-means approach without a large accuracy drop, proving the potential of the efficient fuzzy c-means approach for clustering IoT big data.

5. Conclusion

In this paper, we developed an efficient fuzzy c-means approach based on the tensor canonical polyadic decomposition to cluster IoT big data for drilling smart data. In particular, the tensor canonical polyadic decomposition scheme is employed to reduce the attributes of each object in the raw dataset significantly. On the one hand, the presented approach could reduce the amount of the data greatly, which makes the fuzzy c-means approach possible to run in the low-end devices in the IoT systems. On the other hand, the presented approach could enhance the clustering efficiency of the fuzzy c-means approach greatly for IoT big data clustering. Furthermore, we compared the clustering efficiency and running time between the presented efficient fuzzy c-means approach and the standard fuzzy c-means approach on two representative IoT datasets, i.e., eGASD and sWSN. Results clearly point out that the presented approach achieved more high efficiency than the standard fuzzy c-means approach while the presented approach obtained the considerable clustering accuracy with the standard fuzzy c-means approach regarding $E*$ and ARI. Such results proving the potential of the presented efficient fuzzy c-means approach based on the tensor canonical polyadic decomposition. Therefore, we will judge the clustering accuracy and the clustering efficiency of the developed approach in the real IoT systems with low-end computing devices in the future work. In addition, the fuzzy c-means approach is generally affected by the initialization, so the future work will investigate the effective method to initialize the membership to further improve the performance of the presented approach.

6. Acknowledge

This work support was partially supported by National Natural Science Foundation of China Grant No. 61762068, Natural Science Foundation of The Inner Mongolia Autonomous Region Grant No. 2017MS0610 and Higher Educational Scientific Research Projects of Inner Mongolia Autonomous Region No. NJZY16145

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications, IEEE Communications Surveys and Tutorials 17(4)(2015) 2347-2376.

[2] D. Zhang, L. T. Yang, M. Chen, S. Zhao, M. Guo, Y. Zhang, Real-Time Locating Systems Using Active RFID for Internet of Things, IEEE Systems Journal 10(3)(2017) 1226-1235.

[3] Q. Zhang, L. T. Yang, Z. Chen, P. Li, F. Bu, An Adaptive Dropout Deep Computation Model for Industrial IoT Big Data Learning with Crowdsourcing to Cloud Computing, IEEE Transactions on Industrial Informatics 2018, 10.1109/TII.2018.2791424.

[4] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. siddiqa, I. Yaqoob, Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges, IEEE Access 5(2017) 5247-5261.

[5] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, P. Li, Energy-Efficient Scheduling for Real-Time Systems Based on Deep Q-Learning Model, IEEE Transactions on Sustainable Computing 2017, DOI: 10.1109/TSUSC.2017.2743704..

[6] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. R. Patel, A. Tiwari, M. J. Er, W. Ding, C. Lin, A Review of Clustering Techniques and Developments, Neurocomputing 267(2017) 664-681.

[7] J. C. Bezdek, R. Ehrlich, W. Full, FCM: The Fuzzy c-Means Clustering Algorithm, Computers and Geosciences 10(2-3)(1984) 191-203.

[8] J. Nayak, B. Naik, H. S. Behera, Fuzzy C-Means (FCM) Clustering Algorithm: A Decade Review from 2000 to 2014, Proceedings of Computational Intelligence in Data Mining, (2015), pp. 133-149, Springer.

[9] Q. Zhang, L. T. Yang, Z. Yan, Z. Chen, P. Li, An Efficient Deep Learning Model to Predict Cloud Workload for Industry Informatics, IEEE Transactions on Industrial Informatics 2018, 10.1109/TII.2018.2808910.

[10] A. Cichocki, Era of big data processing: A New Approach via Tensor Networks and Tensor Decompositions, arXiv preprint arXiv: 1403.2048, 2014.

[11] Q. Zhang, L. T. Yang, Z. Chen, P. Li, An Improved Deep Computation Model Based on Canonical Polyadic Decomposition, IEEE Transactions on Systems, Man, and Cybernetics: Systems 2017, DOI: 10.1109/TSMC.2017.2701797.

[12] T. C. Havens, J. C. Bezdek, C. Lechkie, L. OL.

Hall, M. Palaniswami, Fuzzy c-Means Algorihms for Very Large Data, IEEE Transactions on Fuzzy Systems 20(6)(2012) 1130-1141.

[13] R. J. Hathaway, J. C. Bezdek, Extending Fuzzy and Probabilistic Clustering to Very Large Data Sets, Computational Statistics and Data Analysis 51(1)(2006) 215-234.

[14] C. Hung, S. Kulkarni, B. Kuo, A New Weighted Fuzzy C-Means Clustering Algorithm for Remotely Sensed Image Classification, IEEE Journal of Selected Topics in Signal Processing 5(3)(2011) 543-553.

[15] M. Gong, Y. Liang, J. Shi, W. Ma, J. Ma, Fuzzy C-Means Clustering With Local Information and Kernel Metric for Image Segmentation, IEEE Transactions on Image Processing 22(2)(2013) 573-584.

[16] D. Zhang, S. Chen, Clustering Incomplete Data Using Kernel-Based Fuzzy C-means Algorithm, Neural Processing Letters 18(3)(2003) 155-162.

[17] Q. Zhang, Z. Chen, Y. Leng, Distributed Fuzzy c-Means Algorihms for Big Sensor Data Based on Cloud Computing, International Journal of Sensor Network 18(1-2)(2015) 32-39.

[18] P. Li, Z. Chen, L. T. Yang, L. Zhao, Q. Zhang, A Privacy-preserving High-order Neuro-fuzzy c-Means Algorithm with Cloud Computing, Neurocomputing 256(2017) 82-89.

[19] P. Hore, L. Hall, and D. Goldgof, Single Pass Fuzzy c-Means, Proceedings of IEEE International Conference on Fuzzy Systems, (2007), pp. 1-7, IEEE.

[20] P. Hore, L. Hall, D. Goldgof, Y. Gu, and A. Maudsley, A Scalable Framework for Segmenting Magentic Resonance Images, Journal of Signal Processing Systems 54(1-3)(2009) 183-203.

[21] D. Kaur, G. S. Aujla, N. Kumar, A. Zomaya, C. Perera, R. Ranjan, Tensor-based Big Data Management Scheme for Dimensionality Reduction Problem in Smart Grid Systems: SDN Perspective, IEEE Transactions on Knowledge and Data Engineering 2018, DOI: 10.1109/TKDE.2018.2809747.

[22] Q. Zhang, L. T. Yang, Z. Chen, P. Li, A Survey on Deep Learning on Big Data, Information Fusion 42(2018) 146-157.

[23] Q. Zhang, L. T. Yang, Z. Chen, P. Li, A Tensor-train Deep Computation Model for Industry Informatics Big Data Feature Learning, IEEE Transactions on Industrial Informatics 2018, DOI: 10.1109/TII.2018.2791423.

[24] A. Novikov, D. Podoprikhin, A. Osokin, D. Vetrov, Tensorizing Neural Networks, Proceedings of Advances in Neural Information Processing Systems, (2015), pp. 442-450, MIT.

Fanyu Bu received the Bachelor’s degree in computer science from Inner Mongolia Agricultural University, Hohhot, China, in 2003, and the Master’s degree in computer application from Inner Mongolia University, Hohhot, China, in 2009. He is currently an assistant professor at Department of Biomedical Informatics in Inner Mongolia University of Finance and Economics, China. His research interests include Big Data and Internet of Things.