Implementation of SeeDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics

Aaditree Jaisswal, Pranathi Rao Bora, Rachna Ajit Soundatti

Github Repo: `https://github.com/pranathibora14/CS645-Mini-Project-SeeDB`

1 Abstract

Our report presents the implementation and evaluation of the SeeDB framework's shared-based and pruning-based optimizations, as described in the original research paper written by Manasi Vartak [3] et.al. The focus is on reproducing the algorithm based on the definitions in Section 2, Shared-based Optimization (through query rewriting) in Section 4.1, and Pruning-based Optimization (using Hoeffding-Serfling inequality) in Section 4.2. The evaluation is conducted using the census dataset [2], with user-specified queries for married and reference queries for unmarried people, and the K-L Divergence as the utility measure. The report also discusses the choices made during the implementation process, the dataset used, and the interpretation of the results. Additionally, it will provide insights into the distribution of work among team members and their respective responsibilities throughout the project.

2 Introduction

The research paper introduces a new framework, SeeDB, to facilitate fast visual analysis. It highlights the challenges associated with manually specifying and examining numerous visualizations in high-dimensional datasets. The authors' user study conducted user evaluations, where participants performed visual analysis tasks using SeeDB and manual methods, providing insights into the bookmarking behaviour and the effectiveness of SeeDB in enabling fast visual analysis. The study collected data through interaction logs, surveys, and exit interviews, revealing that SeeDB facilitated a more thorough exploration of data and resulted in a higher number of bookmarked visualizations compared to manual methods. The results of this study demonstrated that all participants preferred SeeDB to manual methods for visual analysis, with 79% of them finding the recommendations helpful in identifying unexpected trends.

The paper also discusses the utility metrics used in the system, focusing on deviation-based metrics and the potential for incorporating other dimensions of visualization quality, the performance studies of SeeDB, and evaluating the impact of sharing and pruning optimizations on latency and accuracy. It emphasizes the potential of automated visualization recommendation systems and the effectiveness of SeeDB in enabling efficient and insightful visual analysis.

3 Problem Statement

The objective of this report is to detail the implementation of the algorithm outlined in Sections 2 and 4 of the paper, focusing on Shared-based Optimization through query rewriting and Pruning-based Optimization using the Hoeffding-Serfling inequality. Our implementation will be evaluated using census data[2], with user-specified and reference queries designed to target married and unmarried individuals, respectively. The utility measure employed for evaluation will be the K-L Divergence. Our goal is to identify the top-5 aggregate views based on this utility measure.

3.1 Dataset Selection

The census dataset[2] is used for evaluating the implementation of SeeDB. The user-specified query or target query is set to include the married people, and the reference query is set to include unmarried people.

3.2 Aggregate Function Selection

In our implementation, we have decided to include all common aggregation functions which include COUNT, MIN, MAX, AVERAGE and SUM. For some measures like capital_gain and capital_loss, aggregation functions like MIN do not make much sense but it is naturally assigned a low KL-divergence score and therefore, is not returned as an interesting visualization. This is why we have not removed any measure and function pairs from our consideration.

3.3 Utility Measure

SeeDB suggests that visualizations that depict deviations from a reference are potentially interesting. Therefore, we needed a deviation based metric, called the utility measure, to compute the distance between probability distributions extracted from a reference query and a target query. A number of different distance measures can be used including Earth Mover's Distance, Euclidean Distance, Kullback-Leibler Divergence (K-L divergence), and Jenson-Shannon Distance. We have used the K-L Divergence distance as the utility measure. Before calculating the K-L Divergence between the target and reference datasets, we have first normalized the values by dividing each element by the sum of all elements in that distribution. This ensures that it represents a valid probability distribution with values summing up to 1. Next, we have clipped the values of the probability distribution between a very small positive value, (np.finfo(float).eps), which represents the smallest positive floating-point number that can be represented in Python) and positive infinity. This is done to avoid potential issues with division by zero when calculating the KL-divergence. By ensuring that the probability is not equal to zero, it prevents potential numerical instability or errors.

3.4 Sharing Based Optimizations

For evaluating how interesting a visualization is, we need to execute two queries on the database independently, target and reference, that work on the same underlying data. This presents opportunities to intelligently merge and batch queries to minimize the number of scans of the underlying data. The paper mentions four such sharing-based optimization techniques: Combine Multiple Aggregates, Combine Multiple Group Bys, Combine target and reference view query and Parallel Query Execution. We have restricted the scope of our implementation to include Combine Multiple Aggregates and Combine target and reference view query.

3.5 Pruning Based Optimizations

The paper states that in practice, a majority of the visualizations are not very interesting and have a low utility value. Therefore, the authors introduce a phased implementation framework where at

the end of every phase, the aggregate views with the lowest utility are dropped. They introduce two pruning schemes - Confidence Interval-based Pruning and Multi-Armed Bandit Pruning. We have chosen to implement the Confidence Interval Based Pruning scheme which uses statistical confidence intervals derived from the Hoeffding-Serfling inequality to bound utilities of views. During each phase, we keep an estimate of the mean utility for every aggregate view V_i and a confidence interval around that mean. If the upper bound of the utility of view V_i is less than the lower bound of the utility of k or more views, then V_i is discarded. This is how pruning takes place.

4 Dataset Description

The census dataset, created by Ron Kohavi [2], contains a diverse range of attributes, including demographic information, educational attainment, employment status, and financial indicators. Each row in the dataset represents information about a single individual.

The dataset consists of several columns, each representing a different attribute:

1. Age: This column indicates the age of each individual in the dataset.
2. Workclass: It categorizes the type of employment for each individual, such as private, self-employed, or government.
3. Final Weight (fnlwgt): This variable represents the final weight assigned to each individual.
4. Education: It records the highest level of education attained by each individual, ranging from basic to advanced degrees.
5. Education Number: This numeric representation corresponds to the education level.
6. Marital Status: It denotes the marital status of each individual, indicating whether they are single, married or divorced.
7. Occupation: This column specifies the occupation or job role of each individual, such as clerical, managerial, or technical.
8. Relationship: It describes the individual's role within the family structure.
9. Race: This attribute captures the racial background of each individual.
10. Sex: It indicates the gender of each individual.
11. Capital Gain: This column records any financial gains realized by individuals through investments or other means.
12. Capital Loss: It represents any financial losses incurred by individuals.
13. Hours per Week: This attribute records the average number of hours worked per week by each individual.
14. Native Country: It specifies the country of origin of each individual.
15. Income: This categorical variable classifies individuals based on their income level, distinguishing between those earning above or below a certain threshold, i.e. $50,000 annually.

5 METHODOLOGY

First, we have downloaded the data from https://archive.ics.uci.edu/dataset/20/census+income, converted it and saved it into a csv file. This step was done in R programming language. The R script and the saved csv file are in the census_data folder. Secondly, we established a connection to the PostgreSQL database and created a census table. Then we imported data from the census csv file into the newly created table. Our database is ready to be used. These steps can be found in the schema.sql file.

We have implemented the SeeDB[1] framework in Python. This implementation can be found in the Jupyter Notebook called SeeDB_Implementation.ipynb. We have chosen to include the following in our list of dimensions under consideration for finding the most interesting visualizations:

- workclass
- education
- occupation
- relationship
- race
- sex
- native_country
- income

We have chosen to include the following measures:

- age
- capital_gain
- capital_loss
- hours_per_week

The common aggregate functions considered are as follows:

- avg
- sum
- min
- max
- count

Here, we would like to mention that we have chosen to omit the fnlwgt dimension from the dataset since this dimension does not represent information that is relevant to our analysis. We have also decided to omit education_num from our analysis because it is numerical representation of the categorical column education.

We classify adults with marital status as Married-civ-spouse, Married-spouse-absent, or Married-AF-spouse as married adults, and thus include them in the user-defined query condition. Conversely, adults with marital status as Divorced, Never-married, Separated, or Widowed are considered unmarried adults and included in the reference query condition.

6 IMPLEMENTATION OF THE SEEDB FRAMEWORK

Firstly, we have created a dictionary object called views that represents all the possible combinations of dimension, measure and feature views that we are considering. We have followed

the phased execution framework explained in the paper and and in each phase we operate on a subset of the data. To create these phases, we have divided the data into 10 partitions. For every partition, we implement sharing based optimization through query rewriting and pruning-based optimization to discard views with the lowest utility in that phase. In the following subsections, we have explained our implementation of the two optimizations.

6.1 Sharing Based Optimization

The code iterates through each attribute(dimension) in the views dictionary. For each attribute, it construct an SQL query that groups by that particular attribute and then applies all the aggregate functions for every measure in the dataset using the COALESCE keyword. This is how we have combined multiple aggregates into a single SQL query. Therefore, we only run those many SQL queries as the number of attributes that we have to group by. We have used the CASE keyword in constructing the SQL query to get aggregates for the target and reference view simultaneously in just one query. This is how we have combined target and reference view query.

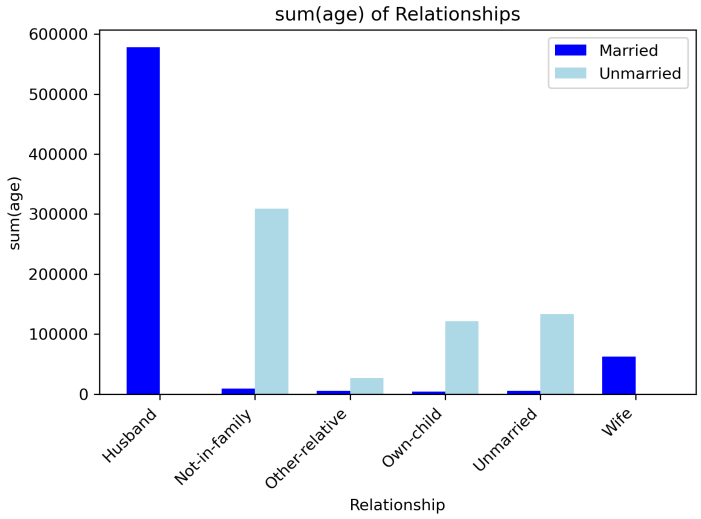
6.2 Pruning Based Optimization

In the pruning based optimization, we iterate through every (dimension, measure, feature) combination and calculate the KL-divergence measure for target and reference query data for that combination. We append this into an array of KL-divergence scores. We do not prune any views in the first phase. In the subsequent phases, we start pruning views with the lowest utility score using confidence intervals. We uses worst case confidence intervals, derived from the Hoeffding-Serfling inequality as suggested in the paper. We calculated a running confidence interval around the current mean of selected values such that the actual mean of all values is always within this confidence interval with a probability of $1-\delta$. We have considered δ to be 0.05. We execute this pruning step till the last phase where we directly extract the top-5 aggregate views with the highest KL-divergence.

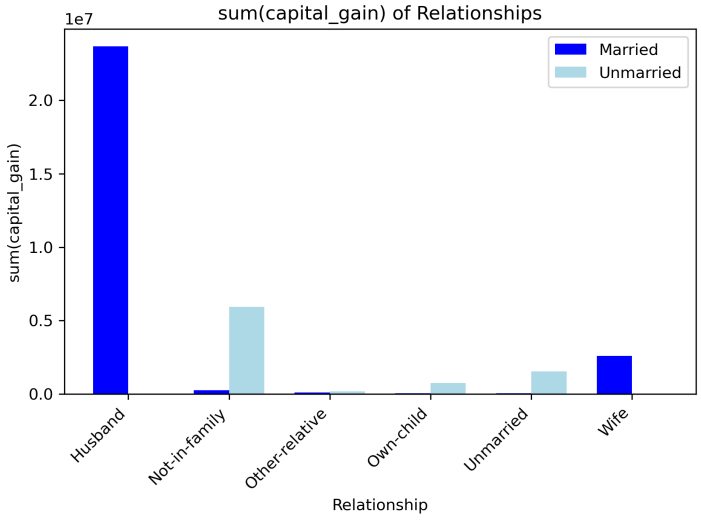
7 Results

In the below plots we can observe considerable divergence in the query results on the target data i.e. married adults, represented in blue and the reference data i.e. unmarried adults, represented in light blue. We have extracted the views which have the highest KL divergence as discussed in the paper SeeDB which states that visualizations which show high utility are the most interesting. The final recommended top 5 views that we got as output are listed below:

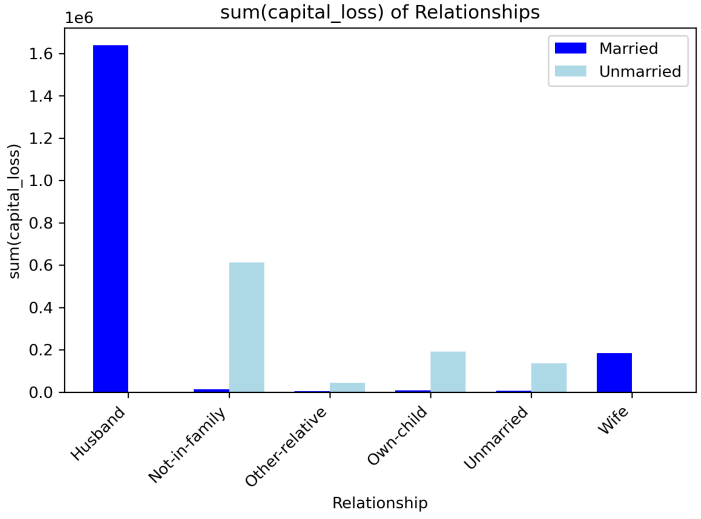
- **Dimension:** Relationship, **Measure:** Age, **Function:** Sum



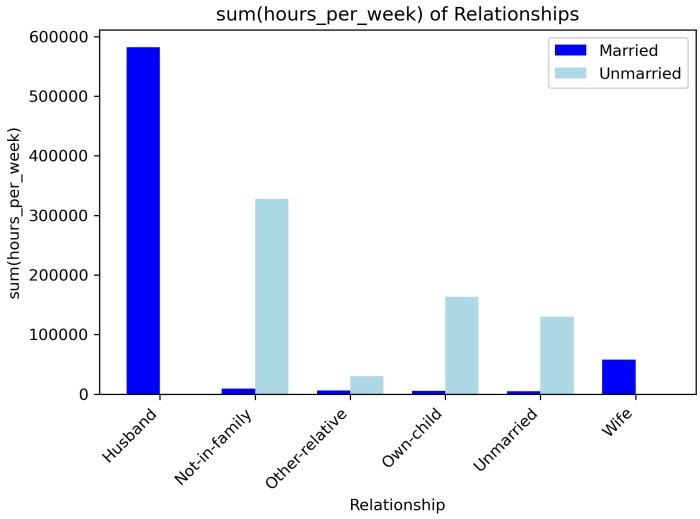
- **Dimension:** Relationship, **Measure:** Capital Gain, **Function:** Sum



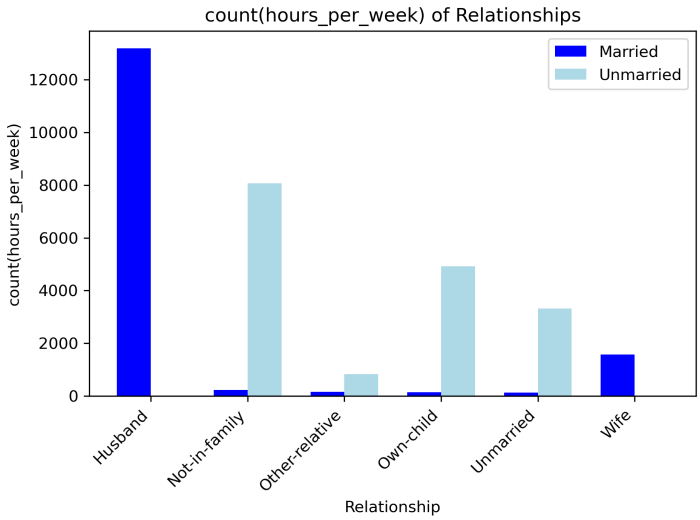
- **Dimension:** Relationship, **Measure:** Capital Loss, **Function:** Sum



- **Dimension:** Relationship, **Measure:** Hours per Week, **Function:** Sum



- **Dimension:** Relationship, **Measure:** Hours per Week, **Function:** Count



8 Conclusion

We have presented the top 5 interesting visualizations(i.e. top 5 aggregate views) on the census dataset by implementing the algorithms based on the definitions of sharing and pruning based optimization algorithms provided in Seedb [3]. We set the user-specified query to include married people, and the reference query to include unmarried people and used the K-L Divergence as the utility measure. For future scope, we can consider implementing parallel query execution and combining multiple group-by optimization that has been mentioned in the sharing optimization section of SeeDB paper.

9 Contribution

The work was equally divided between all three team members with respect to data querying and preparation, KL divergence calculation, view generation, sharing-based and pruning-based optimization, getting the final recommended views, visualization, and writing the report.

- **Aaditree:**

- Implementing the sharing based optimization algorithm.
- Writing the report.

- **Pranathi:**
  - Writing the schema and SQL file that would be used to load the data from the CSV file to the PostgreSQL database.
  - Implementing the pruning based optimization algorithm and KL divergence and getting the top 5 aggregate views.
  - Writing the report.
- **Rachna:**
  - Converting the downloaded dataset using a CSV file.
  - Connecting the database to the Python notebook using psycopg2 adapter.
  - Writing code to visualize the top aggregate views.
  - Writing the report

REFERENCES

[1] Pranathi Rao Bora, Aaditree Jaisswal, and Rachna Ajit Soundatti. CS645-Mini-Project-SeeDB. https://github.com/pranathibora14/CS645-Mini-Project-SeeDB.

[2] Ron Kohavi. Census Income. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5GP7S.

[3] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. Seedb: Efficient data-driven visualization recommendations to support visual analytics. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, volume 8, page 2182. NIH Public Access, 2015.