Improved Cardinality Estimation using Entity Resolution in Crowdsourced Data

Indulakshmi K R  
M. Tech. Student  
Department of Computer Science & Engineering  
NCERC Thrissur, India

Preethymol B  
Assistant Professor  
Department of Computer Science & Engineering  
NCERC Thrissur, India

Silja Varghese  
Assistant Professor  
Department of Computer Science & Engineering  
NCERC Thrissur, India

Abstract

Crowdsourcing platforms adopt the new Labour as a Service model and allow for easy distribution of small tasks to a large number of workers. Crowdsourced systems introduce the open world model of databases. In the open world model, the database is considered to be incomplete and data needs to be collected in real time. For enumeration queries, the cardinality estimation of crowd collected data determines the query progress monitoring. A statistical tool is proposed to estimate the cardinality which enables users to judge the query completeness of crowdsourced data. Moreover, the crowdsourced database contains the records representing the same real world entity. A hybrid human-machine approach is proposed in which machines first, coarse pass over all the collected data, and crowd workers verifies only the most likely matching pairs. The entity resolution merges the duplicate records and hence can improve the cardinality estimation.

Keywords: Cardinality Estimation, Entity Resolution, Hybrid Human Machine, Crowdsourced Data

I. INTRODUCTION

Crowdsourcing was introduced which allows programmers to encompass “human computation” as a fundamental unit in algorithms that cannot be fully programmed. Humans can accomplish many tasks with ease that remains difficult or impractical for computers. Human computation makes use of human abilities to accomplish computation tasks that are complex for computers to process. Crowdsourcing is a Web based collaboration model in which tasks are outsourced to an anonymous workforce [3]. Jeff Howe coined the term “crowdsourcing” in 2006 [4].

Fig. 1 illustrates a typical crowdsourcing scenario. To crowdsource a task, the task owner, also known as requester, submits the task to a crowdsourcing platform. Workers are people who can perform the task, can choose to work on it and devise solutions. Workers then submit these accomplished tasks to the requester via the crowdsourcing platform. The requester evaluates the posted contributions’ quality and might pay workers for their contributions which have been accepted. This payment can be monetary, material, psychological, and so on [5].

In the Open world model of databases, all of the tuples relevant to a query are not assumed to be in the database apriori, hence needs to be retrieved in real time. For these types of queries human input is required for query processing. Crowds can be incorporated for set enumeration tasks. Estimating the cardinality of crowd based data is a major issue. The cardinality estimation for enumeration queries is analogous to species estimation in biology. Hence efficient species estimation techniques are adopted for the cardinality estimation [1].

A real world entity can be represented by distinct records in a database. Entity Resolution (ER) is the process of identifying and merging records that correspond to the same real-world entity. Often, people are better than a machine at deciding if records correspond to the same entity. A hybrid human machine approach based entity resolution can be to identify and merge duplicate records [2].
II. CARDINALITY ESTIMATION METHODS

The number of rows in a query result is referred to as cardinality. Cardinality estimation is a fundamental issue that has been studied for several decades in database community. Most of the system considers query plan for cardinality estimation. They require cardinality estimates in order to obtain cost estimates for various query execution plans.

A. Cardinality Estimation in Deco

H. Park et al. [6] suggested a top down method is used for cardinality estimation in Deco. Deco is a declarative crowdsourcing system that acquires records on demand. While executing a query in Deco, it stores the crowdsourced data as it arrives. The cardinality estimation in Deco is based on the estimated final database state, hence the algorithm used simultaneously estimates cardinality and end-state. The database end state depends on the entire query plan. So the database cardinality must be complete.

B. Selectivity Estimation in Qurk

A. Marcus et al. [7] proposed Qurk, a crowd worker-aware database system. Qurk uses an asynchronous query executor and a crowd aware optimizer which considers monetary cost and result accuracy in addition to time. Qurk can issue HITs that extract, order, filter, and join complex data types, such as images and large text blobs. Qurk may retain outcomes of prior tasks for reuse or fitting classifiers with the aim of reducing cost, using an underlying storage engine. The adaptive approach to query processing is used in Qurk since selectivity of operators cannot be predicted a priori. Qurk utilizes selectivity estimation to order filters and joins in the query plan. In selectivity estimation, for a given expression or predicate, the number of results that will satisfy the expression is estimated. Selectivity estimators can be used to estimate answers to COUNT, SUM, and AVERAGE aggregate queries with GROUP BY clauses.

C. Cardinality Estimation using Sample Views

Larson, et al. [8] proposed a novel cardinality estimation method that makes use of random sampling and materialized view technology and gives accurate estimates even in diverse situations. It can be queries containing complex predicates, correlation among columns, or predicates containing user-defined functions. Here sample views are used as a means to augment traditional cardinality estimation methods. The cardinality estimates for SPJG-expressions, that is, select-project-join expressions with at most one group-by on top can be computed using sample views. The basic idea of sample view technique is to do a sequential sampling. The process is to find only as many rows of the sample as is needed to compute a sufficiently accurate estimate. Then probe queries are generated for selectivity estimation and distinct value estimation by sequential sampling and sample views is refreshed to assure quality.

D. Black Box Approach In Databases

Tanu Malik et al [9] suggested a black-box approach to estimating query cardinality that has no knowledge of query execution plans and data distribution, yet provides accurate estimates. It does so by grouping queries into syntactic families and learning the cardinality distribution of that group directly from points in a high-dimensional input space constructed from the query attributes, operators, function arguments, aggregates, and constants. The black-box approach offers an alternative to estimating query cardinalities that is compact, efficient, and accurate. It does not require knowledge of the distribution of data and avoids inaccuracies from modeling assumptions, such as the conditional independence of attributes. Several emerging applications, such as proxy-caching, data-centric grids, and federated query processing, need to estimate query cardinalities, but have neither access to data distributions nor do they require sub-query cardinalities. Hence, the black-box approach suits them well.

III. PROPOSED WORK

For crowd sourced enumeration queries, query progress monitoring becomes difficult due to non-uniformities in the appearance of crowd sourced data and the method of people working in crowdsourcing systems. To address these issues for crowdsourced enumeration queries, statistical tools are developed that enable users and systems developers to reason about query completeness. To evaluate progress as answers are appearing, the system needs an estimate of the result set’s cardinality in order to calculate the percentage complete.

A. Cardinality Estimation

Requesters post HITs which are enumeration queries in a crowdsourcing platform and workers answer the queries in return for a payment. Getting answers from workers is similar to obtaining samples from some underlying distribution which has an unknown size N, each answer resembles to one sample from the item distribution.

The problem is analogous to species estimation problem and can be rephrased as follows:

The set of HITs received from AMT is a sample of size n obtained from a population in which items can be from N different classes, numbered 1=c=N (N, unknown), c is the number of unique classes (species) seen in the sample. The number of elements in the sample is ni that belong to class i, with1≤i≤N. Of course some ni=0 because they have not been found in the sample.

Let pi be the probability that an item from class i is chosen by a worker, the cardinality estimation is on the basis of “frequencies of frequencies”, (also called f-statistic).The f-statistic can be used to estimate the number of unobserved items for
non-parametric algorithms. It represents the relative frequency of observed classes in the sample. Let $f_j$ be the number of classes that have exactly $j$ members in the sample. The number of unseen classes is denoted by $f_0$. By predicting $f_0$, cardinality can be estimated.

1) **Chao92 Estimator**

The Chao92 estimator is proposed to estimate the cardinality. It makes use of the sample coverage to predict the cardinality $N$. The sample coverage is the sum of probabilities of the observed classes. It can also be calculated by equation 1.

$$\hat{c} = 1 - \frac{f_1}{n}$$  \hspace{1cm} (1)

$$\hat{N}_{chao92} = \frac{c}{\hat{c}} + \frac{n(1 - \hat{c})}{\hat{c}} \gamma^2$$ \hspace{1cm} (2)

The sample coverage is entirely based on percentage of $f_i$ answers. The estimate $\hat{N}_{chao92}$ is the total set size as per the information in the initial sample.

2) **Streaker Tolerant Estimator**

The streaker-tolerant estimator identifies the $f_i$ outlier workers. The contribution of each worker $i$ to the singleton answers is calculated. The $f_i$ count of worker $i$ is then compared to the mean of sample $\bar{x}_i$ and the sample standard deviation $\hat{\sigma}_i$.

$$\bar{x}_i = \sum_{j \neq i} \frac{f_j(1)}{W-1}$$ \hspace{1cm} (3)

$$\hat{\sigma}_i = \sqrt{\sum_{j \neq i} \frac{(f_j(1) - 1)^2}{W-2}}$$ \hspace{1cm} (4)

$$\hat{N}_{crowd} = \frac{cn}{n - \sum \min(f_i(1), 2\hat{\sigma}_i + \bar{x}_i)}$$ \hspace{1cm} (5)

The estimate $\hat{N}_{crowd}$ gives the total set size estimation. It removes the $f_i$ outliers. Hence a rare item does not affect the cardinality estimation.

3) **Shen Estimator**

The Shen estimator is based on the assumption that the estimate of the number of unobserved elements $\hat{f}_0$ and that the unobserved elements have equal relative abundances. The goal is to predict $\hat{N}_{shen}$, the number of new species that will be discovered in a second sample of size $m$, given the information of the initial sample.

$$\hat{N}_{shen} = \hat{f}_0 \left[ 1 - \left( 1 - \frac{1 - \hat{c}}{\hat{f}_0} \right)^m \right]$$ \hspace{1cm} (6)

### B. Entity Resolution

Entity resolution in database systems is the task of finding different records that refer to the same entity. Entity resolution is particularly important when cleaning data or when integrating data from multiple sources. In such scenarios, it is not uncommon for records that are not exactly identical to refer to the same real-world entity. The goal of entity resolution is to find all duplicate records.

1) **Hybrid Human Machine Entity Resolution**

People are often better than algorithms at detecting when different terms actually refer to the same entity. A hybrid human-machine approach has the potential to combine the efficiency of machine-based approaches with the answer quality that can be obtained from people.

![Fig. 2: Hybrid Human Machine Workflow](image)
A hybrid human-machine workflow is shown in Fig. 2. The workflow first uses machine-based techniques to compute for each pair the likelihood that they refer to the same entity. A pre-defined similarity metric, such as Jaccard similarity, is computed for each pair of records. Jaccard similarity over two sets is defined as the size of the set intersection divided by the size of the set union. It uses similarity-based techniques which require a similarity function and a threshold. The similarity function takes a pair of records as input, and outputs a similarity value. The more similar the two records, the higher the output value. The similarity of all pairs of records is computed. If a pair of records has a similarity value no smaller than the specified threshold, then they are considered to refer to the same entity.

Then, only those pairs whose likelihood exceeds a specified threshold are sent to the crowd. Given the set of pairs to be sent to the crowd, the next step is to generate HITs so that people can check them for matches. HIT Generation is a key component of the workflow. Finally, generated HITs are sent to the crowd for processing and the answers are collected.

IV. ANALYSIS

Table 1 shows the result obtained when answers for crowdsourced enumeration task is considered. For a sample size \( n = 35 \), the number of distinct values is represented by ‘\( c \)’ and the number of answers that appears only once in the answer set is represented by singleton, ‘\( f_1 \)’. The table shows the various estimator values before and after entity resolution.

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>c</th>
<th>( f_1 )</th>
<th>Chao92</th>
<th>Streaker</th>
<th>Shen</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Before Entity Resolution</strong></td>
<td>35</td>
<td>20</td>
<td>11</td>
<td>29</td>
<td>26</td>
<td>9</td>
</tr>
<tr>
<td><strong>After Entity Resolution</strong></td>
<td>35</td>
<td>18</td>
<td>7</td>
<td>23</td>
<td>20</td>
<td>4</td>
</tr>
</tbody>
</table>

Fig. 3: Cardinality Estimation before and after entity resolution

Fig. 3: represents the three cardinality estimation before and after entity resolution. The three estimators estimate three different cardinalities. The percentage of reduction in the cardinality depends on the distinct answer set and singleton answer set. If more answers in these sets are resolved by entity resolution more will be the change in cardinality prediction.

The Chao92 estimator overestimates the cardinality when rare items arrive quickly. The rare items appear due to streaker impact. Streakers are workers who provide more singleton answers. Hence a streaker tolerant estimator is used to ameliorate the problem of over estimation in Chao92 estimator. The streaker –tolerant estimator reduces the streaker impact and hence it gives a better estimate of total set size. Shen estimator is used to estimate the cardinality for an increased sample size ‘\( m \)’. It uses the chao92 estimate to estimate the unseen answer ‘\( f_0 \)’. But Shen estimator underestimates the unobserved answer set size for greater values of \( m \). Hence streaker tolerant estimates the set size more accurately.

Also when machine based technique is applied for entity resolution, if there are \( n \) records then the pair of records for entity resolution will be \( n \times (n-1)/2 \). The machine based similarity helps to identify the most likely pairs. Hence the number of pairs will be less, which makes crowd based entity resolution possible.

V. CONCLUSION

In the open world model of database, the cardinality of crowd sourced data becomes a major issue. Hence statistical tools are developed to estimate the cardinality of crowd sourced data. The streaker tolerant estimator accurately predicts the total set cardinality based on the result obtained so far. The crowd collected data may still contain duplicate records, as same real world
entity can be represented in distinct ways. This leads to wrong cardinality estimation. A hybrid human-machine approach is proposed for entity resolution. The crowd based entity resolution makes data cleansing more easily. From the results obtained it is clear that entity resolution of the crowd sourced data eliminates the issue of cardinality overestimation and hence improves the cardinality estimation.

REFERENCES