A Progressive Sampling based Approach to Reduce Sampling Time

Nandita Bangera
Dept of CSE
BMSCE
Bengaluru, India
nanditamanohar@gmail.com

Dr. Kayarvizhy N.
Dept. of CSE
BMSCE
Bengaluru, India
kayarvizhy@gmail.com

***Abstract*— Analytics plays vital role in Data Science. It involves finding trends and patterns from the huge repository of data. Scanning huge amount of data consumes lot of time, which can be reduced by sampling. In this paper we have demonstrated effectiveness of Progressive sampling wherein the sample size is gradually increased till it reaches a desired accuracy. By applying an algorithm based on Rademacher average to mine frequent datasets using Progressive sampling, we have shown that the runtime and the sampling time is considerably reduced as compared with static sampling.**

Keywords—sampling, static sampling, progressive sampling, runtime, data mining

I. INTRODUCTION

There is a rapidly growing availability of huge datasets from diverse sources [1] driven by easy access to Internet & mobile computing like web, social media networking, blogs, forums and sensors. The vast repository of structured and unstructured data poses a huge challenge in terms of capturing, cleaning, integration, processing, searching, sharing, transferring and analysis. Analysis plays a major role in evaluation of data or information by breaking it into its modules or partitioning it to understand their inter-relationships, patterns thus enabling researchers, analysts and business users to have a better insights for appropriate decision making.

The growing data and real-time nature of some applications means that standard statistical techniques to process the data is highly inadequate due to computational and performance requirements. Therefore some scaling down of dataset is required by way of sampling.

Sampling plays a major role in Data Science considering the outsized volume of data. Considering the type of data and domain specific sampling methods are implemented to reduce the time and memory overhead. There are diverse methods of sampling like static, random, strata, cluster, multistage, systematic random sampling and progressive sampling. To extract useful and required information we perform querying. Querying large amount of data requires a considerable amount of time which can delay the response time. Several studies have been conducted to provide online interactive query handling techniques which can provide partial results of desired accuracy [15], [16], [17], [18]. Large size of data in a database may gradually reduce the speed of the query retrieval process thus hampering the analytical speed and accuracy. Also, the user response time is increased which may further have a greater impact on the credentials of the organization. Most of the queries whether it is for the data analyst or business users or any other commercial applications we do not expect the accurate results. For e.g. How much percentage of rainfall was deficit in this particular year? In most of the cases the analysis is performed to investigate the facts and patterns and conclude with a understanding of the system. So, to find an approximate answer to a query we need not scan the whole dataset. Since most queries do not require accurate answers, we can use sampling.

II. BACKGROUND

A. Data Sampling

Dataset can be reduced by sampling. Sampling creates a very small subset of the database and then uses this smaller representative dataset to answer the query. The selection of data is based on various sampling techniques which retains the order and relationships between various tuples. Sampling involves breaking the huge datasets into small samples. Each partition of the samples should derive meaningful patterns for analysis. The most easiest technique in sampling is the Random sampling where the sample set is drawn in random, but has probability of higher error rate.

B. Application Domains

In the field of machine learning (ML) we implement Prediction model which uses the concept of training data and test data is used wherein we train the data to learn the characteristics of the output and test the data with the learned data. Training too much of data leads to valuable processing time hence increasing the computational cost. Progressive sampling (PS) thus can avoid scanning the whole training set since it converges to a certain accuracy limit. Thus progressive sampling can be carried out with a particular approach for approximately accurate results with desirable error rate.

In the domain of Data mining the Association rule mining involves two phases. In the first phase frequent itemsets (FI) are generated and in the second phase the association rules are

978-1-7281-0630-4/18/$31.00 ©2019 IEEE

generated. To mine huge datasets progressive sampling can be used to considerably reduce the time needed to mine by increasing the samples progressively till the desired accuracy.

Sampling is the most effective tool to reduce the dataset but it comes with a price of accuracy reduction. Progressive sampling tends to reduce the time required for the sampling by quick convergence and maintaining the accuracy. But determining the sample size and achieving the desired accuracy always remains a big challenge.

III. RELATED WORK

Various sampling methods were implemented depending on the nature of particular application [14]. Static sampling techniques was used for application reliability of communication network in the simulation environment [12].

A work was carried out in the field of datamining for generating association rules using Progressive Sampling (PS) [7]. It was a refined method of PS by using a equivalence, class which is able to converge to the desired sample size very quickly and accurately. Another related work on PS [8] in which the sampling estimates are improved along with the sample size. The method was used for different applications where retrospective sampling over multiple period was required. Considering the time factor, PS was carried out in the field of neonatal intensive care [4], where the training data was reduced to one third of its original size maintaining the accuracy of the performance. An improved PS algorithm [2] was implemented in the field of data mining to generate association rules extending the work carried out in [7]. The frequent itemsets were mined using Apriori algorithm. The support level was determined by scanning the midpoint itemset to the remaining set of records. If the support level of the midpoint itemset is greater than the user specified support then the chosen sample size is progressively increased. This procedure is repeated until an optimal sample size is obtained and then association rules are mined from the optimal sample. Finally the support of midpoint item is analysed with different percentage of dataset. In yet another work [13], improvements in memory and time complexity has been obtained by partitioning the itemsets instead of midpoint itemset in the negative border itemsets. A classic work of a combination of PS and classifier wrapper [9] was implemented on the training data sets. The wrapped PS algorithm takes input as a dataset which is represented as a vector of feature value and labelled with one set of possible output labels. This dataset is used as a base for further sampling of training and test dataset. The wrapping algorithm tends to optimize the algorithmic parameter based on PS by searching all possible combination of parameter setting on all training data. It progressively increases the training set and check on the convergence by decreasing the amount of parameter combinations. The PS based Bayesian Optimization method is an efficient and automatic method [10] for selecting both machine learning (ML) algorithm and hyper parameter values which reduces the search time, classification error rate and standard deviation of error rate. They have implemented Bayesian optimization based on sequential model to find the combination of ML and feature selection techniques. Most recent research [6] was the combination of PS and ML in which the Batch Mode Uncertainty Sampling from the field of Active Learning (semi-supervised ML algorithm) was used to progressively grow the sample by fetching the most significant datapoints to be included in the sample.

IV. OVERVIEW OF SAMPLING METHODS

A. Static Sampling

Static Sampling (SS) involves extracting samples of fixed or equal quantity of data .The increments is calculated at specific sample points. Static sampling does not require any other information apart from what the dataset provided [11]. Random sampling is the most common form of Static sampling.

Algorithm 1: Static Sampling

1. Consider an initial sample size n0 of S from D
2. **repeat**
3. Schedule sample size increase Δn either arithmetic ($\Delta n_i = \lambda$) or geometric schedule ($\Delta n_i = n_i$-1)
4. **until** new dataset **M** reaches convergence
5. **return M**

B. Progressive Sampling

Progressive Sampling (PS) starts with a small data sample from the full dataset and use progressively larger samples until the model accuracy cannot increase substantially. PS techniques attempts to efficiently maximize model accuracy by using growing the sample size.

The challenges in PS are : generating the right sample size (scheduling) and efficiently testing the accuracy convergence (stopping condition).

Initial sample size and scheduling is of utmost importance since if we increase the sample it may lead increasingly larger sample and a smaller sample may increase the cost of computation and time. The two most commonly used techniques as in the literature is the Arithmetic sampling and the Geometric Sampling. In the AS framework an initial sample of $S0$ of fixed size is selected from the data set. Subsequently a set number of data points $N\theta$ is consecutively added to this initial sample such that $Si = S0 + i * N\theta$. A arithmetically generated progressive samples where $S0 = 500$ and $N\theta = 100$ is 500, 600, 700, … , 10000. A disadvantage of this technique is that if $N\theta$ is too small, a very large number of iterations will be needed to achieve. convergence. In GS framework the initial sample $S0$ is grown geometrically by multiples of a predefined number θ such that $Si = \theta\ i * S0$ (2) A simple example of a geometrically generated progressive samples where $S0 = 500$ and $\theta = 2$, is { 500, 2000, 4000, … . ,16000 } With GS, a major drawback of the technique is its tendency to overshoot, since the sample size rapidly grows.

Sufficient Research work has been conducted to improvise the progressive sampling techniques by considering various

parameters such as starting sample size, memory and time complexity [5].

Algorithm 2: Progressive Sampling

6. Compute the schedule of the sample by dividing the samples into k sets {n0, n1, n2, … nk}
7. n ← n0
8. **repeat**
9. new sample n` ← n + schedule
10. develop model **M** from n`
11. **until M** reaches stopping condition
12. **return M**

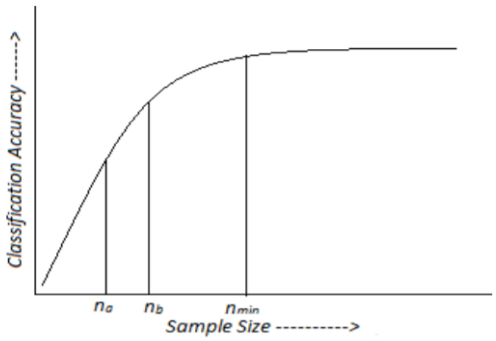

Fig. 1. Progressive sampling curve

V. IMPLEMENTATION METHODOLOGY

Our primary aim in this study was to check the suitability of sampling algorithm for use with progressive analytics & visualization system. This system requires intermediate results of query for varying confidence levels.

We use PS to retrieve a top k frequent itemsets (FIs) from random samples of a transactional dataset. The retrieved sample will have data almost approximate to the actual dataset with almost all frequent itemset included whose frequency has higher threshold.

A. Progessive Sampling for FI mining

Input: Dataset D, parameters $\theta, \varepsilon, \delta \in (0, 1)$, a sampling schedule $(|S_i|)_{i \geq 1}$
Output: An (ε, δ)-approximation to $FI(D, I, \theta)$

Algorithm 3: Modified Progressive Sampling

1. i ← 0, S0 ← ∅, |S0| ← 0
2. **repeat**
3. i ← i + 1
4. S∗ ←random_sample(D, |Si|-|Si-1|)
5. Si ← Si-1 ∪ S∗
6. **until** fstop()
7. **return** FI(Si, I, θ - ε/2)

Function random_sample(D, n) returns n random samples from D. Function fstop() returns 1 if stopping condition is not met. Threshold frequency $\theta \in (0, 1)$, $\varepsilon \in (0, 1)$ is the accuracy parameter and $\delta \in (0, 1)$ is the confidence parameter.

B. Stopping Condition

The algorithm implements progressive sampling with a stopping condition based on a statistical concept known as Rademacher average [3]. The computed bounds can obtain an approximate itemset with a single scan thus avoiding to mine each sample.

The main aim of this study is to prove that progressive sampling can be much faster than the static sampling as it avoids the scanning the whole database and also maintains the characteristics of the real datasets. The algorithm stops at smaller samples. To represent a big data analysis the dataset is replicated so that we get a broader insights of the actual facts and figures. The frequency distribution and the transaction length of the item sets is preserved. The accuracy of the algorithm in terms of precision, accuracy and error in frequency estimation is remarkably good.

VI. EXPERIMENTAL EVALUATION

A. Implementation Details

The code is implemented in ANSI C/C++ standards and compiled using the GCC compiler 8.0. It can be easily ported to any of the major OS

B. Datasets

We ran the algorithm on real dataset available at: http://www.cs.uef.fi/~whamalai/datasets.html http://fimi.uantwerpen.be/data/ http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. The datasets are suitable for frequent itemsets and data mining. For our experimental evaluation we have used sampling time and runtime as the attributes to determine and varying confidence parameter for analysis.

C. Results

Table 1 contains the comparative data for Static and Progressive Sampling performance when run on different datasets.

The table has the sample size, sampling time and runtime of the data used in the experimental run. The runtime and sampling time of progressive sampling is far less as compared to the static sampling thus concluding that progressive sampling can considerably reduce the response time with respect to query handling and analysis.

Fig.2 and 3 shows the static and progressive sampling time & processing time for the datasets mentioned in the table below, plotted on the X-axis. As can been seen the process time is small fraction of sampling time.

TABLE I. COMPARING STATIC AND PROGRESSIVE SAMPLING

S. No.	Dataset Name	Static Sampling			Progressive Sampling		
		Sample Size	*Sampling Time (ms)*	*Runtime (ms)*	*Sample Size*	*Sampling Time (ms)*	*Runtime (ms)*
1	KDDCUP99.TXT	183026	2899	4348	106451	671	751
2	BOGPLANTS.TXT	243026	4481	4499	27918	177	251
3	FORESTS.TXT	733026	25660	25853	59915	450	641
4	RETAI.DAT	603026	11906	12247	704860	3054	11920
5	CHESS.DAT	403026	9960	10004	285258	1123	2058

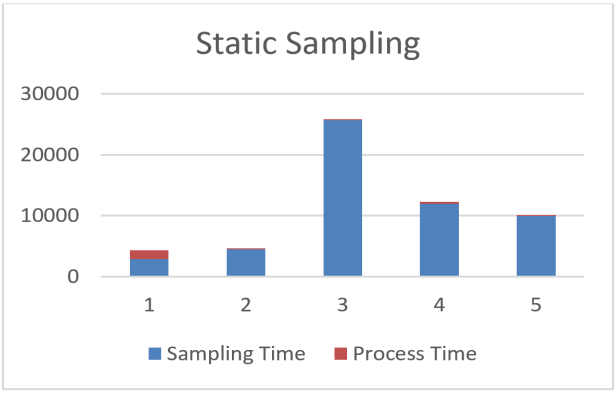

Fig. 2. Static Sampling timing

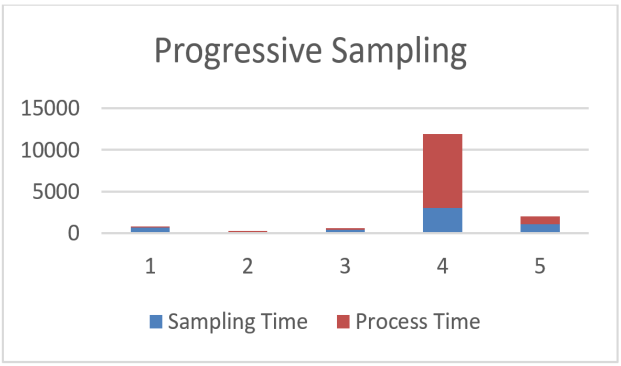

Fig. 3. Progressive Sampling timing

Fig. 4 compares the total runtime of static and progressive sampling for all the datasets mentioned earlier. In all cases PS outperforms SS.

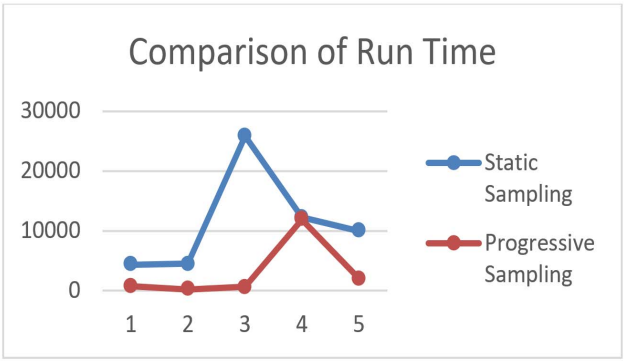

Fig. 4. Progressive Sampling Timing

Fig 5 is the comparative runtime performance results of the parametric study for different values of confidence parameter. For higher confidence levels (1 - δ), a greater number of samples are needed and as can been seen the sampling time increases. In case of CHESS dataset, there is small aberration observed where sampling time actually increased for lower confidence level. This could be a edge case with the algorithm and we need to investigate this further. However, the difference is not excessive.

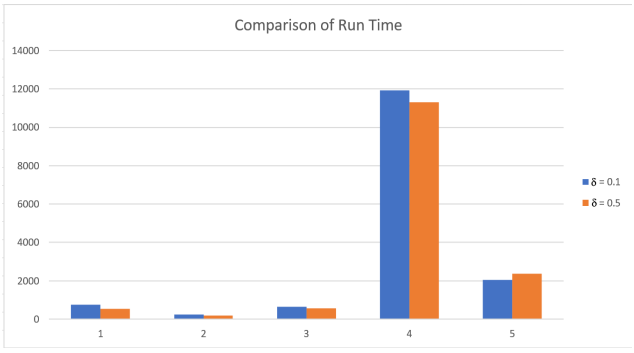

Fig. 5. Progressive Sampling runtimes with varying confidence parameter

VII. CONCLUSION AND FUTURE WORK

In this paper we presented an overview of Progressive Sampling and Static Sampling along with some experimental results which proved that Progressive Sampling indeed reduces the time factor which is essential for all the processes involved in data science domain most importantly in the field of Data Analysis. We have also surveyed all the progressive sampling related work in different fields of database and data science. We have compared the results of progressive and static sampling using different datasets of frequent items mining and concluded that the progressive sampling can considerably reduce the time required for sampling, runtime by maintaining the performance and accuracy.

Though this sampling method has its own drawbacks in terms of choice of learning algorithms, the stopping condition and the accuracy level, it has outgrown all the disadvantages in most of the applications and proved to be the most efficient sampling method with desirable accuracy.

We intend to incorporate the progressive algorithm in a progressive visualization system allowing data analyst to gain quick insight into large data without having to wait for processing of entire database records.

REFERENCES

[1] IBM, Big Data Analytics. https://www.ibm.com/analytics/hadoop/big-data-analytics

[2] S. S. Thakur, Shalini Zanzote Ninori, "An Improved Progressive Sampling based Approach for Association Rule Mining International Journal of Computer Applications" (0975 –8887), Volume 165 – No.7, May 2017.

[3] Matteo Riondato, Eli Upfal, "Mining Frequent Itemsets through Progressive Sampling with Rademacher Averages".

[4] François Portet, Feng Gao, Jim Hunter and René Quiniou, "Reduction of Large Training Set by Guided Progressive Sampling: Application to Neonatal Intensive Care Data".

[5] Foster Provost, David Jensen and Tim Oates, "Efficient Progressive Sampling", ACM 1999.

[6] Amr ElRafey and Janusz Wojtusiak, "A Active Learning and Progressive Sampling Algorithm, International Journal of Machine Learning and Computing", Vol. 8, No. 5, October 2018.

[7] S.Parthasarathy, "Efficient progressive sampling for association rules", IEEE International Conference on Data Mining, 2002.

[8] P.A. De los Santos, R.J. Burke, J.M. Tien, "Progressive random sampling: A multiperiod estimation technique with applications IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews", Volume: 30, Issue: 4, Nov 2000.

[9] Antal van den Bosch, "Wrapped Progressive Sampling for optimizing Learning Algorithm Parameters,Netherlands Organisation for Scientific Research".

[10] Zeng X, Luo G, "Progressive sampling Based Bayesian optimization for Efficient and Automatic Machine Learning Model Selection", Springer 2017.

[11] Mohamed Aounallah, Sebastien Quirion and Guy W. Mineau, "Distributed Data Mining vs. Sampling Techniques: A Comparison".

[12] The applicability of static sampling techniques for measuring the application realiability of the communication network in simulation environment.

[13] Venkatapathy Umarani Muthusamy Punithavalli- Analysis of the progressive sampling-based approach using real life datasets https://link.springer.com/journal/13537.

[14] Peter J. Haas, "Data-Stream Sampling: Basic Techniques and Results".

[15] Joseph M.Hellersten, Peter,J.Hans, Helen,J.Wang, "Online Aggregation: ACM SIGMOD International Conference on Management of Data", May 1997.

[16] Nikolay Laptev,Kai Zeng,Carlo Zaniolo, "Early Accurate Results for Advanced analytics on Map reduce", VLDB 2012.

[17] Vijayshankar Raman, Joseph Hellerstein, "Partial Results for Online Query Processing.ACM Sigmod", June 2002.

[18] Joseph Hellerstein, Ron Anvur, Vijayshankar Raman, "Infomix under control: online query processing, Data Mining and Knowledge Discovery", 2000.