International Journal of Humanities and Social Science Invention (IJHSSI)
ISSN (Online): 2319 – 7722, ISSN (Print): 2319 – 7714
www.ijhssi.org ||Volume 9 Issue 6 Ser. III || June 2020 || PP 41-45

Fast Communication-Efficient Over Distributed Data Using Spectral Clustering

Sheela N J [1], Dr K Sundeep kumar [2]
[1] *Student in Department of CS&E SEA College*
[2] *Professor & HOD Department of CS&E SEA College*

ABSTRACT: *The last decades have seen a surge of interests in distributed computing thanks to advances in clustered computing and big data technology. Existing distributed algorithms typically assume all the data are already in one place, and divide the data and conquer on multiple machines. However, it is increasingly often that the data are located at a number of distributed sites, and one wishes to compute over all the data with low communication overhead. For spectral clustering, we propose a novel framework that enables its computation over such distributed data, with "minimal" communications while a major speedup in computation. The loss in accuracy is negligible compared to the non-distributed setting. Our approach allows local parallel computing at where the data are located, thus turns the distributed nature of the data into a blessing; the speedup is most substantial when the data are evenly distributed across sites.*

KEYWORDS: ***Bid Data, Hdfs, Mapping & Reducing, Clustering.***

Date of Submission: 07-06-2020 Date of Acceptance: 22-06-2020

I. INTRODUCTION

SPECTRAL clustering refers to a class of clustering algorithms that work on the engender composition of the Gram matrix formed by the pair wise similarity of data points. It is widely acknowledged as the method of choice for Clustering, due to its typically superior empirical performance, its flexibility in capturing a range of geometries such as nonlinearity and non-convexity, and its nice theoretical properties. For example, a major retail vendor, such as Wal-Mart, has sales data collected at walmart.com, or its Wal-Mart stores, or its warehouse chains—Sam's Club etc. Such data from different sources are distributed as they are owned by different business groups, even all within the same corporation. Indeed there is no one central data center at Wal-Mart, rather the data are either housed at Wal-Mart's Arkansas headquarter, or its e-commerce labs in the San Francisco Bay area, CA, due possibly to historical reason—Wal-Mart was a pioneer of commercial vendors for a large scale adoption of digital technology in the late seventies to early eighties, while started its e-commerce business during the last decade.

There are several challenges in the spectral clustering of data across distributed sites. The data at individual sites may be large. Many existing divide-and- conquer type of algorithms [14, 30] would collect data from distributed sites first, re-distribute the working load and then aggregate results computed at in- dividual sites. The communication overhead will be high. Moreover, the data at individual sites may not have the same distribution. Additionally, the owners of the data at individual sites may not be willing to share either because the data entails value or the data itself may be too sensitive to share (not the focus of this work though). Now the question becomes, can we carry out spectral clustering for data distributed over a number of sites without transmitting large amount data? One possible solution is to carry out spectral clustering at individual sites and then ensemble. However, as the data distribution at individual sites may be very different, and to ensemble needs distributional information from individual sites which is often not easy to derive. Thus, an ensemble type of algorithm will not work, or, at least not in a straightforward way. Another possibility is to modify existing distributed algorithms, such as [14, 30], but that would require support from the computing infrastructure—to enable a close coordination and frequent communication of intermediate results among individual nodes—which is not easy to implement and the solution may not be generally applicable.

II. A FRAMEWORK OF SPECTRAL DISTRUSTED DATA

The idea underlying our approach is the notion of continuity. That is, similar data would play a similar role in learning and inference, including clustering. Such a notion suggests a class of data transformations called distortion min- imizing local (DML) transformation. The idea of DML transformation is to represent the data by a small set of representative points (or codewords). One can think of this as a "small-loss" data compression, or the representative set as a sketch of the full data. Since the representative set resembles the full data, learning based on it is expected to be close to that on the full data. Similar idea was explored in [56, 5] and has been

successfully applied to some computation- intensive algorithms, assuming the data are non-distribted. In all these previous work, DMLs were mainly introduced to address the computational challenge.

2.1 Introduction to spectral clustering

Spectral clustering works on an affinity graph over data points $X_1, ..., X_N$ and seeks to find a "minimal" graph cut. Depending on the choice of the similarity metric and the objective function to optimize, there are a number of variants, in- cluding [45, 41]. Our discussion will be based on normalized cuts [45]. An affinity

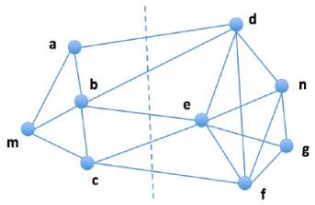

Figure 1: Illustration of a graph cut. graph is defined as a weighted graph G = (V, E , A) where as

V = $\{X_1, ..., X_N\}$ is the set of vertices, E is the edge set, and A = $(a_{ij})_{i,j=1}^{N}$ is the affinity matrix with a_{ij} encoding the similarity between X_i and X_j . Figure 2 is an illustration of graph cut.

2.2 Distortion minimizing local transformation

Attacks A key property that makes DMLs applicable to distributed data is being local. That is, such a data transformation can be done locally, without having to see the full data. Thus DML can be applied at individual distributed nodes sepa- rately. If one can pool together all those codewords, then overall inference or data mining can be easily carried out. Thus, as long as the local data transfor- mations are fine enough, a large class of inference or data mining tools will be able to yield result as good as using the full data. As we are dealing with big volumes of data, a natural requirement for a DML is its computational efficiency while incurring very "little" loss in information. We will briefly describe two concrete implementations of the DML transformation, one by K -means clustering and the other by random projection trees (rpTrees), both proposed in [56].

2.2.1 K-means Cluster

K -means clustering was developed by S. Lloyd in 1957 [38], and remains one of the simplest yet popular clustering algorithms. The goal of K -means clustering is to split data into partitions (clusters) and assign each point to the "nearest" cluster (mean). A cluster mean is the center of mass of all points in a cluster; it is also called cluster centroid or codewords. The algorithm starts with a set of randomly selected cluster centers, then alternates between two steps: 1) assign all points to their nearest cluster centers; 2)recalculate the new cluster centers. The algorithm stops when the change is small according to a cluster quality measure. For a more details about K-means clustering, please refer to the appendix and [19, 18].

2.2.2 Random Project trees

Similarly, To see how the K-D tree [8] can be used for a DML transformation, we first describe how a K-D tree grows. Let the collection of all data at a distributed site correspond to the root node, $D^{(0)}$, of a tree. Now one variable, say v (index of a variable), is selected to split the root node; the root node is split into two child nodes, $D^{(1)}_L$ and $D^{(1)}_R$, according to whether a data points has its v-th coordinate smaller or larger than a cutoff point. For each of $D^{(1)}_L$ and $D^{(1)}_R$, we will follow a similar procedure recursively. This process continues until some stopping criterion is met.

III. SYSTEM ARCHITECTURE

The idea underlying our approach is the notion of continuity. That is, similar data would play a similar role in learning and inference, including clustering. Such a notion suggests a class of data transformations called distortion minimizing local (DML) transformation. The idea of DML transformation is to represent the data by a small set of representative points (or codeword's). One can think of this as a "small-loss" data compression, or the representative set as a sketch of the full data. Since the representative set resembles the full data, learning based on it is expected to be close to that

on the full data. Similar idea was explored in and has been successfully applied to some computation-intensive algorithms, assuming the data are non-distributed.

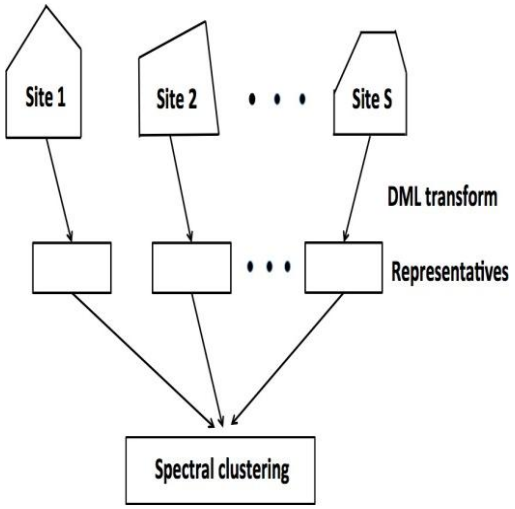

Fig 2 Architecture of spectral clustering over distributed data

In all these previous work, DMLs were mainly introduced to address the computational challenge It is clear that algorithms designed under this framework would eliminate the need of having to transmit large amount of data among distributed nodes—only those codeword's need to be transmitted. As the codeword's are DML-transformed data, data privacy may be ensured since no original data are transmitted. Additionally, as spectral clustering is only performed on the set of codeword's, the overall computation involved will be greatly reduced. Also as the DMLs and the recovering of cluster membership are performed at individual nodes, such computation can be done in parallel. We start by a brief introduction to spectral clustering.

DMLs can be used to enable spectral clustering over distributed data, that is, the data are not stored in one machine but over a number of distributed nodes. Our Frame work for spectral clustering over distributed data is surprisingly simple to implement. It consists of three steps:

1) Apply DML to data at each distributed node
2) Collect code words from all nodes, and carry out spectral clustering on the set of all code words
3) Populate the learned clustering membership by spectral clustering back to each distributed node. The DML-based framework we propose readily facilitates data mining tasks such as spectral clustering over distributed data while eliminating the need of big data transmission.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we report our experimental results. This includes simulation results on synthetic data, and on data from the UC Irvine Machine Learn- ing Repository [15]. We will compare the performance of distributed vs non- distributed (where all the data are assumed to be in one place). The spectral clustering algorithm used is normalized cuts [45], and the Gaussian kernel is used in computing the affinity (or Gram) matrix with the bandwidth chosen via a cross-validatory search in the range (0, 200] (with step size 0.01 within (0,1], and 0.1 over (1,200]) for each data set. All algorithms are implemented in R programming language, and the kmeans() function in R is used for which details are similar as those documented in [56]. The metric for clustering performance is clustering accuracy, which counts the fraction of labels given by a clustering algorithm that agree with the true la- bels (or labels come with the dataset). Let {1, ..., K } denote the set of class labels, and h(.) and h(.) are the true label and the label given by the clustering algorithm, respectively. The clustering accuracy is defined as [16] where I is the indicator function and Π is the set of all permutations on the class labels {1, ..., K }.

While there are dozens of performance metrics around for clus- tering, the clustering accuracy is a good choice for the evaluation of clustering algorithms. This is because the label (or cluster membership) of individual data points is the ultimate goal of clustering, while other clustering metrics are often a surrogate of the cluster membership, and they are used in practice mainly due to the lack of labels. For the evaluation of clustering algorithms, we do have the freedom of using those datasets coming with a label. Indeed the clustering accuracy is commonly used for the evaluation of clustering; see, for example, [4,16]. We also consider the time for computation. The elapsed time is used. It is counted from the moment when the data are loaded into the memory (R running time environment), till the time when we have obtained the cluster label for all data points.

We assume all the distributed nodes run independently, so the longest computation time among all the sites is used (instead of adding them up). As we do not have multiple computers for the experiments, we do not explore the communication time for transmitting representative points and the clustering results. Indeed, for the dataset used in our experiments, such time could be ignored when compared to the computation time, as the number of representative points are all less than 2000. The time reported in this work are produced on a MacBook Air laptop computer, with 1.7GHz Intel Core i7 processor and 8G memory. 13

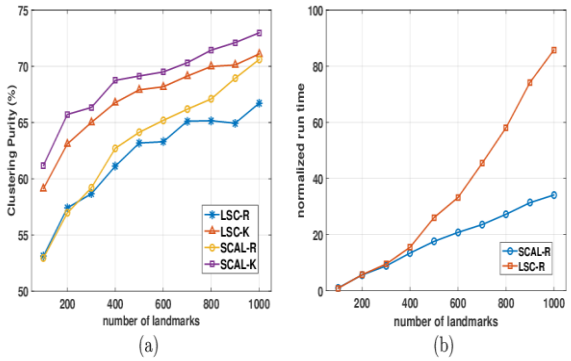

Fig. 3: Performance of different methods versus number of landmarks (a) cluste.

V. CONCLUSION AND FUTURE WORK

We have proposed a novel framework that enables spectral clustering for distributed data, with "minimal" communication overhead while a major speedup in computation. Our approach is statistically sound in the sense that the achieved accuracy is as good as that when all the data are in one place. Our approach achieves computational speedup by deeply compressing the data with DMLs (which also sharply reduce the amount of data transmission) and leveraging existing computing resources for local parallel computing at individual nodes. The speedup in computation, compared to that in a non distributed setting, is expected to scale linearly (with a potential of even faster when the data is large enough) with the number of distributed nodes when the data are evenly distributed across individual sites. Indeed, on all large UC Irvine datasets used in our experiments, our approach achieves a speedup of about 2x when there are two distributed sites. Two concrete implementations of DMLs are explored, including that by K-means clustering and by rpTrees. Both can be computed efficiently, i.e. Computation (nearly) linear in the number of data points. One additional feature of our framework is that, as the transmitted data are not in their original form, data privacy may also be preserved.

Our proposed framework is promising as a general tool for data mining over distributed data. Methods developed under our framework will allow practitioners to use potentially much larger data than previously possible, or to attack problems previously not feasible, due to the lack of data for various reasons, such as the challenges in big data transmission and privacy concerns in data sharing.

REFERENCES

[1]. M. Aledhari, M. D. Pierro, M. Hefeida, and F. Saeed. A Deep LearningBased Data Minimization Algorithm for Fast and Secure Transfer of Big Genomic Datasets. IEEE Transactions on Big Data, PP:1–13, 2018, doi: 10.1109/TBDATA.2018.2805687. 22

[2]. A. C. Arpaci-Dusseau, R. H. Arpaci-Dusseau, D. E. Culler, J. M. Hellerstein, and D. A. Patterson. High-performance sorting on networks of workstations. In ACM SIGMOD International Conference on Management of Data, May 1997.

[3]. D. Arthur and S. Vassilvitskii. How slow is the K-means method? In Proceedings of the Symposium on Computational Geometry, 2006.

[4]. F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. Journal of Machine Learning Research, 7:1963–2001, 2006.

[5]. M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via coresets. In Fortieth ACM Symposium on Theory of Computing (STOC), 2002.

[6]. E. Banijamali and A. Ghodsi. Fast spectral clustering using autoencoders and landmarks. In 14th International Conference on Image Analysis and Recognition, pp. 380–388, 2017.

[7]. H. Battey, J. Fan, H. Liu, J. Lu, and Z. Zhu. Distributed testing and estimation under sparse high dimensional models. The Annals of Statistics, 46(3):1352–1382, 2018.

[8]. J. Bentley. Multidimensional binary search trees used for associative searching. Communications of the ACM, 18(9):509–517, 1975.

[9]. E. Brunskill, T. Kollar, and N. Roy. Topological mapping using spectral clustering and classification. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, pp. 3491–3496, October 2007.

[10]. D. Cai, X. He, Z. Li, W. Ma, and J. Wen. Hierarchical clustering of www image search results using visual, textual and link information. In Proceedings of the 12th Annual ACM International Conference on Multimedia, pp. 952–959, 2004.

[11]. E.-C. Chang, S.-C. Huang, H.-H. Wu, and C.-F. Lo. A case study of applying spectral clustering technique in the value analysis of an outfitter's customer database. In Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, 2007.

[12]. F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A Distributed Storage System for Structured Data. In Seventh Symposium on Operating System Design and Implementation (OSDI),

November 2006.
[13]. M. Chen, S. Mao, and Y. Liu. Big data: A survey. Mobile Networks and Applications, 19:171–209, 2014.
[14]. W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang. Parallel spectral clustering in distributed systems. IEEE Transactions on Pattern Analyses and Machine Intelligence, 33(3):568–586, 2011.
[15]. X. Chen and D. Cai. Large scale spectral clustering with landmark-based representation. In AAAI, 2011. 23
[16]. X. Chen and M. Xie. A split-and-conquer approach for analysis of extraordinarily large data. Statistica Sinica, 24:1655–1684, 2014.
[17]. F. Chung. Spectral Graph Theory. American Mathematical Society, 1997.
[18]. S. Dasgupta. Learning mixtures of Gaussians. In Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS), 1999.
[19]. S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In Fortieth ACM Symposium on Theory of Computing (STOC), 2008.
[20]. J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters . In Sixth Symposium on Operating System Design and Implementation (OSDI), December 2004.
[21]. S. Dolev, P. Florissi, E. Gudes, S. Sharma, and I. Singer. A Survey on Geographically Distributed Big-Data Processing using MapReduce. IEEE Transactions on Big Data, 3:79–90, 2017.
[22]. B. Efron. Bootstrap methods: another look at the Jacknife. Annals of Statistics, 7(1):1–28, 1979.
[23]. C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nystr¨om method. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(2):214–225, 2004.
[24]. A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, and P. Gauthier. Cluster-based scalable network services. In 16th ACM Symposium on Operating Systems Principles, pp. 78–91, 1997.
[25]. S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google file system. In 19th ACM Symposium on Operating Systems Principles, pp. 29–43, 2003.

Sheela N J, et. al. "Fast Communication-Efficient Over Distributed Data Using Spectral Clustering." *International Journal of Humanities and Social Science Invention (IJHSSI),* vol. 09(6), 2020, pp 41-45. Journal DOI- 10.35629/7722