Abstract

Document clustering (or text clustering) is the application of cluster analysis to textual documents. It has applications in automatic document organization, topic extraction and fast information retrieval or filtering. For text clustering we are using decentralized probabilistic text clustering Algorithm to mine the data, it is an traditional centralized approach by using this approach analyzing massive distributed data. But it is extremely difficult to draw conclusions based on the collective characteristics of disparate data. The goal is to achieve modularity and scalability. Decentralized probabilistic text clustering Algorithm is less scalable in distributed clustering. Distributed Hierarchically Peer-to-Peer Clustering (DHP2PC) Algorithm is scalable and efficient algorithm. In this a subset of the document collection is centrally partitioned into clusters, for which cluster signatures are created. The DHP2PC algorithm finds its roots in a parallel implementation. By using cluster signatures we can mine the Massive distributed data. The algorithm offers probabilistic guarantees for the correctness of each document assignment to a cluster.

Index Terms-- Distributed clustering, text clustering.

1. INTRODUCTION

A recent shift towards distributed data mining (DDM) was sparked when it was realized that analyzing massive distributed data sets using traditional centralized approaches can be intractable. Huge data sets are being collected daily in different fields but it is still extremely difficult to draw conclusions based on the collective characteristics of disparate data. Four main approaches for performing DDM can be identified. A common approach is to bring the data to a central site, then apply centralized data mining on the collected data. Such approach clearly suffers from a huge communication and computation cost to pool and mine the global data. A smarter approach is to perform local mining at each site to produce a local model. All local models can then be transmitted to a central site that combines them into a global model [1]. Ensemble methods also all into this category [10]. While this approach may not scale well with the number of sites, it is a better solution than pooling the data. Another approach is for each site to carefully select a small set of representative data objects and transmit it to a central site, which combines the local representatives into one global representative data set. Data mining can then be carried on the global representative data set [7]. All previous three approaches involve a central site to facilitate the DDM process. A more departing approach does not involve any centralization, and thus belongs to the peer-to-peer (P2P) class of algorithms. In P2P DDM, sites communicate directly with each other to perform the data mining task [4, 2]. Communication in P2P networks can be very costly if care is not taken to localize traffic. In this paper we introduce a hybrid approach for distributed clustering, based on a hierarchically distributed architecture. The goal is to achieve modularity and scalability. The model is called Hierarchically-distributed Peer-to-Peer Clustering (HP2PC). It involves a hierarchy of P2P neighborhoods, in which each neighborhood is responsible for building a clustering based on the data it has access to in a P2P fashion. As we go up the hierarchy, peers merge clusters from lower levels in the hierarchy. At the root one global clustering is computed. Experiments performed on document clustering show that we can achieve comparable results to centralized clustering with better gain in speedup.

The central assumption in DDM is that data is distributed over a number of sites, and that it is desirable to derive, through data mining techniques, a global model that reflects the characteristics of the whole data set. A number of challenges (often
conflicting) arise in DDM, including communication complexity, quality of global model, and privacy of local data. Communication models. The data communicated between nodes in distributed clustering algorithms can be categorized into: (i) models, (ii) representatives, and (iii) raw data. The first case involves calculating local models, such as cluster centroids, e.g., \( \text{P2P k-means} \) [2], that are sent to peers or a central site. In the second case, nodes select a number of representative samples of the local data to be sent to a central site for global model generation, such as the case in the KDEC algorithm [7]. The last case involves exchange of actual data objects to facilitate construction of clusters that exist in certain sites only, such as the case in the collaborative clustering scheme in [6]. Distributed Text Mining. Applications of DDM in the text mining area such as text classification and clustering have received little attention. The work presented by Eisenhardt et al. [4] achieves document clustering using a distributed peer-to-peer network. They use the k-means clustering algorithm, modified to work in a distributed P2P fashion using a probe-and-echo mechanism. A similar system can be found in [9], but the problem is posed from the information retrieval point of view. In this work, a subset of the document collection is centrally partitioned into clusters, for which “cluster signatures” are created. Each cluster is then assigned to a node, and later documents are classified to their respective clusters by comparing their signature with all cluster signatures. A few algorithms were presented representing the state of the art in DDM. Datta et al. [2] described an exact local algorithm for monitoring a k-means clustering, as well as an approximate local k-means clustering algorithm for P2P networks. The P2P k-means algorithm finds its roots in a parallel implementation of k-means proposed by Dillon and Modha [3]. Although the k-means monitoring algorithm does not produce a distributed clustering, it helps a centralized k-means process know when to recompute the clusters by monitoring the distribution of centroids across peers, and triggering a re-clustering if the data distribution significantly changes over time. On the other hand, the P2P k-means algorithm in works by updating the centroids at each peer based on information received from their peers.

This paper is organized as follows. In the next section, we will discuss related work. Section 2 discusses the Distributed K-Means Algorithm. Section 3 illustrates the technique of Probabilistic Clustering. Section 5. The experimental results are shown in Section 6 concludes the paper.

2. EXISTING SYSTEM

In order to provide a distributed implementation of the k-means algorithm, we need a mechanism to diffuse information among the nodes and a mechanism to refine the centroid of a subset of nodes in the network. The average-consensus and max-consensus algorithms proved their effectiveness in composing local observations by means of one-hop communication only. Let \( G = \{V; E; W\} \) be a weighted graph, where \( V \) is a set of \( n \) nodes \( v_1 \ldots v_n \) and \( E \) is the set of edges \( \{(v_i; v_j)\} \) and \( W \) is the set of weights \( w_{ij} \) associated to each edge \( (v_i; v_j) \). A graph is said to be undirected if \( (v_i; v_j) \in E \) whenever \( (v_j; v_i) \in E \), and is said to be directed otherwise. A graph \( G \) is connected if for any \( v_i \) \( \in \) \( V \) there is a path whose endpoints are in \( v_i \) and \( v_j \), without necessarily respecting the orientation of edges. A graph \( G \) is balanced if for each node \( v_i \) \( 2 \) \( V \)

\[
\sum_{j=1}^{n} w_{ij} = \sum_{j=1}^{n} w_{ij}
\]

i.e., the sum of the weights of incoming and outgoing edges coincide. Let the neighborhood \( N_i \) of node \( v_i \) be the set of nodes \( (v_j; v_i) \in E \). Let \( A \) be a set of \( n \) nodes, each associated to a node in the graph \( G \) and described by the following discrete-time single integrator dynamics:

\[
z_i(t + 1) = z_i + u_i(t), \quad z_i(0) = z_{i0}
\]

Assuming that the graph \( G \) is connected and that it is undirected (or directed and balanced) the problem is known to have an asymptotic solution [16], [17] if the following control law is chosen:

\[
u_i(N_i, t) = T \sum_{j \in N_i} (z_j(t) - z_l(t))
\]

Let \( n \) nodes deployed in \( R^2 \) and suppose that each node is able to communicate with the other nodes provided that their distance is less than a communication range which, for the sake of simplicity, is the same for all the nodes. Each node is endowed with a real vector \( x \) \( 2 \) \( R^d \) representing a piece of information or measure. The objective of the distributed k-means algorithm is to partition the nodes in \( k \) clusters minimizing the functional \( D \) specified in eq. (1) (or eq. (4)) via a fully decentralized and distributed approach involving only local interaction among neighbors.
In the following we will neglect, without loss of generality, the procedure to mitigate the risk of obtaining a local minimum by iterating the algorithm several times and selecting the best result. The vector $x_i$ may represent the position of nodes in $\mathbb{R}^2$, resulting in a position clustering, or an observation made by node $i$ of a set of variables of interests, such as temperature, humidity, etc., thus the clustering is a measure clustering.

It is also possible to consider a hybrid clustering problem where vector $x_i$ contains both the position of the node and other sensorial information (eventually scaled to rank the importance of the different information), hence the clustering will divide the nodes into groups depending on both position and measures.

Data: $M, t_{\text{max}}, c_i, N_i$

Result: $c_j (M), \ldots, c_k (M), N_j (M), k_j (M)$

/* Initialization */

$\mu_{ij} = 0$ for all $j=1, \ldots, k$;

$c_{ij}(0) = \left[ \alpha_j \right] T$ for all $j=1, \ldots, k$;

$N_j(0) = N_i$;

$k_j^0(0) = 0$;

/* Loader Election */

$t \leftarrow \max\text{-consensus}(i, N_i, t_{\text{max}})$;

if $t \leftarrow t$ then

node $i$ chooses random Centroids $c_{ij}(0)$ for all $j=1, \ldots, k$;

end

/* main cycle */

for $T=1$ to $M$ do

/* Centroid propagation */

$z_{ij}(T, 0) = \left( c_{ij}(0) T, \ldots, c_{ik}(0) T \right)^T$ if $T = 1$

$z_{ij}(T, 0) = \mu_{ij}(T) \odot c_{ik}(T)$ if $T > 1$

/* Nearest Centroid Choice */

$k_j^T(T) = \arg \min |c_{ih}(T) - x_i|$

$\mu_{ij}(T) = \begin{cases} 1 & \text{if } j = k_j^T(T) \\ 0 & \text{else} \end{cases}$

/* Nearest Centroid Choice Broadcast */

Each node provides $k_j^T(T)$ to the neighbors in $N_i$;

/* Cluster Neighborhood Choice */

Each node select $N_j^T(T) \subseteq N_i$ based on $k_j^T(T)$

For each $j, N_i$;

/* Centroid Refinement */

$c_{ij}(T) = \text{average consensus}(x_i, N_j^T, t_{\text{max}}, \epsilon)$

end

3. PROPOSED APPROACH

In this section a popular method for probabilistic document clustering is that of topic modeling. The idea of topic modeling is to create a probabilistic generative model for the text documents in the corpus. The main approach is to represent a corpus as a function of hidden random variables, the parameters of which are estimated using a particular document collection. The primary assumptions in any topic modeling approach (together with the corresponding random variables) are as follows:

Then documents in the corpus are assumed to have a probability of belonging to one of $k$ topics. Thus, a given document may have a probability of belonging to multiple topics, and this reflects the fact that the same document may contain a multitude of subjects. For a given document $D_i$ and a set of topics $T_1 \ldots T_k$, the probability that the document $D_i$ belongs to the topic $T_j$ is given by $P(T_j | D_i)$. We note that the topics are essentially analogous to clusters, and the value of $P(T_j | D_i)$ provides a probability of cluster membership of the $i$th document to the $j$th cluster. In non-probabilistic clustering methods, the membership of documents to clusters is deterministic in nature, and therefore the clustering is typically a clean partitioning of the document collection. However,
this often creates challenges, when there are overlaps in document subject matter across multiple clusters. The use of a soft cluster membership in terms of probabilities is an elegant solution to this dilemma. In this scenario, the determination of the membership of the documents to clusters is a secondary goal to that of finding the latent topical clusters in the underlying text collection. Therefore, this area of research is referred to as topic modeling, and while it is related to the clustering problem, it is often studied as a distinct area of research from clustering. The value of $P(T_j|D_i)$ is estimated using the topic modeling approach, and is one of the primary outputs of the algorithm. The value of k is one of the inputs to the algorithm and is analogous to the number of clusters. Each topic is associated with a probability vector, which quantifies the probability of the different terms in the lexicon for that topic. Let $t_1...t_d$ be the d terms in the lexicon. Then, for a document that belongs completely to topic $T_j$, the probability that the term $t_l$ occurs in it is given by $P(t_l|T_j)$.

Note that the number of documents is denoted by $n$, topics by $k$ and lexicon size (terms) by $d$. Most topic modeling methods attempt to learn the above parameters using maximum likelihood methods, so that the probabilistic fit to the given corpus of documents is as large as possible. There are two basic methods which are used for topic modeling, which are Probabilistic Latent Semantic Indexing (PLSI) [9] and Latent Dirichlet Allocation (LDA) [10] respectively. In this section, we will focus on the probabilistic latent semantic indexing method. Note that the above set of random variables $P(T_j|D_i)$ and $P(t_l|T_j)$ allow us to model the probability of a term $t_l$ occurring in any document $D_i$. Specifically, the probability $P(t_l|D_i)$ of the term $t_l$ occurring in document $D_i$ can be expressed in terms of above-mentioned parameters as follows:

$$P(t_l|D_i) = \sum_{j=1}^{k} P(t_l|T_j) \cdot P(T_j|D_i)$$

II. Repeat
(Exactly same as original algorithm)
UNTIL $CH_{previous} = 1$

III. Finalize
1. If(is_final_CH = FALSE)  
2. If (($S_{CH} \leftarrow v; v \text{ is a final CH } \neq \emptyset$)  
3. My_cluster_head $\leftarrow$ least_cost($S_{CH}$)  
4. Join_cluster(cluster_head_ID, NodeID)  
5. Else  
6. Find the least cost node among its neighbors as CH  
7. If(cluster_head_ID = NodeID)  
8. Cluster_head_msg(NodeID, final_CH, cost)  
9. Else  
10. Join_cluster(cluster_head_ID, NodeID)

4. EXPERIMENTAL ANALYSIS
5. CONCLUSION

In this paper, we presented an implementation of the DK-Means clustering algorithm and PC2PP using the reciprocal nearest neighbor technique. We also described two speed-up measures that improve the algorithm’s execution time. We gave a probabilistic model using random sample points and assumptions about the RNN conditional probabilities to explain the algorithm’s average-time complexity. Our experimental results suggested the validity of these probabilistic assumptions.

REFERENCES


