DECENTRALIZED WEIGHTED K-MEANS FOR CLUSTERING LARGE DATASETS OVER PEER-TO-PEER NETWORK

BINJU THANKACHAN\(^1\) and S. V. SUDHA\(^2\)

Department of Computer Science and Engineering, Dr. N. G. P. Institute of Technology, Coimbatore, Tamil Nadu, India. \(^1\)binjuelizabeth20@gmail.com; \(^2\)vsudha@rediffmail.com

ABSTRACT

For several Enterprise Applications, increase of data volumes turned to be infeasible to be kept in a single machine. Distributed storage has become the most efficient way for maintaining huge amounts of data. Consequently, Distributed Data Mining (DDM) become an active research area. Distributed Data Mining applies techniques to mine distributed data sources by avoiding the necessity to first collect the information into a central site. This has a significant appeal when issues of communication cost and privacy place a restriction on traditional centralized methods. The paper describe a general fully decentralized clustering method to cluster distributed data across peer-to-peer environments. The proposed methodology can be instantiated to partitioned-based clustering algorithm. Nodes gradually build a summarized view on the global data set which is the basis for executing weighted versions of the clustering algorithm to build clustering model. Through experimental results the effectiveness of method to achieve a high-quality global clustering solution is demonstrated, which approximates centralized clustering.

Keywords: Distributed Data Mining, Distributed Clustering, Peer to Peer Data Mining, Partition-based Clustering.

INTRODUCTION

A recent shift toward distributed data mining (DDM) was sparked by the data mining community since the mid-1990s. It had been realized that analyzing massive data sets, which often span different sites, using traditional centralized approaches can be infeasible due to computation, storage and communication overheads. In addition, DDM is being used for recent advances in grid infrastructures and distributed computing platforms. DDM works on distributed data directly, by avoiding pooling of data into a central location first. The main goal is to derive, by exploring methods of various data mining techniques, a global model that presents the characteristics of a data set distributed across different sites.

Data Clustering an unsupervised classification is one of the major data mining problems. Clustering is a method of partitioning a set of data into groups of meaningful sub-classes, called clusters [4]. In a distributed computing paradigm, the goal must be achieved when data cannot be concentrated on a single machine, due to huge amount of distributed data. Typical applications requiring distributed clustering include: clustering different media metadata from different locations; clustering nodes activity history data; clustering books in a distributed network of libraries; clustering scientific achievements from different institutions and publishers.

Among the various distributed computing paradigms distributed Peer-to-Peer (P2P) computing is presently the topic of one of the largest bodies of both theoretical and applied analysis [2]. Distributed data mining is gaining increasing attention in this domain for advanced data driven applications. Peer to Peer Data Mining (P2PDM) is a type of DDM where there is no notion of centralization to the mining method, all nodes are considered as peers. P2PDM scenarios typically exist when no single node owns the whole data set. In such cases, the data will naturally distributed over a large number of nodes and the aim is to find a clustering solution that takes into account the entire data set. Here all nodes cooperate with each other to perform a critical function in a completely decentralized manner [12,14]. The main goal of P2P data mining is to attain an equivalent (or close) data mining result that an centralization approach obtains, without moving any data from its original location.

Common approach in distributed clustering method is to combine and merge local representation in a central node, or aggregate local models in hierarchical structures [6]. Distributed clustering carried out in two levels, i.e., the local level and the global level. Such approach suffers from huge communication and computation cost to pool and mine the global data. Another problem is that it may not scale well with the number of sites. Some methods even though fully decentralized, include maintaining history of clustering process [8,9,10].

In the paper, an fully decentralized weighted K-means algorithm is proposed for clustering large data sets distributed over P2P network. We first introduce a basic methodology during which nodes gradually build a summarized read of the data set by continuously exchanging their information about data items using a gossip-based communication [11]. Gossiping is employed as an efficient dissemination technique, which assumes no predefined structures in the network. The summarized read is a basis for executing weighted versions of clustering algorithm. This method achieve a high-quality global clustering solution, which is an approximation of centralized clustering. We also explain effects of accuracy of our derived clustered model.

The paper has the following sections. Section 2 discuss about the related works. Section 3 describes about the system model for the proposed system, section 4 describes the experimental results and section 5 describes conclusion and future work of the paper.
RELATED WORKS

Distributed Data Mining where originally developed for mining over decentralized data sources. A discussion of several Distributed Data Mining algorithms, methods in order to discover knowledge is provided in [3,29,21]. Clustering distributed data has been addressed in many publications over the past decade. K-window algorithm [27] for distributed setting is proposed that transfers local clusters into a central site to merge as a global model. Eyal et al. [30] provide a generic algorithm for clustering in a static network.

Apart from existing distributed clustering methods, our algorithm does not require a central site to coordinate execution rounds, and/or merge local models [1]. Some distributed clustering methods need a special structure in the network. A hierarchical clustering method based on K-means for P2P network is provided in [22]. Eshref Januzaj [5] introduces a distributed density-based clustering algorithm, Scalable Density-Based Distributed Clustering (SDBC) that summarizes local representatives, and transmit them to a central site to be merged. Fatta et al. [23] propose a gossip-based distributed k-means clustering, initiated with similar centroids, and proceeds towards centroid convergence with rounds of gossiping. Datta et al. [7] propose a decentralized K-means algorithm for P2P network in which nodes communicate with their immediate neighbours. A probabilistic model of the data at each local site is built to account for privacy requirements in distributed clustering is proposed in [19].

Some methods consider pure unstructured networks, require state-aware operation of nodes, work in static settings, or aimed for computing basic functions like average and sum. Fellus et al. [10] propose a decentralized K-means algorithm which executes in iterations, and in each iteration nodes compute an approximation of the new centroids in a distributed manner. A K-means monitoring algorithm is proposed in [20]. This method executes K-means by iteratively combining data samples at a central site. A distributed partition-based clustering algorithm for clustering documents in a P2P is proposed by Eisenhardt et al. [17]. An parallel implementation of k-means for distributed memory multiprocessor is proposed based on message passing model [25]. RACHET [28] proposes an hierarchical clustering method for very large, high dimensional, and horizontally distributed datasets. In this method, each site will execute the algorithm in a local manner and transmit a set of representatives to a central site.

The major drawback of the majority of existing approaches, is there is no efficient solutions for handling intersection between summarized data of different representatives. Also, majority of approaches have larger communication overhead due to data transfer between local and global site.

SYSTEM MODEL

We consider a set \( P = \{p_1, p_2, ..., p_n\} \) of \( n \) networked nodes. Each node \( p \in P \) stores and shares a set of data points \( D_p^{\text{int}} \), denoted as its internal data, points may added or removed from this set. \( D = \bigcup_{p \in P} D_p^{\text{int}} \) is the set of all data points offered within the network. While discovering clusters, \( p \) may additionally store data points from other peers in the network \( D_p^{\text{ext}} \), denoted as the external data of \( p \). The union of internal and external data of peer \( p \) contributes the local data of \( p \). That is, \( D_p = D_p^{\text{int}} \cup D_p^{\text{ext}} \).

During the execution of algorithm, each node \( p \) gradually builds a summarized read of \( D \), by maintaining representatives, denoted as \( R_p = \{r_1^p, r_2^p, ..., r_{k_p}^p\} \). Each representative \( r \in R_p \) is an artificial data item, summarizing a subset \( D_r \) of \( D \). Each data item \( x \) in \( p \) has an associated weight \( w_p(x) \), which is equal to the number of data items that \( x \) is composed of.

The goal of this paper is to make sure that full data set will cluster in a fully decentralized manner, which means each node \( p \) will obtain an accurate clustering result. The representation of the clustering model depends on the particular clustering type. Whenever the particular type of clustering is not necessary, we refer to the method simply by \( F \) which can be seen in Fig. 1. In this paper \( F \) represent partition-based clustering. For this method centroid act as cluster indicators. Fig. 1 provides a graphical view of summarized view of the system model.

![Fig. 1. Graphical view of the system model](Image)

RESULTS AND DISCUSSIONS

Experiments are performed publically using Shuttle Data set which contains 9 attributes and are clustered into 7 clusters. From the data set, a random sample of 1000 and 2000 instances are used in the experiments. To assign the data set \( D \) to nodes, random data assignment is used. In random data assignment each node is assigned data randomly chosen from \( D \). After selecting the data set, algorithm is evaluated in static setting. In order to assess the efficiency of our algorithm in detecting clusters, mainly compare its outcome to that of centralized using the same initial centroids in the central and distributed settings [18].

The implementation module of the paper is divided into 3 modules. They are summarization, weight calculation, final clustering.
The first module is summarization in which each node \( p \) in the network is responsible for deriving accurate representatives for part of data set located near \( D_p^{\text{init}} \). For other parts, it collects representatives. Accordingly a global view of \( D \) is built by node. Two tasks are performed repeatedly and continuously in parallel by each node: i) Representative derivation, called DERIVE and ii) Representative collection, called COLLECT. Both tasks have active and passive threads. An outline of each task is demonstrated in Fig. 2.

In each round of the DERIVE task, each node \( p \) will selects another node \( q \) for a three-way information exchange, as shown in Fig 2. Selection of node is based on peer-sampling service that return a node selected uniformly in a random way among all live nodes in the system [24]. It should first send its internal data to node \( q \). If size of this data is large, it summarize the internal data by an method such as grouping the data using clustering, and from each group sending one data. Node \( p \) then receives from \( q \) data items located in radius \( r \) of each data item in internal data of \( p \), based on a distance function. \( r \) is a user-defined threshold, which is adjustable as \( p \) continues to discover data. In the same manner, it will also send to \( q \) the data in its node that lie within the \( r \) radius of data in internal data of node \( q \). Knowing some data located within radius \( r \) of some internal data item \( d \), node \( p \) can summarize all this data into one representative. This is performed periodically every \( t \) gossip rounds.

![Fig. 2. Overall view of the algorithm tasks](image)

The update Local Data operation shown in the Fig 3 is used to store the received data and to decide whether any internal data can be promoted to a centroid. Only the owner of a data point can decide to promote it to a centroid. To save bandwidth, a node may decide to transfer a representative of several adjacent data objects, in the active thread of DERIVE. In the passive thread of DERIVE however, the amount of transmitted data is bounded and those data objects which are repetitive are passed only once.

To fulfill the COLLECT task, each node \( p \) selects a random node in every \( T \) time units with a purpose to exchange their set of representatives with each other. Both nodes store the full set of representatives. The summarize function, will returns all the representatives given to it as input, as shown in Fig. 4. Initially, each node has only a set of internal data items, \( D_p^{\text{init}} \). Thus, the set of representatives at each node is initialized with all of its data items.

![Fig. 3. Communication sequence of DERIVE tasks](image)

For both COLLECT and DERIVE all the operation performed by node \( p \) is done in active thread of \( p \) and all the operation performed by node \( q \) is done in passive thread of node \( q \) which is shown in Fig. 5. The function remove repetitive representatives includes the weight calculation to accomplish resource constraints which is described in next section.

The second module handles the possibility of intersection between summarized data of different representatives during the merging operation of COLLECT task. The algorithm does not record the entire set due to resource constraints. To address the weight calculation issue, representative points are accompanied by a small estimation field, which allows to approximate the number of actual items that it represents. For this method of distributed computing of a sum of numbers [14] is used.

Node \( p \) should store \( s \) values, per each data item \( d \in D_p \). These values associated with each real data item are deterministically generated by any node. When a representative is first created weight estimators are assigned to it with initial null values. These are updated during merging function. These weight estimators are accompany representative when it is transferred to another node in task COLLECT.

In third module the final clustering algorithm which is based on partition based clustering is executed on the set of representatives in a node. Node \( p \) can execute a weighted version of the clustering algorithm any time it desires, to achieve the final clustering result. K-means [13,15] partition based clustering considers data items to be placed in an \( m \)-dimensional metric space, with an associated distance measure \( d \). It partitions the data set into \( k \) clusters, \( C_1;C_2;...;C_k \). Each cluster \( C_j \) has a centroid \( \mu_j \), which is defined as the average of all data assigned to that cluster. This algorithm tries to minimize the objective function:

\[
\sum_{j=1}^{k} \left( \sum_{d \in C_j} \| d - \mu_j \|^2 \right) = \min
\]
Fig. 4. Communication sequence of COLLECT task

The weighted K-means algorithm is executed on a set of representatives, and its ultimate goal at node p is to compute the mean of data in each cluster. Weighted K-means assumes a positive weight for each data item and uses weighted averaging. The set of representatives in each cluster are identified with the usual nearest centroid method of K-means. Each node uses an arbitrary parameter k with an arbitrary set of initial centroids.

If data items of a typical cluster is in uniform manner the expected value of the representatives computed from data will be equal to centroid otherwise it will deviate from centroid. In such cases, consider subsets of clustered data, each being approximately uniform. The finest region considered to be uniform is the threshold neighbourhood. Representatives in such a neighbourhood share high ratio of data items with each other.

Experimental evaluations show that proposed method can discover the clusters efficiently with scalable transmission cost. Fig 6 shows the number of data items in each cluster for a sample of 1000 data items. The performance metric measured with respect to a node is accuracy which measures the ratio of data items which are located in correct clusters. Fig 7 shows the behavior of algorithm when number of nodes varies. Because the proposed method executes K-means on the representatives instead of data, when compared to actual data labels, accuracy may even surpass the central results for some data sets.

The accuracy values converged to more than 90% for the proposed method. This is estimated on the basis that approximately 80% data belongs to first class which can be seen in Fig 6. When number of nodes increases time taken will be less for processing and result converges more accurately. The interesting point is that, with summarizing internal data before transmission in task DERIVE, the communication overhead can be kept low and independent of size of internal data. Communication overhead of proposed method is due to weight estimators. Nevertheless, these estimators empower the algorithm to preserve its performance even when nodes have repetitive data.

CONCLUSION AND FUTURE WORK

In this work, we discussed some application ranges which benefit from an effective and efficient distributed clustering method. Due to economical, technical and security reasons, it is often not possible to transmit all data from different sites to one central site.
for data clustering. Therefore, we introduced an fully decentralized clustering algorithm which empowers nodes to construct a summarized view of the data to be able to execute a weighted clustering algorithm independently. The implementation of this proposed system demonstrates that it can discover high quality clusters efficiently with scalable transmission cost.

In future work, the problem of outlier detection will be discussed. Weighted version of DBSCAN [16], an density based clustering is used to achieve the outlier detection. To achieve this in our algorithm, task COLLECT should be customized. This causes representatives located outside the actual clusters not to be disseminated in the network, and improves the overall clustering accuracy.

REFERENCES