Kunling Zeng

Review of the Literature Outline

EAP 508 P02

11/9/2016

**Comparing K-Means Variants for web-scale Clustering**

1. **Introduction**

Clustering issues come from many areas such as data mining, vector quantization, image classification and pattern recognition. These fields are closely related to many practical areas like astronomical, medical and biological industry. Clustering spilt the given datasets to several groups, where the data in the same group are more similar to each other than the data in different groups [1]. One of the new trend of clustering research is adapting clustering to web-scale data [2].

Web-scale clustering is much more difficult than normal clustering. Due to the tremendous data emerging from internet in everyday life and the very high dimension of these data, even one of the most well known clustering solutions, K-Means, becomes unbearably slow [2]. Reality applications like user-facing application requires the latency of results within seconds even optimized K-Means can not meet when clustering web-scale data [3]. Many K-Means variants have been proposed in terms of web-scale clustering to address this issue [2,3,4,5,6]. There are literatures [7] review of K-Means algorithm, but there is no article compare and summarize these K-Means variants for web-scale clustering.

This project reviews some web-scale orientated K-Means variants, tries to summarize and compare these solutions for web-scale clustering problem theoretically. We focus on K-Means because it is one of the most well-known solutions for clustering in current literature and have been extended to many fields [8]. As accommodating K-Means algorithm for web-scale clustering is a new trend of research [2], hopefully, this project will offer some reference for future research of web-scale clustering. And it surely will be a good guidance to novices in this field to help them have a clearer mind about different versions of K-Means algorithm specifically for web-scale clustering since there are so many variants of K-Means adapting for different areas. My driving question is, what are the different K-Means algorithm variants for web-scale clustering and their unique modification to traditional K-Means, how might study and compare these methods help to explore the potential solution for web-scale clustering.

1. **Clustering and K-Means**

Clustering issues comes from many areas, it is very helpful for underlying data structure, natural classification and data(vector) compression [8]. Clustering divides the given dataset to several clusters, the data points inside each group are more similar to the one in other cluster. Web-scale clustering arising with the development of Internet. The success of famous IT companies like Google, FaceBook and Amazon etc. facilitate the increase of high dimensional data in the Internet in orders of magnitude. These data are such a treasure if we could extract the information behind them, clustering is the tool to simplify these data for further research and use.

K-Means is one of the most well known solution for clustering problem, it has been studies for decades but is still very popular because of its simplicity and efficiency. In fact, K-Means is ranked as one of the top ten algorithms in data mining [9]. Basically, “K” stands for k clusters, “Means” means the average of the data points in each group (i.e. use the means to be the centers). K-means problem is to find k clusters of a given datasets of n d-dimensional data points, such that the sum of the square distance of each data point to its nearest center is minimum. Traditional K-Means basically contains two main steps:

1) assign the data points to its nearest center (divide data points to different groups)

2) recalculate the centers according to the data points of each group according the last assignment (recalculate the means), then go back to 1) until the centers is convergent during different iterations.

Above iteration is also called Lloyd’s iteration [10]. K-Means has been proved to converge to local optimum [11]. It usually initializes with k centers picked up from the dataset randomly, and its clustering result is very sensitive to initialization [12]. The complexity of K-Means is O(nk).

Although K-Means is favored because of its simplicity and efficiency, the K-Means problem is actually NP-hard. Due to the tremendous high-dimension data points (millions of data points of hundred dimensions), traditional K-Means algorithm become infeasible facing web-scale clustering.

1. **Improving K-Means**

Hundreds of literature have been proposed to improve the traditional K-Means [1,2,3,4,5,6,13,14,15]. Although K-Means is very widely studied and used, it does suffer some disadvantages such as it is very sensitive to initialization [12], it converges to local optimum [11], does not offer quality guarantee [5] and its complexity can’t handle web-scale data [2,3,4,6]. Literature devote to improving the performance and accommodating it to different area always focus on either studying its efficiency or clustering quality [2]. Furthermore, [4] summarized that the methods work on refining the efficiency of K-Means into two categories: one using kd-tree [13,15] to facilitate the distance computation process, the other exploit the subsample methods [3]. But actually, there are also many literature [5,12,14] devoted to the initialization of K-Means as we mentioned before that the clustering result of K-Means is very sensitive to its initialization. And often a good initialization could improve both quality and efficiency [5,14].

Comes to web-scale clustering, there are some novel ways to accommodate K-Means for it [2,3,4,6]. Mini-batch K-Means explores the subsample method [3], wand-k-means combines algorithm WANK with K-Means [4], Bk-means exploits a hash table build on binary coding data, boost K-Means takes advantage of the objective function it derived. These K-Means variants will be our focus in the rest of this paper. And we also include k-means++ in our discussion even it is not web-orientated variant, but obvious the seeding strategy it proposed is very successful and make it efficient enough to handle some degree of web-scale clustering.

For each K-Means variant discussed following, we explain its modification to traditional K-Means, then we show its performance derived from corresponding paper, followed by the discussion about it.

1. **Web-Scale orientated k-means variants**
	1. **k-means ++**

K-Means always suffers from its sensitivity to initialization. [12] discussed this problem and compared four initialization methods, [14] proposed a refinement algorithm for the given starting centers. Usually, traditional k-means starts with the starting centers picked up from datasets randomly. K-Means ++ [5] put its effort on a new initialization method and turns out to be very successful.

The only difference between K-Means++ and traditional K-Means is the initialization. It introduced a weighting strategy of the data points called “D2 weighting” during the initialization process. K-Means++ proposed the new way to select to the beginning centers instead of traditional one choosing randomly, the initialization is divided to three step:

1. first choose one center randomly from the dataset.
2. Choose the new center from the dataset with the probability $\frac{D(x)^{2}}{\sum\_{x\in X}^{}D(x)^{2}}$

(D(x) stands for the shortest distance from a data point to the closest center we have already chosen).

1. repeat 2) until we have k centers. [5]

K-means++ is compared with k-means in 4 datasets, 2 synthesized and 2 real word datasets. For the relatively large real world one(n=494019, d=35) which we are interested in, k-means ++ achieved 10 to 1000 magnitude better result than k-means, and it is up to 70% faster. [5]

The reason that proposed picking strategy works is because of the “D2 weighting” strategy, which makes sure the chosen centers are well separated, which leads to better performance and higher efficiency. However, the origin of its success, D2 weighting, is also where its limitation comes from. The computation cost(computing the “D2 weighting”) during the initialization process is costly. The high efficiency only comes when the k is relatively small compared to n, but still it is useful since that is usually the case in web-scale clustering. And the seeding strategy could also be used for the initialization of other K-Means variants.

* 1. **Mini-batch K-Means**

Traditional K-Means is also the “standard full batch algorithm”, which is too slow for large datasets [3]. Subsample methods are one major group of improving the efficiency of K-Means [4]. [11] proposed a stochastic gradient descent(SGD) strategy which is much faster but of low quality because it suffers from the stochastic noisy of subsample.

Small batch used in mini-batch K-Means alleviate the distortion rate but still retain the fast convergence speed [3]. The main modification of mini-batch K-Means to traditional one are: 1) small group subsample of size b to alleviate the stochastic noise; 2) per-center learning rate [11] to facilitate convergence.

Mini-batch k-means is tested on standard RCV1 collection of documents, compared to full batch algorithm and SGD. b=1000, clustering result was evaluated using objective function on the held-out test set. Its convergence speed is many orders of magnitude faster than L’s batch algorithm and have much better result than SGD. [3]

Though the efficiency improvement of mini-batch k-means is very considerable, and the quality of its result is much better than SGD, but compares to K-Means, it’s quality just very close to k-means. And since K-Means is always known because of its simplicity and efficiency, but offer no quality guarantee [5], the quality of mini-batch should be further improved.

And we notice that mini-batch K-Means is a subsample method, which is predicable won’t perform well when the cluster size is small [4]. Because only if the points in each cluster is big enough, then the random picking up of the samples could make sense, otherwise the samples selected from different clusters could be clustered into the same cluster since the total points of the subsample is much more small than the original data.

* 1. **Wand-K-Means**

The costliest computation step in K-Means is the assignment step. In wand-k-means, the authors viewed assignment procedure from a different perspective, using the “centers picking points” rather then “points picking center”, which reduce tremendously reduce the computation cost of assignment step. The main modifications of wand-k-means are:

1) using centers as query, which in converse to classic k-means.

2) the combination with WANK, which facilitate the procedure of retrieving top-l.

3) sparsify the centers to efficiently use the NNS structure to speed up the algorithm. [4]

Wand-k-means is compared with k-means on two real world datasets (974,738 and 933,763 documents separately). Through the experiments, wand-K-Means shows a much faster (an order of magnitude) convergence speed while maintain the same (within 0.5%) clustering quality as traditional K-means. [4]

We could observe the other characteristics as following:

1)the different value of l(top l be retrieved) doesn’t affect the performance of proposed algorithm, since the number of points examined is much bigger then l. And smaller l actually bring higher similarity since the it exclude outlier more completely.

2)higher p(sparsification parameter) outperform no sparsification at all, but the very high p makes it slow down, so maybe 200 is a good choice if the requirement for quality is not very restrict.

3) adjusts l could control outlier, but even for l=1, the outlier is less then 1%, which is acceptable.

Even the increase of efficiency is substantial, wand-K-Means has the same problem as mini-batch K-Means, the very close to but less than traditional k-means clustering outcome, which itself is quiet no stable.

* 1. **Bk-Means**

Bk-Means attempted using binary code to represent the dataset points and the centers, which makes the memory requirement affordable for web-scale images clustering. And the assignment process is facilitated by a hash table of the centers, exploiting the computation advantages of binary coding. [3]

Bk-Means shows fair performance compares to traditional k-means, just 1 to 2 percent(s) less than the k-means, but the clustering speed is of 60-70 times of traditional k-means, which makes it more suitable for web-scale clustering. KDk-means and k-medoids are also compared, benefit from the binary code and hash table, Bk-means and k-medoids outperform KDk-means in terms both speed and accuracy, and Bk-means outperforms k-medoids. [3]

the improvement in clustering speed of Bk-Means is very fantastic for web-scale clustering, and the application of this algorithm is of reality usage. But since the k-means always suffer from bad performance, and the clustering result of proposed algorithm is even less the traditional one (1-2%), the accuracy of this algorithm needs improving. [3]

* 1. **Boost K-Means**

K-Means clustering are driven by objective function. In other word, it tries to optimize the objective function so to maximize the similarity of data points in the same group (minimize the sum of square distance from each data point to its closest center). For traditional K-Means, the objective function could be present like:

$$E=min\sum\_{x\in X}^{}||x-C(x)||^{2}$$

Where C(x) stands for the closest center for x.

Two objective function was derived in Boost K-Means from the one of traditional K-Means, and finally simplified to the same form:

$Max. I=\sum\_{r=1}^{k}\frac{D\_{r}'D\_{r}}{n\_{r}}$ [2]

Where $D\_{r}$ is the sum of the data points in the same group $S\_{r}$ and $n\_{r}$ is the number of these data points. This function uses the sum of all the inner points in one cluster as the measurement of similarity, makes the computation for optimum easier. All the advantages of the proposed algorithm are driven by this objective function:

1)Firstly, the objective function is feasible for all the I2 distance space, which is compared to the cosine similarity, used in wand-k-means [4].

2) Secondly, the objective function makes the computation for reassignment easier. Instead of calculating the distance between each sample to every centers to decide which center is nearest, it calculate the sum of all vectors in the same cluster.

3)besides, proposed algorithm uses a stochastic assignment strategy at the beginning to avoid the initialization which affects the performance of k-means a lot. [2]

Boost K-Means is fully compared with other variants of K-Means, K-Means++, mini-batch K-Means, LVQ, and of course the traditional one. And it shows a better performance in terms of quality and efficiency. Also, boost K-Means itself compared under the configuration of non initialization, random initialization and K-Means++ initialization, there is no improvement of boost K-Means under K-Means++ initialization over random initialization, which further prove that boost K-Means is well constructed. [2]

One worthy notice is that the experiment shows the poor performance of mini-batch algorithm [3], which confirmed our conservative discussion about its clustering quality. It could also extend to justify our discussion about the bad result of K-M variants which has the result very close but less the traditional one.

1. **Discussion**

Mini-batch K-Means, wand-K-Means and Bk-means are of good efficiency, which makes them more suitable for web-scale clustering. But all these algorithms just tried to maintain the same clustering quality of traditional K-Means, which itself doesn’t offer any guarantee about clustering result, turns out to be of poor clustering outcome. This conclusion is confirmed by [2] which we have discussed and extended in previous section.

K-Means++ offers a very successful seeding strategy, but the cost of choosing starting centers become unaffordable when k is very big. The seeding strategy could also be separately used for other clustering, leading to faster convergence speed and better result. But there is one exception, for boost K-Means, initialization with K-Means makes no difference then random initialization.

Boost K-Means turns to be the most successful web-scale orientated K-Means variant, due to its simple assignment process benefits from the objective function derived.

Our suggestion for future research:

1. There is a lot space of quality improvement for Mini-batch K-Means, wand-K-Means and Bk-Means.
2. Future research of variant of K-Means for web-scale clustering, even focus on improving the efficiency, should also pay some attention to improve the quality, better to make it at least not worse than tradition K-Means.
3. Our analysis and comparison are all theoretically and based on the experiment of the literature reviewed, future could implement empirical research to verify our conclusion.
4. **Conclusion**

Accommodating K-Means to web-scale clustering is a new trend of research, there are some variants of web-scale orientated K-means variant, but no literature review of them. This project reviews some of the K-Means variants for web-scale clustering, analyzes and compares them. We hope this paper could be a good guidance to novices in this field and even offer some reference for future research of web-scale clustering.

**REFERENCES:**

[1] Faber V. Clustering and the continuous k-means algorithm [J]. Los Alamos Science, 1994, 22(138144.21).

[2] Zhao W L, Deng C H, Ngo C W. Boost K-Means [J]. arXiv preprint arXiv:1610.02483, 2016.

[3] Sculley D. Web-scale k-means clustering [C]//Proceedings of the 19th international conference on World wide web. ACM, 2010: 1177-1178.

[4] Broder A, Garcia-Pueyo L, Josifovski V, et al. Scalable K-Means by ranked retrieval [C]//Proceedings of the 7th ACM international conference on Web search and data mining. ACM, 2014: 233-242.

[5] Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding [C]//Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007: 1027-1035.

[6] Gong Y, Pawlowski M, Yang F, et al. Web scale photo hash clustering on a single machine [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 19-27.

[7] Jain A K, Dubes R C. Algorithms for clustering data [M]. Prentice-Hall, Inc., 1988.

[8] Jain A K. Data clustering: 50 years beyond K-means [J]. Pattern recognition letters, 2010, 31(8): 651-666.

[9] Xindong Wu and Vipin Kumar. 2009. The Top Ten Algorithms in Data Mining (1st ed.). Chapman & Hall/CRC.

[10] Lloyd S. Least squares quantization in PCM [J]. IEEE transactions on information theory, 1982, 28(2): 129-137.

[11] Bottou L, Bengio Y. Convergence properties of the k-means algorithms [J]. Advances in neural information processing systems, 1995: 585-592.

[12] J.M Peña, J.A Lozano, P Larrañaga, An empirical comparison of four initialization methods for the K-Means algorithm, Pattern Recognition Letters, Volume 20, Issue 10, October 1999, Pages 1027-1040

[13] Kanungo T, Mount D M, Netanyahu N S, et al. An efficient k-means clustering algorithm: Analysis and implementation [J]. IEEE transactions on pattern analysis and machine intelligence, 2002, 24(7): 881-892.

[14] Bradley P S, Fayyad U M. Refining Initial Points for K-Means Clustering [C]//ICML. 1998, 98: 91-99.

[15] Alsabti K, Ranka S, Singh V. An efficient k-means clustering algorithm [J]. 1997.