Philipp Rosch, Rainer Gemulla, Wolfgang Lehner

Designing Random Sample Synopses with Outliers

Erstveröffentlichung in / First published in:

IEEE 24th International Conference on Data Engineering. Cancun, 07.-12.04.2008. IEEE, S. 1400-1402. ISBN 978-1-4244-1836-7

DOI: https://doi.org/10.1109/ICDE.2008.4497569

Final edited form was published in "IEEE 24th International Conference on Data Engineering. Cancun, 2008. IEEE, S. 1400-1402. ISBN 978-1-4244-1836-7
https://doi.org/10.1109/ICDE.2008.4497569

Designing Random Sample Synopses with Outliers

Philipp Rösch, Rainer Gemulla, Wolfgang Lehner

Technische Universität Dresden / Faculty of Computer Science / Database Technology Group
01062 Dresden, Germany
{philipp.roesch, rainer.gemulla, wolfgang.lehner}@tu-dresden.de

***Abstract*— Random sampling is one of the most widely used means to build synopses of large datasets because random samples can be used for a wide range of analytical tasks. Unfortunately, the quality of the estimates derived from a sample is negatively affected by the presence of "outliers" in the data. In this paper, we show how to circumvent this shortcoming by constructing *outlier-aware sample synopses*. Our approach extends the well-known outlier indexing scheme to multiple aggregation columns.**

I. Introduction

Random sampling is an important tool for many large-scale applications. For instance, it has been used for approximate query processing, query optimization, stream processing, and data analysis. One of its main advantages is its ability to deal with a broad range of queries while providing probabilistic error bounds at the same time. For aggregation queries, however, estimates derived from random samples are sensitive to extreme values in the data. For instance, imagine a hypothetical company with a revenue of $1 billion in 2007 and suppose that there is a single purchase order of $100 million. This order is represented as a single tuple in the database, but it constitutes 10% of the total revenue. Tuples like this one are often called *outliers*; they are different from the rest of the data and contribute significantly to the aggregate. To avoid high estimation errors, outliers should be well represented by a synopsis of the underlying dataset.

Chaudhuri et al. proposed a sampling scheme called *outlier indexing* [1], which stores outliers separately and samples from the remaining part of the dataset only. It has been shown that this technique may significantly reduce the estimation error. In this paper (a short version of [2]), we extend outlier indexing to multiple aggregates. The main challenges are 1) to detect outliers efficiently and 2) to balance the available space between the sample and the outlier part of the synopsis. The difficulty arises from the fact that outliers for one aggregate are not necessarily outliers for the other aggregates. To address this problem, we introduce and make use of error measures which quantify the estimation error over all the aggregates simultaneously. The goal is then to find the synopsis with the lowest error measure. Since the computation of the optimum synopsis is computationally prohibitive, we leverage heuristics to prune the search space and also consider greedy algorithms for picking outliers.

II. Error Measures

Given a dataset $R = \{r_1, r_2, \ldots\}$ and a set of attributes $A_1, \ldots, A_l$, we quantify how well a synopsis Ψ of R can estimate aggregates on the A_j. We focus on sums and averages but our techniques may also apply to other aggregates. We consider synopses which consist of a set $O \subset R$ of outliers and a random sample S from $R \setminus O$—the efficient choice of O is the main contribution of this paper. To estimate an aggregate of the dataset, we first estimate the aggregate for $R \setminus O$ using the sample S and then combine this estimate with the aggregate on the set O of outliers. Since outliers are stored in their entirety, the only source of estimation errors is the sample part of the synopsis.

Denote by r_{ij} the value of attribute A_j of the ith tuple in R. Let $L_j(R) = \sum_{r_i \in R} r_{ij}$ be the linear sum and $Q_j(R) = \sum_{r_i \in R} r_{ij}^2$ be the quadratic sum of these values. Then, $\mu_j(R) = L_j(R)/|R|$ is the average value of A_j and

$$RSD_j(R) = \frac{\sqrt{\frac{Q_j(R)}{|R|} - \left(\frac{L_j(R)}{|R|}\right)^2}}{|\mu_j(R)|}$$

denotes its relative standard deviation. An unbiased estimate of $\mu_j(R)$ is given by $\hat{\mu}_j(R) = L_j(S)/|S|$. For a sample of size n, the relative standard error of the estimate equals [3]

$$RSE_{\hat{\mu}_j}(R, n) = RSD_j(R)\sqrt{\frac{1}{n} - \frac{1}{|R|}}.$$

A value significantly smaller than 1 indicates low-error estimates. Since $\hat{\mu}_j(R)$ is unbiased, the RSE is proportional to the root mean square error (RMSE) but has the advantage of being normalized and unitless. It can therefore be used to compare the estimation error across multiple columns.

An error measure $\mathcal{M}$ is a function which takes a set of outliers as its input and computes a numerical value which quantifies the estimation error when this specific set of outliers is chosen in the synopsis. A synopsis $\Psi = (S, O)$ is said to be *more precise with respect to a measure* $\mathcal{M}$ than another synopsis $\Psi' = (S', O')$ if $\mathcal{M}(O) < \mathcal{M}(O')$. Our error measures are based on the RSE as defined above. Suppose that we have space to store M items and fix a set O of items from R. We require that $|O| < M$ so that the sample contains at least 1 item. The RSE of the estimate on attribute A_j is then given by $RSE_{\hat{\mu}_j}(O) = RSE_{\hat{\mu}_j}(R \setminus O, M - |O|)$. This mirrors the challenge of deciding on the number of outliers to extract from the dataset: The selection of an outlier decreases the RSD but also reduces the space available for the sample of the remaining dataset.

If there is only a single attribute, we can directly use $RSE_{\hat{\mu}_j}(O)$ as an error measure. For multiple attributes, the

Final edited form was published in "IEEE 24th International Conference on Data Engineering. Cancun, 2008. IEEE, S. 1400-1402. ISBN 978-1-4244-1836-7
https://doi.org/10.1109/ICDE.2008.4497569

situation gets more complex because the estimates of each aggregate may have a different RSE. Perhaps the most intuitive way of combining the individual RSEs is to take their maximum:

$$\mathcal{M}_{MAX}(O) = \max_{j=1}^{l} RSE_{\hat{\mu}_j}(O).$$

Here, only the column with the highest RSD has an influence on the overall measure. The problem with this measure is that other columns—which might also benefit from outlier indexing—are simply ignored. Thus, we may try to minimize the average RSE of the estimates:

$$\mathcal{M}_{AVG}(O) = \frac{1}{l}\sum_{j=1}^{l} RSE_{\hat{\mu}_j}(O).$$

Using this measure, the influence of each column is proportional to its RSD. Going one step further, our last measure is independent from the absolute values of the RSEs:

$$\mathcal{M}_{GEO}(O) = \sqrt[l]{\prod_{j=1}^{l} RSE_{\hat{\mu}_j}(O)} \propto \sqrt[l]{\prod_{j=1}^{l} \frac{RSE_{\hat{\mu}_j}(O)}{RSE_{\hat{\mu}_j}(\emptyset)}}.$$

Here, $RSE_{\hat{\mu}_j}(\emptyset)$ is the RSE achieved by simple random sampling and $\propto$ indicates proportionality. Intuitively, this measure quantifies the improvement in the RSE compared to simple random sampling. When this measure is minimized, estimates on columns with outliers are improved significantly no matter whether or not they already had a low RSE.

III. Synopsis Computation

In this section, we show how to compute the set of outliers which minimize a measure $\mathcal{M}$. Our algorithm decides on both the number and the composition of the outliers. It is structured into 4 phases. In phase 1 (initialization), we build a random sample S_R of size M and construct a set C of outlier candidates. The candidate set prunes the search space for the next phases. As soon as phase 1 is complete, no further access to the base data is required. In phase 2 (selection), we construct sets $O_0, \ldots, O_{M-1}$ of outliers so that $O_k \subseteq C$ and $|O_k| = k$, $0 \leq k < M$. The set O_k can be seen as the optimum (or close to optimum) choice of outliers from C when exactly k outliers are chosen. In phase 3 (decision), we decide on one of the O_i as the final set of outliers. In phase 4 (finalization), we construct the synopsis Ψ using the outlier set and the random sample S_R. We now describe each of the phases in more detail.

Phase 1 (Initialization). In the first phase, outlier candidates are computed. In the single-column case, the optimum set of outliers solely consists of tuples from the lower and/or upper end of A's value range [1]. Thus, we may restrict C to the tuples with the $M-1$ smallest and $M-1$ largest values of A. A straightforward extension to multiple columns is to include the tuples with the $M-1$ smallest and largest values of *each* column into C. However, there is no guarantee that the optimum set of outliers is a subset of C. The reason lies in the correlation between the columns. To understand this, consider a relation R with two columns consisting of tuples $(4,20)$, $10\text{x}(5,5)$, $(19,19)$ and $(20,4)$. For $M=2$, the set C would contain items $(4,20)$ and $(20,4)$. However, the optimum choice for O_1 is $(19,19) \notin C$. Aside from non-optimality, this choice of C is prohibitive in terms of memory consumption: C can grow as large as $2l(M-1)$ tuples, which is infeasible for the large amount of columns found in practical scenarios. We therefore propose a heuristic approach which selects exactly $M-1$ tuples from the base relation, *no matter how many columns are subject to optimization*. In general, the selected tuples satisfy one or both of the following properties: (1) the tuple has extreme values in *some* of the columns or (2) the tuple has values which differ from the average—though not necessarily extremely—in *most* of the columns. Our approach is to assign a *weight* to each tuple $r_i \in R$ and to then select the tuples with the $M-1$ largest weights. The weight is chosen in such a way that it is large whenever (1) or (2) holds and small otherwise. Note that the absolute values of the weights are unimportant; their purpose is to order the tuples in the relation with respect to their outlier status. We make use of a priority queue to determine the tuples with the $M-1$ largest weights, so that phase 1 requires $O(|R| \log M)$ time and $O(M)$ space.

The assignment of good weights to the individual tuples is crucial for our algorithm. Clearly, the weights should correlate with the measure we try to minimize. Inspired by the computation of the RSD, we might weight the tuples according to their relative distance from the mean:

$$\mathcal{W}_{DistMean}(r_i) = \sum_{j=1}^{l} \left(\frac{r_{ij} - \mu_j(R)}{\mu_j(R)}\right)^2.$$

If $\mathcal{W}_{DistMean}$ is large for some tuple r_i, the tuple almost certainly satisfies (1) or (2). We also consider two alternative weights:

$$\mathcal{W}_{SumRSD}(r_i) = -\sum_{j=1}^{l} RSD_j(R \setminus \{r_i\})$$

$$\mathcal{W}_{ProdRSD}(r_i) = -\prod_{j=1}^{l} RSD_j(R \setminus \{r_i\}).$$

These weights are tailored to $\mathcal{M}_{AVG}$ and $\mathcal{M}_{GEO}$, respectively. They equal the sum/product of the RSDs for each column if tuple r_i were removed. We want to select the tuples with the largest decrease in RSD as candidates; for this reason, a minus sign has been added to each weight. Note that the time complexity of weight computation is proportional to the number of attributes but independent of $|R|$ because $RSD_j(R \setminus \{r_i\})$ can be computed from r_i, $L_j(R)$ and $Q_j(R)$. In our experiments, we found that $\mathcal{M}_{MAX}$ performs best with $\mathcal{W}_{DistMean}$, while $\mathcal{M}_{AVG}$ and $\mathcal{M}_{GEO}$ produce the highest quality synopses with $\mathcal{W}_{SumRSD}$ and $\mathcal{W}_{ProdRSD}$, respectively.

Phase 2 (Selection). Let $O_k \subseteq C$ be the optimum set of outliers (with respect to C) with $|O_k| = k$. The output of this phase consists of the sets $O_0, \ldots, O_{M-1}$. We will also discuss approximate methods; in this case, the O_k are only approximate.

Final edited form was published in "IEEE 24th International Conference on Data Engineering. Cancun, 2008. IEEE, S. 1400-1402. ISBN 978-1-4244-1836-7
https://doi.org/10.1109/ICDE.2008.4497569

We first consider the single-column case, in which the set C contains the $M-1$ smallest and largest tuples. A naive approach is to exhaustively try out all possible combinations to find the one which minimizes the RSE. This approach requires $O(M^2)$ time to compute $O_0, \ldots, O_{M-1}$. One might hope that the O_k can be computed incrementally. However, the relationship $O_{k-1} \subset O_k$ does not hold in general. To understand this, consider the relation $R = \{1, 2, 3, 5, 1000, 1000, 1000\}$ and set $M = 6$. The optimum choices for the O_k are: $O_1 = \{1000\}$, $O_2 = \{1000, 1000\}$, $O_3 = \{1000, 1000, 1000\}$ but for $O_4 = \{1, 2, 3, 5\}$. Clearly, $O_3 \not\subset O_4$. In this pathological example, the variance among the outliers is low and the synopsis size is high. In practice, however, the relationship almost always holds, so that we can speed up outlier computation by selecting outliers one by one. The resulting greedy algorithm starts with $O_0 = \emptyset$. To compute O_k from O_{k-1}, the tuple with the highest distance to the mean (of the remaining dataset $R \setminus O_{k-1}$) is added. In addition to being far more efficient, the greedy algorithm almost always produced the same result as the exhaustive algorithm in our experiments.

We now consider the multi-column case, in which the set of outlier candidates consists of the $M-1$ tuples with the largest weights. It is computationally prohibitive to try out all combinations for computing the O_k. In fact, there are $\binom{M-1}{k}$ possible combinations for each O_k, which sums to 2^{M-1} combinations in total. Therefore, we again select the outliers in a greedy way. Since the tuples are already weighted, we select outliers in the order of their weights. Though this procedure is fairly simple, it produces good results and is very efficient [2].

Phase 3 (Decision). The third phase decides on the optimum value k_{opt} for k. We compute $\mathcal{M}(O_0), \ldots, \mathcal{M}(O_{M-1})$ and choose k_{opt} so that $\mathcal{M}(O_{k_{opt}})$ is minimized. To increase efficiency, an implementation might precompute the $\mathcal{M}(O_k)$ during the selection phase.

Phase 4 (Finalization). In the last phase, we construct $\Psi = (S, O)$. We set $O = O_{k_{opt}}$ and compute S from S_R by subsampling $S_R \setminus O$ down to size $M - k_{opt}$.

IV. Case Study

To investigate the efficiency and effectiveness of our approach in comparison to the naive method of exhaustive enumeration, we conducted an experiment on a very small synthetic dataset (100 tuples, 2 columns); see [2] for a detailed description of the dataset. Figure 1a plots the total execution time for synopsis sizes ranging from 1 to 50 tuples. As can be seen, exhaustive enumeration is infeasible because the execution time is exponential in M and explodes with an increasing synopsis size. In contrast, the execution time of our heuristic algorithms depends only logarithmically on M so that it is significantly faster. In Figure 1b, we plot the value of the $\mathcal{M}_{AVG}$ measure for the resulting synopses. As can be seen, both the exhaustive and the heuristic algorithm outperform SRS. Also, our heuristic approach produces synopses close to the optimum.

We now briefly report results of a case study on a large real-world dataset. The dataset contains market research data

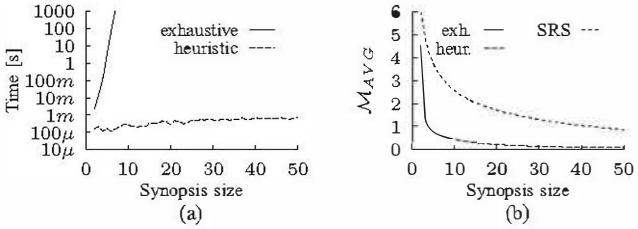

Fig. 1: Efficiency and effectiveness

TABLE I: RSE of average estimate

	Price	Stock	Purchase	Sales
SRS	0.133%	0.356%	1.577%	1.606%
$\mathcal{M}_{MAX}$	0.185%	0.210%	0.191%	0.147%
$\mathcal{M}_{AVG}$	0.169%	0.213%	0.191%	0.149%
$\mathcal{M}_{GEO}$	0.168%	0.214%	0.191%	0.149%
Lower bound	0.123%	0.148%	0.136%	0.125%

and consists of $4,383,694$ tuples with 5 columns. Each row represents an item available in a specific year (`YEAR`), its price (`PRICE`), the amount of stocked items (`STOCK`), and the number of sales (`SALES`) and purchases (`PURCHASE`) of the item. We evaluated the estimation error for the average of the `PRICE`, `STOCK`, `SALES` and `PURCHASE` column. We created a synopsis of size 10% of the dataset using simple random sampling (SRS) and multi-column outlier indexing (MCOI) with varying measures. Table I shows the RSE for each of the 4 columns. We also give a theoretical lower bound which is achieved by optimizing *only* the respective column; we cannot give a tighter bound because exhaustive enumeration is infeasible for such a large dataset. As can be seen, MCOI significantly reduces the estimation error in high-error columns. On low-error columns—which do not contain outliers—MCOI might induce additional estimation error. In any case, the resulting RSE is close to the lower bound. In this experiment all the measures show a similar performance, see [2] for examples where this is not the case.

V. Conclusions

The presence of extreme values in the data has a negative impact on the quality of estimates derived from random samples. We introduced several error measures, paying respect to the fact that synopses are often used to estimate a variety of aggregates. We extended the well-known outlier indexing scheme so that it produces synopses optimized for multiple aggregates simultaneously. We introduced a heuristic and greedy algorithm which handles the resulting combinatorial explosion of possible synopses and is by several orders of magnitude faster than the exhaustive approach.

References

[1] S. Chaudhuri, G. Das, M. Datar, and R. M. V. Narasayya, "Overcoming Limitations of Sampling for Aggregation Queries," in *ICDE*, 2001.

[2] P. Rösch, R. Gemulla, and W. Lehner, "Designing random sample synopses with outliers," Tech. Rep., 2007. [Online]. Available: http://wwwdb.inf.tu-dresden.de/publications

[3] W. Cochran, *Sampling Techniques*, 3rd ed., ser. Wiley Series in Probability & Mathematical Statistics. John Wiley & Sons, 1977.