CS 8803-MDS
Human-in-the-loop Data Analytics

Lecture 20

11/02/22

Today's class

SeeDB: efficient data-driven visualization recommendations to support visual analytics

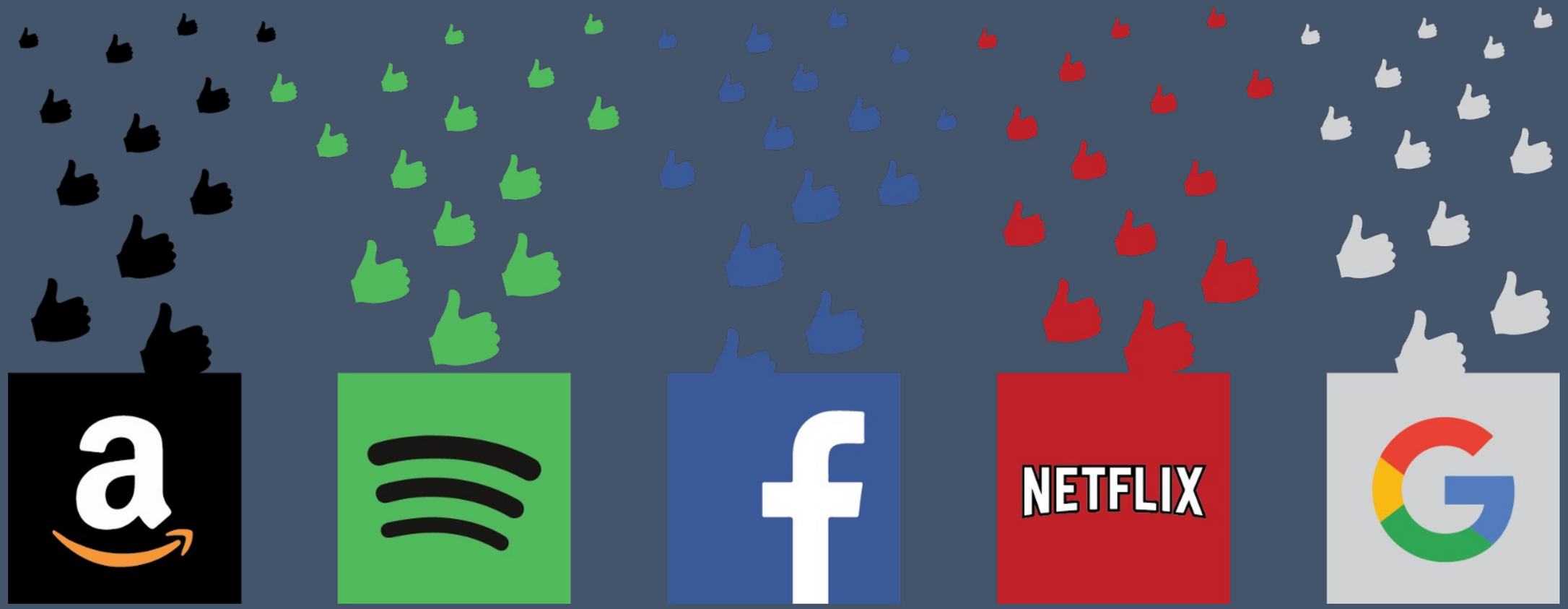
Authors: Ting

Reviewer: Shen En, Harshal

Archaeologist: Cangdi

Practioner: Ashmita

Recommendation systems



Recommendation systems

Content-based filtering: recommend items that are similar to those user liked in the past

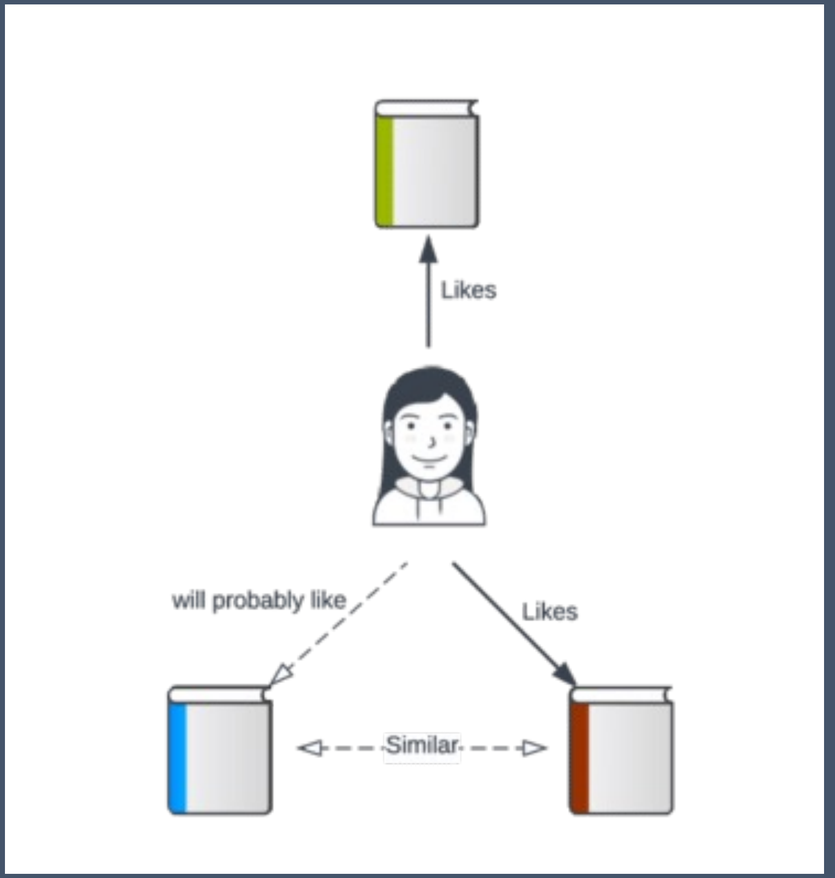


Image source: A new horizon for the recommendation: Integration of spatial dimensions to aid decision making

Recommendation systems

Content-based filtering: recommend items that are similar to those user liked in the past

Collaborative filtering: recommend items that similar users liked

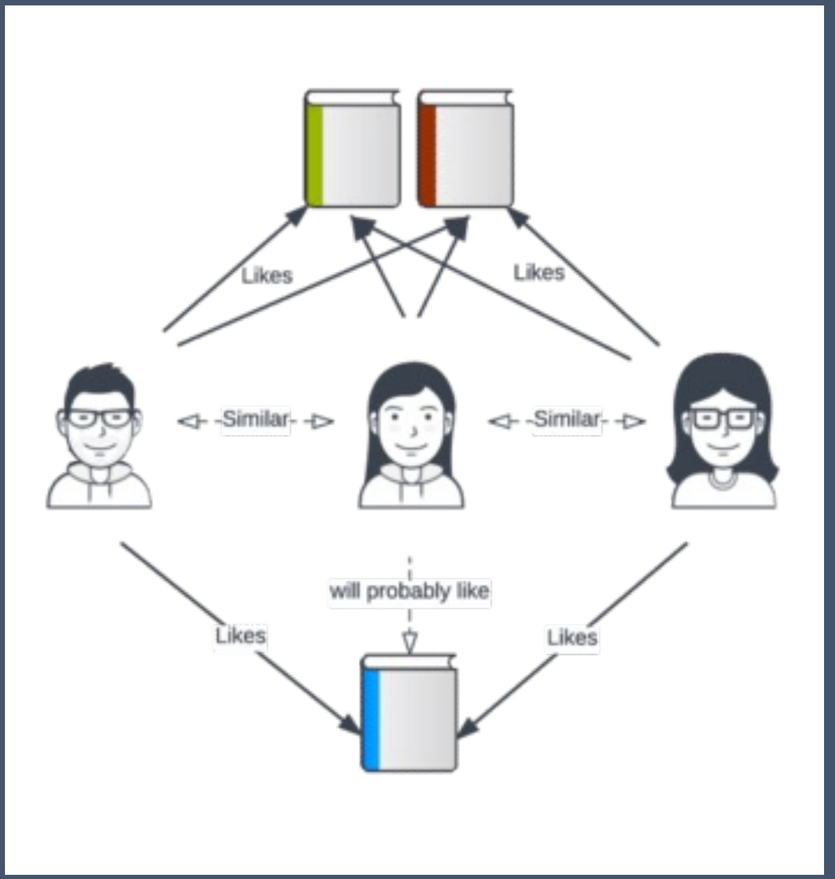


Image source: A new horizon for the recommendation: Integration of spatial dimensions to aid decision making

Recommendation systems

Content-based filtering: recommend items that are similar to those user liked in the past

Collaborative filtering: recommend items that similar users liked

Knowledge-based filtering: recommend items based on knowledge base

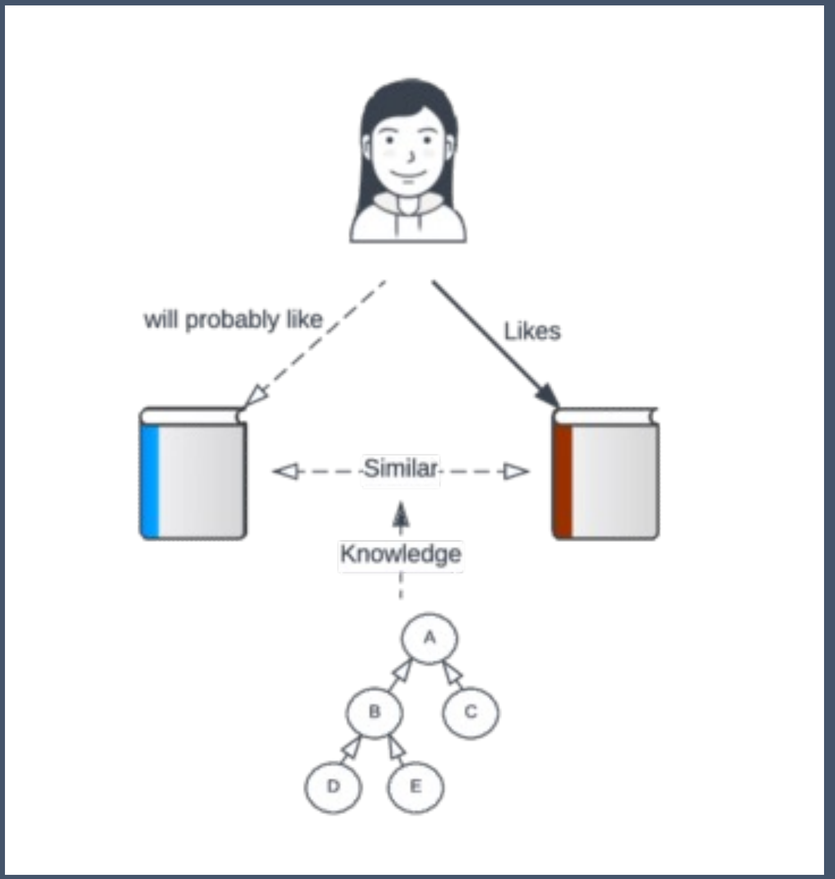


Image source: A new horizon for the recommendation: Integration of spatial dimensions to aid decision making

Recommendation systems

Content-based filtering: recommend items that are similar to those user liked in the past

Collaborative filtering: recommend items that similar users liked

Knowledge-based filtering: recommend items based on knowledge base

Q: Can these techniques apply to viz recommendation systems?

Viz recommendation systems

Q: Can these techniques apply to viz recommendation systems?

Observations:

- common use case: new datasets analyzed by new users
- lack of historical ratings
- the set of ”items” is not fixed; depends on the the specific task
- false discoveries is a unique concern for viz recommendation systems

Today's class

SeeDB: efficient data-driven visualization recommendations to support visual analytics

Authors: Ting

Reviewer: Shen En, Harshal

Archaeologist: Cangdi

Practioner: Ashmita

42nd International Conference on

VERY LARGE DATA BASES

New Delhi, India • September 5 – 9, 2016

Very Large Data Bases INDIA 2016

Very Large Data Bases INDIA 2016

VLDB is a premier annual international database research conference. It was held in India first in Mumbai in 1996 in Mumbai and then returned after 20 years as VLDB 2016 with Persistent Systems being the key organizing sponsor.

The conference consisted of peer-reviewed contributed papers, tutorials, workshops and keynote sessions by Prof. Ion Stoica from University of California and Dr. Anand Rajaraman, a well know serial entrepreneur from Silicon Valley. The event well attended by more than 900 academicians, practitioners and students from 39 countries.

With a population in excess of 1.3 billion, nearly a billion Aadhar cards and 200+ million Jan-Dhan Yojana accounts, anything India-scale implies large data. In conjunction with the VLDB 2016 conference, the conference hosted a one-day Digital India Symposium which highlighted India-scale challenges and showcased innovative solutions. The conference also hosted a data hackathon which created excitement about data management issues and received some great participation.

Keynote session by Prof. Ion Stoica of University of California, Berkeley about Trends and Challenges in Big Data Processing

VIEW PHOTO GALLERY

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

Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, Neoklis Polyzotis

Presenter: Ting Yu

Problem Motivation

Data visualization is often the first step in data analysis.

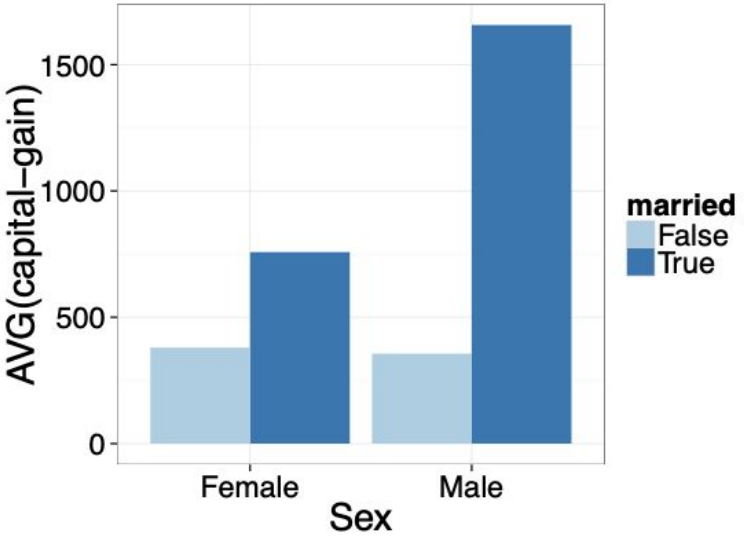
We use visualization to...

- get a feel for the data,
- find anomalies and outliers,
- identify patterns that might merit further investigation

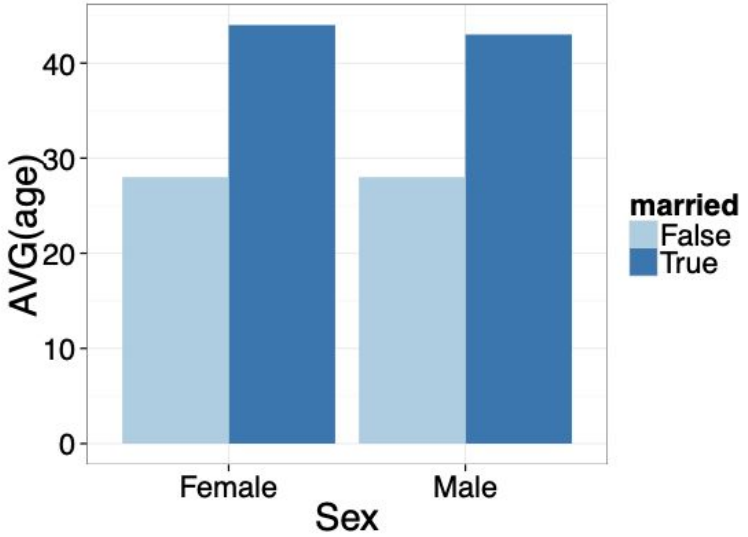
But identifying useful visualization is a non-trivial task, especially given higher dimensional data.

An Motivation Example

How marital-status impacts socio-economic indicators like education and income?

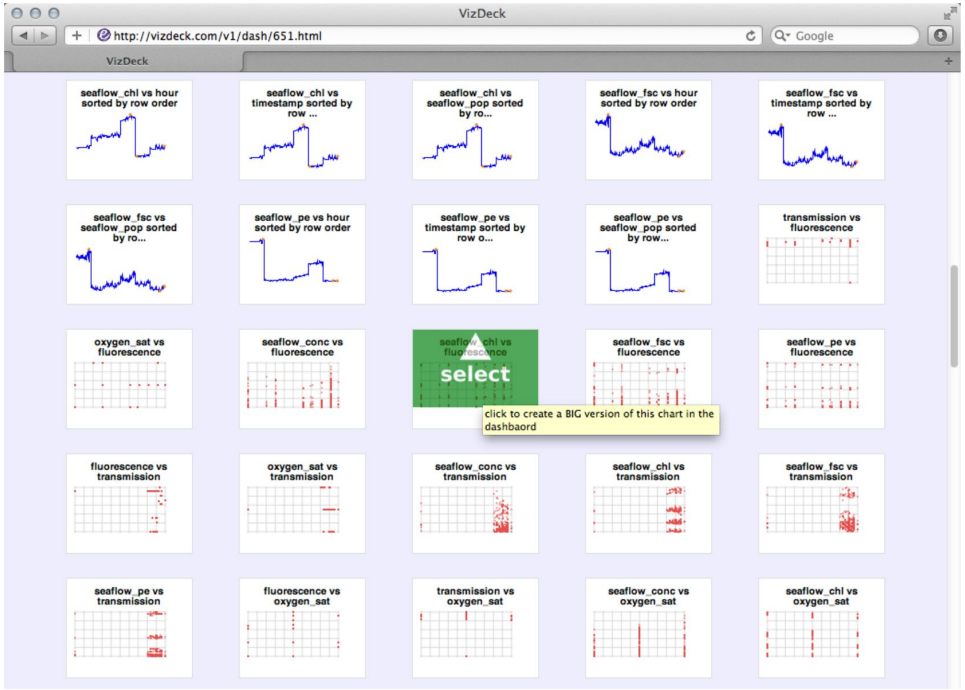


(a) Interesting Visualization



(b) Uninteresting Visualization

One approach could be...



But this solution does not scale well to higher dimensional data.

2 challenges in selecting the most interesting visualizations

1. What visualization is worth recommending?

   “Deviation from reference” as an measure of interestingness

2. How do we make the recommendation in interactive speed?

   An engine that uses sharing computation and pruning computation to speed up

Contributions

Metric: deviation from reference, as a measure of interestingness

System: an engine optimized with sharing computation and pruning computation

SeeDB can be deployed as a middleware layer for any SQL-compliant DBMS

Formulating the Recommendation Problem

Problem Formulation - Data

D, a database

Q, a generic select-project-join query (we assume that the analyst has indicated a desire to explore a subset of data in a joined table from all possible schemas)

D_Q, the result of Q on D, **essential a subset of D**

D_R, a reference dataset, i.e. some D_Q, default D_R = D

Problem Formulation - Aggregate View

A, an attribute we want to groupby

M, the other attribute we want to aggregate

F, the aggregate function

V, the view defined by some (a, m, f), i.e.

$$Q_T = \texttt{SELECT}\ a, f(m)\ \texttt{FROM}\ D_Q\ \texttt{GROUP BY}\ a$$

An aggregate view is just an query!

Problem Formulation - Utility Function

We have two data, one data in interest and the other as a reference

D_Q, the result of Q on D

D_R, a reference dataset, default = D

$$Q_T = \texttt{SELECT}\ a, f(m)\ \texttt{FROM}\ D_Q\ \texttt{GROUP BY}\ a$$

$$Q_R = \texttt{SELECT}\ a, f(m)\ \texttt{FROM}\ D_R\ \texttt{GROUP BY}\ a$$

By applying the same view V, i.e. (a, m, f)

Problem Formulation - Utility Function

To ensure all aggregate summaries have the same scale, we **normalize each aggregate summary into a probability distribution** (i.e. the values of f (m) sum to 1).

Then we have **two probability distributions to compare**.

We then choose **a distance measure, S,** between probability distributions. S could be KL-divergence, Euclidean Distance, etc. We use Earth Mover's Distance by default.

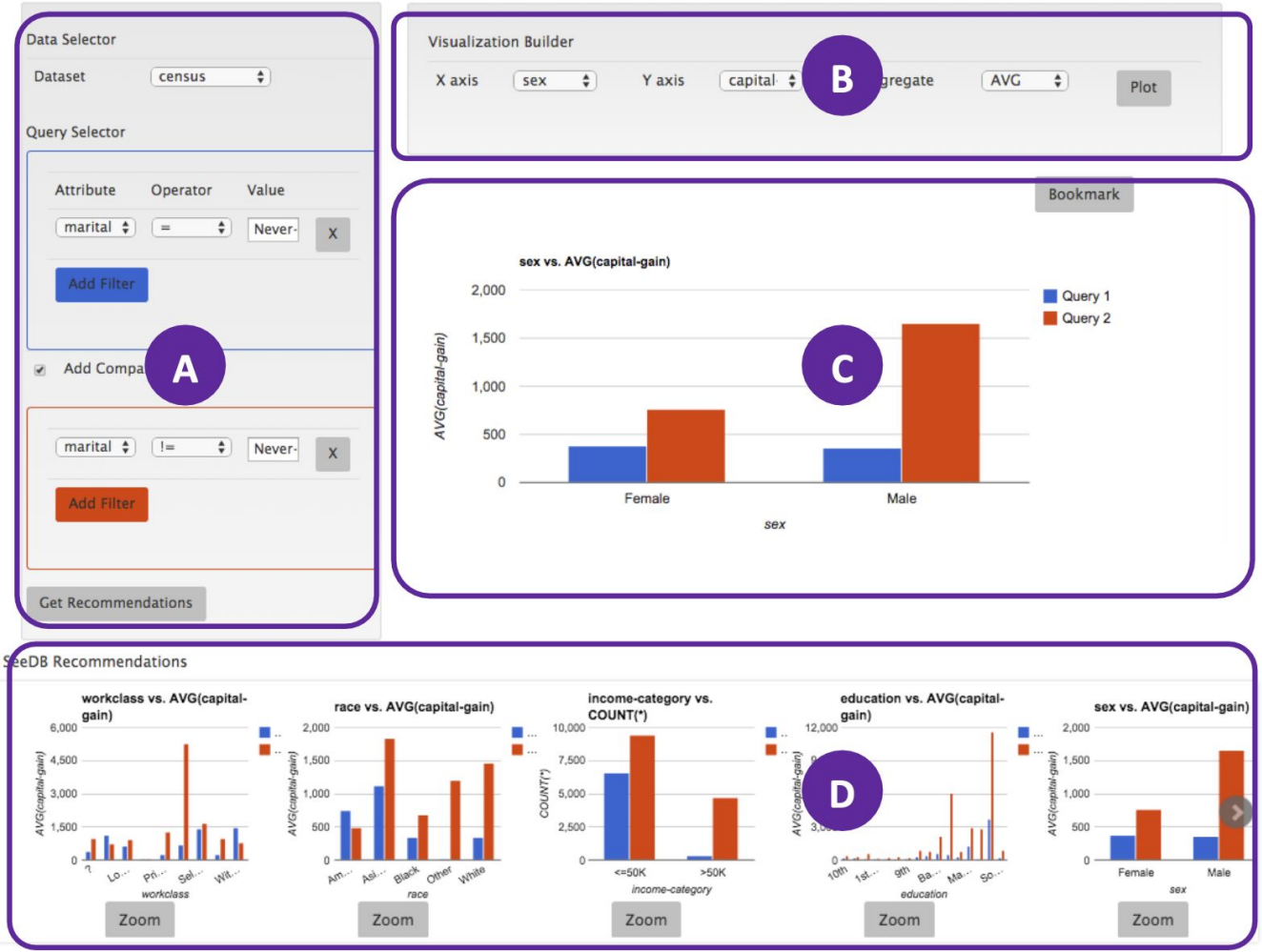
$$U(V_i) = S(P[V_i(D_Q)], P[V_i(D_R)])$$

The reference distribution is not uniform!

Overall Problem Formulation

Given a user-specified query Q on a database D, a reference dataset DR, a utility function U, and a positive integer k, **find k aggregate views V ≡ (a, m, f) that have the largest values of U (V)** among all the views (a, m, f), while minimizing total computation time.

Front-end


Data Selector
Dataset
census
Query Selector
Attribute
Operator
Value
marital
=
Never-
X
Add Filter
A
Add Compa
marital
!=
Never-
X
Add Filter
Get Recommendations
Visualization Builder
X axis
sex
Y axis
capital
B
gregate
AVG
Plot
Bookmark
sex vs. AVG(capital-gain)
2,000
1,500
1,000
500
0
AVG(capital-gain)
Query 1
Query 2
C
Female
Male
sex
SeeDB Recommendations
workclass vs. AVG(capital-gain)
race vs. AVG(capital-gain)
income-category vs. COUNT(*)
education vs. AVG(capital-gain)
sex vs. AVG(capital-gain)
D
Zoom
Zoom
Zoom
Zoom
Zoom

System Design for Interactivity

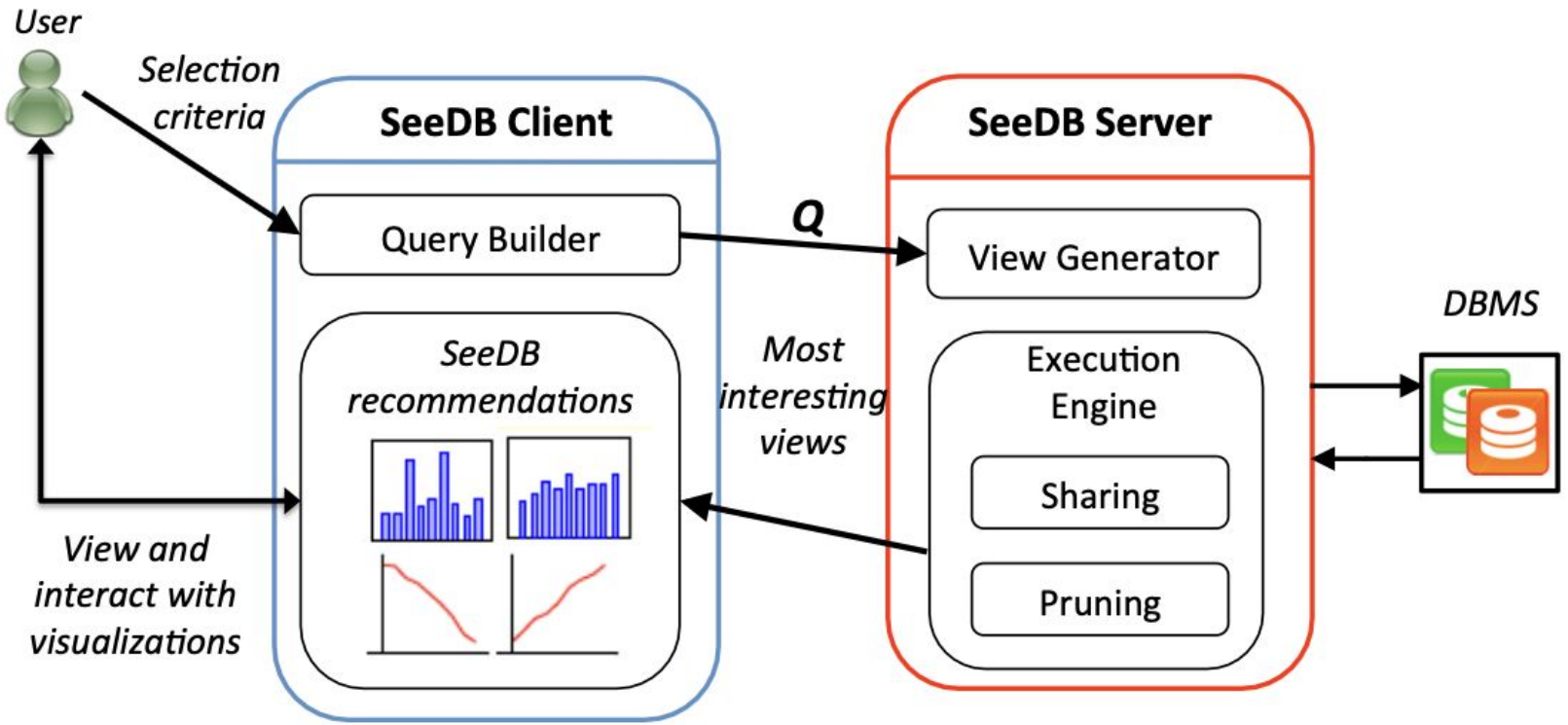


Figure 3: SeeDB Architecture

Basic execution without optimization

For each aggregate view, it generates a SQL query corresponding to the target and reference view, and issues the two queries to the underlying DBMS.

There are in total 2 x A x M x F queries.

$$Q_T = \texttt{SELECT}\ a, f(m)\ \texttt{FROM}\ D_Q\ \texttt{GROUP BY}\ a$$

$$Q_R = \texttt{SELECT}\ a, f(m)\ \texttt{FROM}\ D_R\ \texttt{GROUP BY}\ a$$

Sharing Computation

Sharing computation in our setting is a special case of the general problem of multi-query optimization

1. Combine multiple aggregates
2. Combine multiple groupbys
3. Combine target and reference view query
4. Parallel query execution

Sharing Computation - Combine multiple aggregates

Given a groupby attribute a1,

Instead of (a1, m1, f1), (a1, m2, f2) ...(a1, mk, fk),

Do (a1, {m1, m2 . . . mk}, {f1 , f2 . . . fk}) in one query.

Sharing Computation - Combine multiple groupbys

We verify that **grouping can benefit performance so long as memory utilization for grouping stays under a threshold S**.

OPTIMAL GROUPING PROBLEM: Given memory budget S and a set of dimension attributes A = {a1 . . . an }, divide the dimension attributes in A into groups A1,...,Al (where Ai $\subseteq$ A and UAi = A) such that if a query Q groups the table by any Ai, the memory utilization for Q does not exceed S.

Isomorphic to NP-Hard bin packing problem.

We use the standard **first-fit algorithm** to find the optimal grouping.

Sharing Computation - Combine target and reference

$$Q1 = \texttt{SELECT } a, f(m) \texttt{ FROM } D \texttt{ WHERE x} < 10 \texttt{ GROUP BY } a$$

$$Q2 = \texttt{SELECT } a, f(m) \texttt{ FROM } D \texttt{ GROUP BY } a$$

$$Q3 = \texttt{SELECT } a, f(m), \texttt{CASE IF x} < 10 \texttt{ THEN 1 ELSE 0 END}$$
$$as\ g1,\ 1\ as\ g2 \texttt{ FROM } D \texttt{ GROUP BY } a,\ g1,\ g2$$

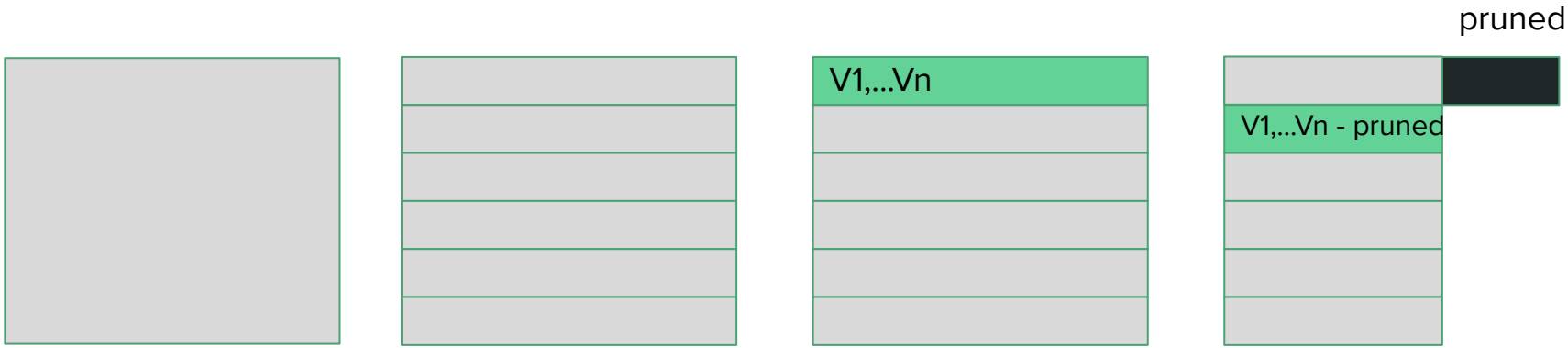
Sharing Computation - Parallel query execution

SeeDB executes multiple view queries in parallel as these queries can often share buffer pool pages, reducing disk access times.

Pruning Computation

In practice, most visualizations are low-utility, meaning computing them wastes computational resources.

The core idea is to **use partial results** for each view based on the data processed so far **to estimate utility** and views with low utility are dropped.



Pruning Computation

2 strategies:

1. Confidence Interval-Based Pruning
2. Multi-Armed Bandit Pruning

Pruning Computation - Confidence-interval based pruning

During each phase, we keep **an estimate of the mean utility for every aggregate view Vi and a confidence interval** (derived from Hoeffding-Serfling inequality).

At the end of a phase, if the upper bound of the utility of view Vi is less than the lower bound of the utility of k or more views, then Vi is discarded.

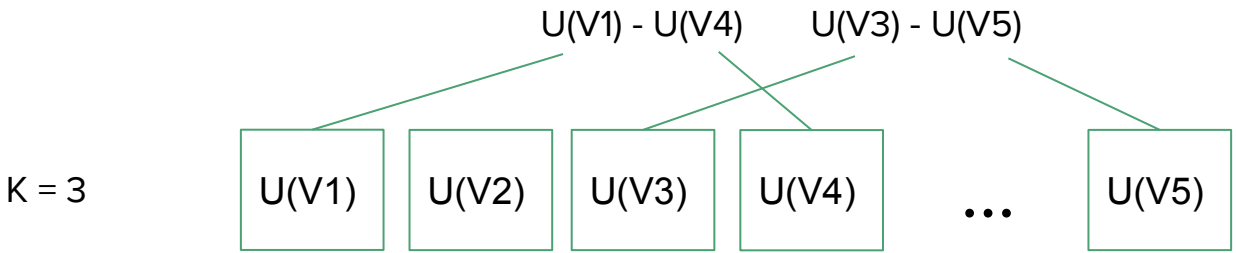
THEOREM 4.1. *Fix any* $\delta > 0$. *For* $1 \leq m \leq N-1$, *define*

$$\varepsilon_m = \sqrt{\frac{(1 - \frac{m-1}{N})(2\log\log(m) + \log(\pi^2/3\delta))}{2m}}.$$

Then: $$\Pr\left[\exists m, 1 \leq m \leq N : \left|\frac{\sum_{i=1}^{m} Y_i}{m} - \mu\right| > \varepsilon_m\right] \leq \delta.$$

Pruning Computation - Multi-Armed Bandit Pruning

Multi-Armed Bandit strategy (MAB) is an online algorithm repeatedly chooses from a set of alternatives (arms) over a sequence of trials to maximize reward.



Ordered from highest **mean utility** to -> lowest mean utility

If U(V1) - U(Vk+1) > U(Vk) - U(Vn), then V1 is accepted as part of top-k and no longer in the process.

Pruning Computation - Consistent Distance Functions

How do you guarantee the estimation selects the top-k?

We can show that, as we sample more and more, the estimated utility $\hat{U}$ can be made to be arbitrarily close to U for all aggregate views. Essentially, this means that **a pruning algorithm that uses a sufficiently large sample will prune away low utility views with high probability**.

We find distance functions that have satisfy the above property as consistent distance functions. **Consistent distance functions allow pruning schemes to gather increasingly better estimates of utility values over time** (as long as the samples are large enough).

System Evaluation

Testing Data

Name	Description	Size	$\|A\|$	$\|M\|$	Views	Size (MB)
Synthethic Datasets						
SYN	Randomly distributed, varying # distinct values	1M	50	20	1000	411
SYN*-10	Randomly distributed, 10 distinct values/dim	1M	20	1	20	21
SYN*-100	Randomly distributed, 100 distinct values/dim	1M	20	1	20	21
Real Datasets						
BANK	Customer Loan dataset	40K	11	7	77	6.7
DIAB	Hospital data about diabetic patients	100K	11	8	88	23
AIR	Airline delays dataset	6M	12	9	108	974
AIR10	Airline dataset scaled 10X	60M	12	9	108	9737
Real Datasets - User Study						
CENSUS	Census data	21K	10	4	40	2.7
HOUSING	Housing prices	0.5K	4	10	40	<1
MOVIES	Movie sales	1K	8	8	64	1.2

Evaluation Method

3 performance metrics:

1. **Time taken** to return the top-k visualizations.

For experiments involving pruning strategies, we also measure quality of results:

2. **Accuracy**
3. **Utility distance**

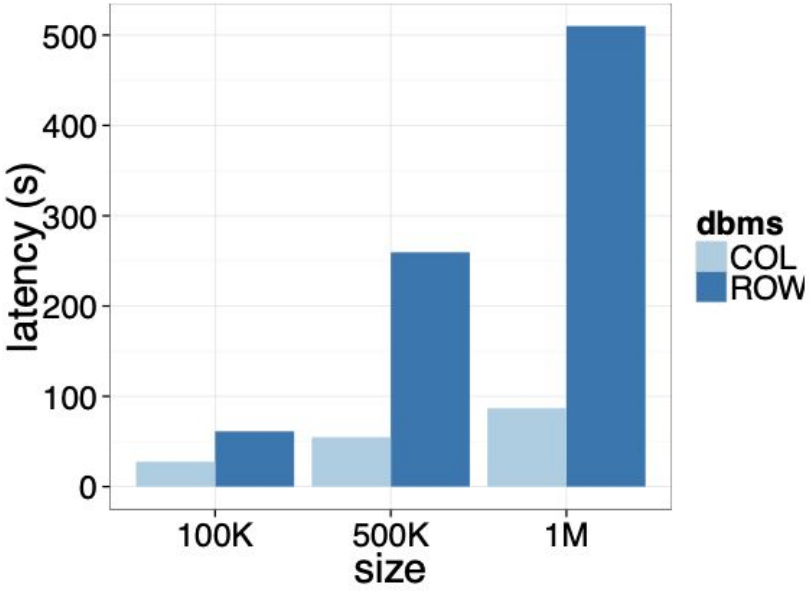
Since data layout to impact the efficacy of optimizations, we evaluate on both a row-oriented database (denoted **ROW**) a column-oriented database (denoted **COL**).

The experiments use earth mover's distance (EMD) as distance function .

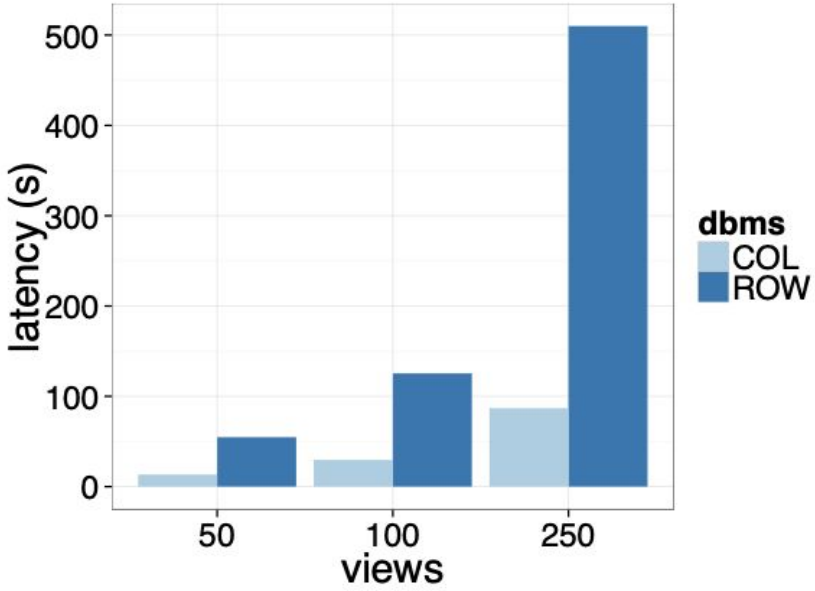
Results Summary

1. 6–40X speedup from sharing
2. 5X speedup from pruning without loss of accuracy
3. Multiplicative gains
4. > 100X speedup overall
5. Gains improve on larger datasets

Baseline - 2 x (A x M x F)



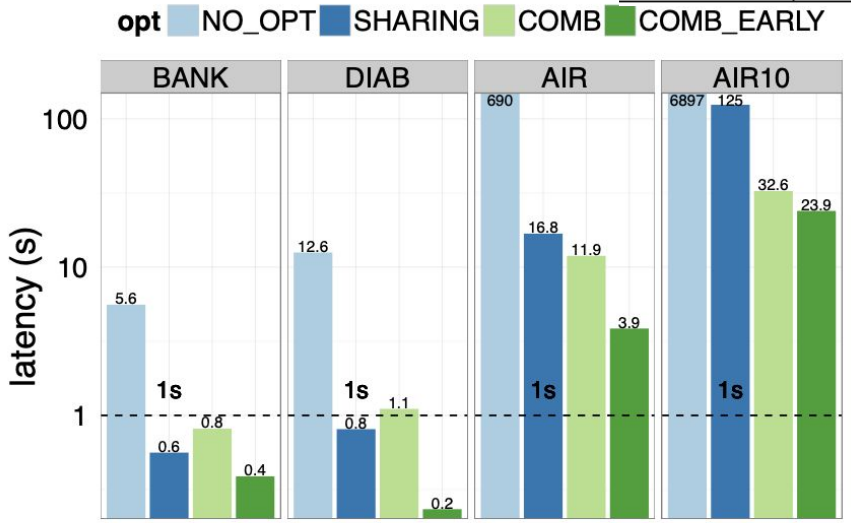
(a) Latency vs. Table size



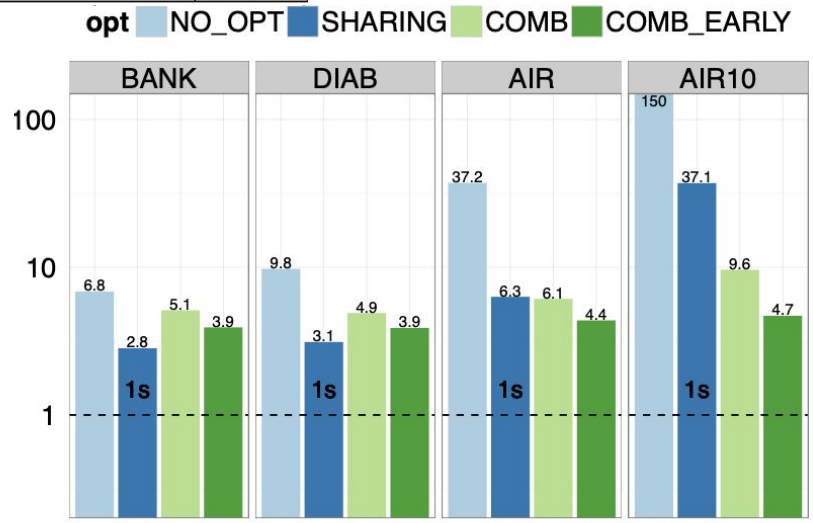
(b) Latency vs. Num Views

Results - Latency

		Real Datasets
BANK	Customer Loan dataset	40K
DIAB	Hospital data about diabetic patients	100K
AIR	Airline delays dataset	6M
AIR10	Airline dataset scaled 10X	60M



(a) Optimization results for ROW

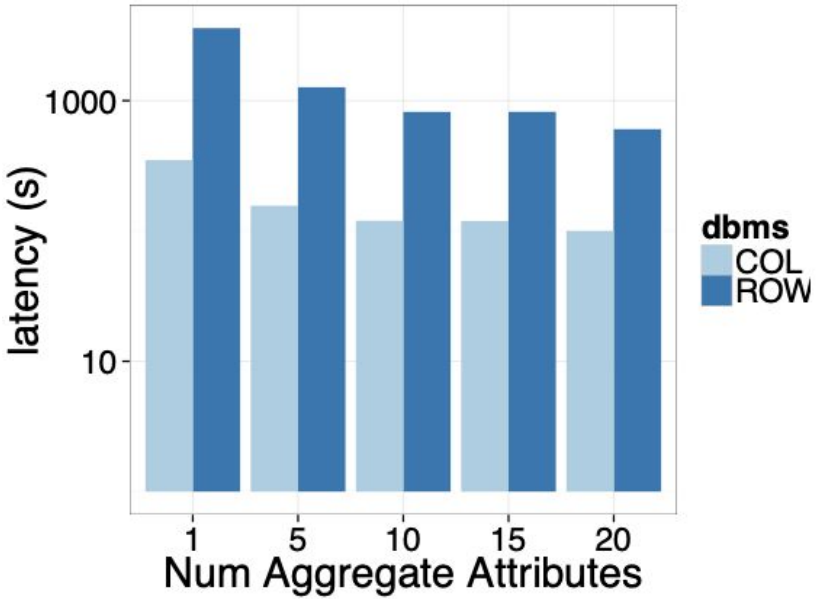


(b) Optimization results for COL

COMB: both sharing and pruning applied. COMB (COMB EARLY): return approximate results as soon as the top-k visualizations have been identified.

Results obtained with confidence Interval (CI) pruning scheme and k=10.

Results - Sharing Computation - Multiple Aggregates



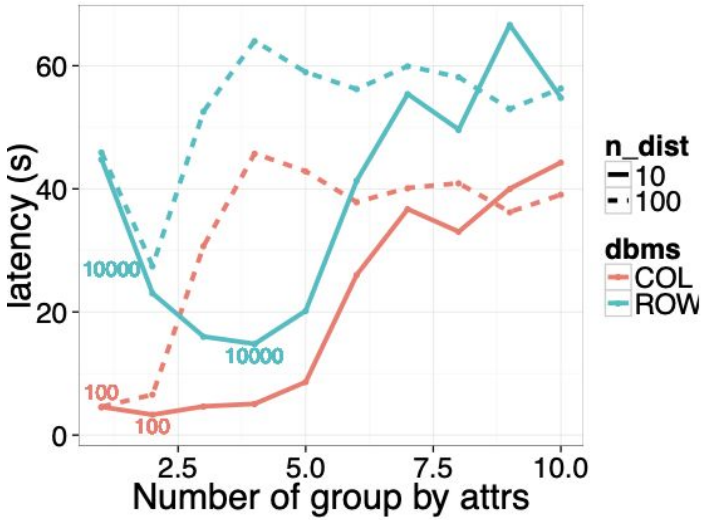
(a) Latency vs. aggregates

Overall, combining aggregates provides a 4X speedup for ROW and 3X for COL

Results - Sharing Computation - Multiple GroupBys

For ROW, once the memory budget (proxied by the number of distinct groups) exceeds 10000, latency increases significantly. We see a similar trend for COL, but with a memory budget of 100.

Thus, we find empirically that memory usage from grouping is, in fact, related to latency and that **optimal groupings must respect the memory threshold**.

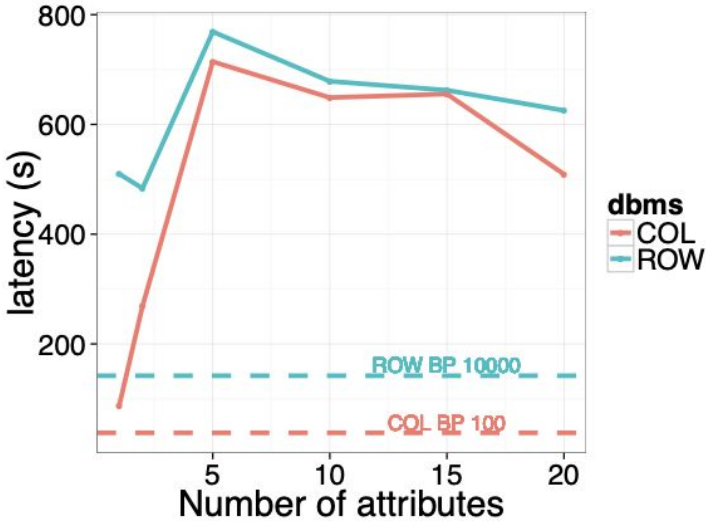


(a) Latency vs. Num of Groups

n_dist: n distinct values

Results - Sharing Computation - Multiple GroupBys

Bin packing strategy (dashed line), consistently keeps memory utilization under the memory budget, compared to simply setting a limit on the number of group-bys in each query (solid line).

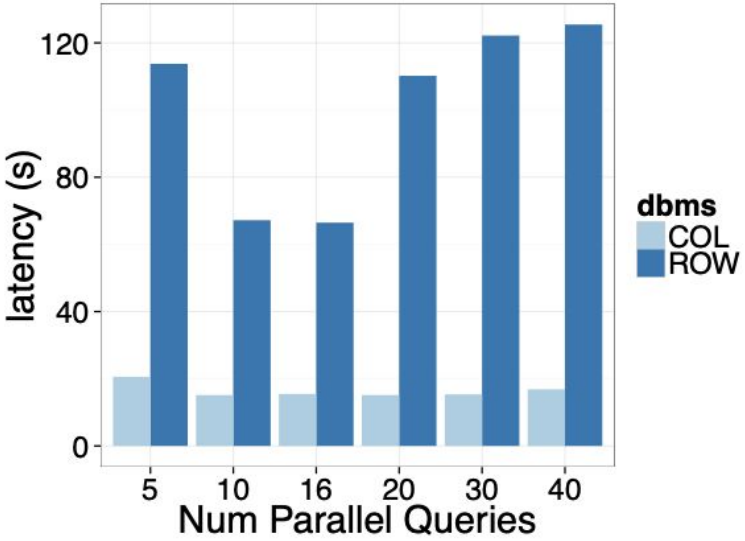


(b) Latency vs. Num Dimensions

Results - Sharing Computation - Parallelism

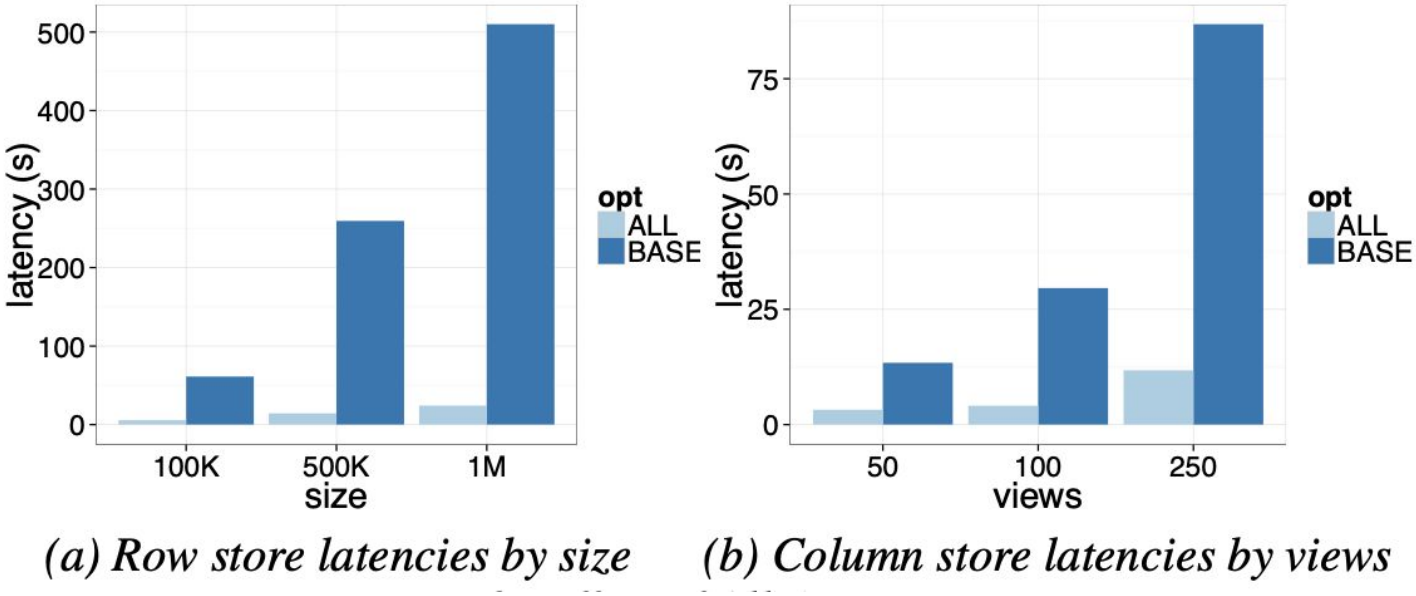
Low levels of parallelism produce sizable performance gains but high levels degrade performance.

The optimal number of queries to run in parallel is approximately 16 (equal to the number of cores)



(b) Effect of parallelism

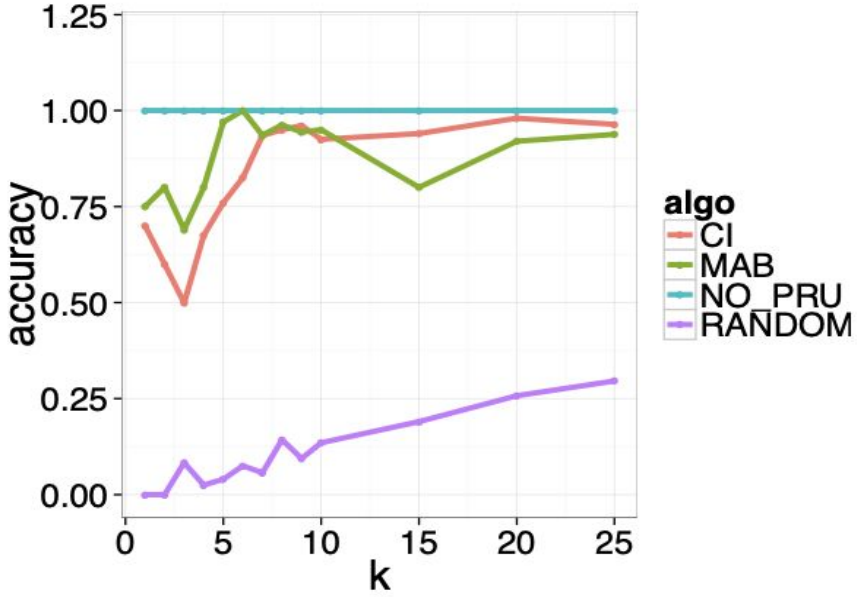
Results - Sharing Computation - Overall

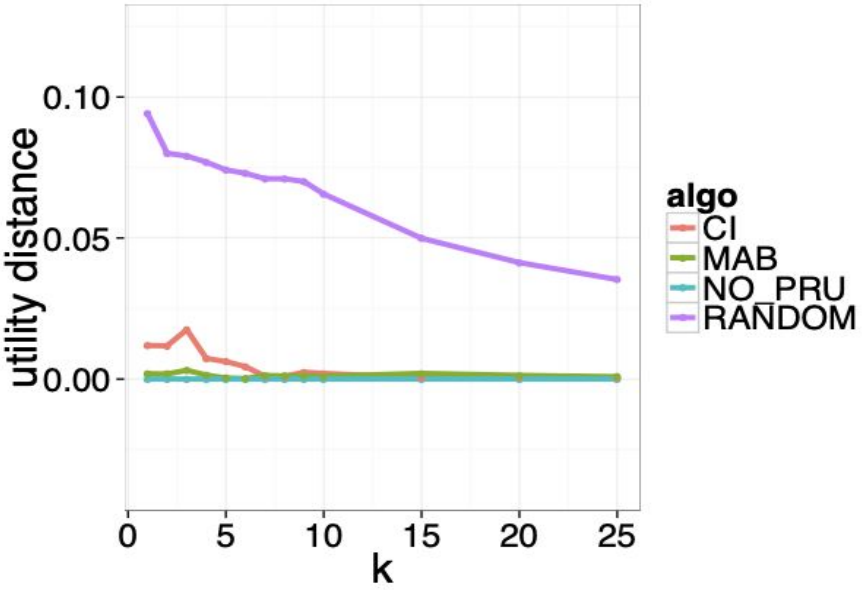


(a) Row store latencies by size *(b) Column store latencies by views*

Overall speedup of up to 40X for row stores, and 6X for column stores; column stores are still faster than row stores.

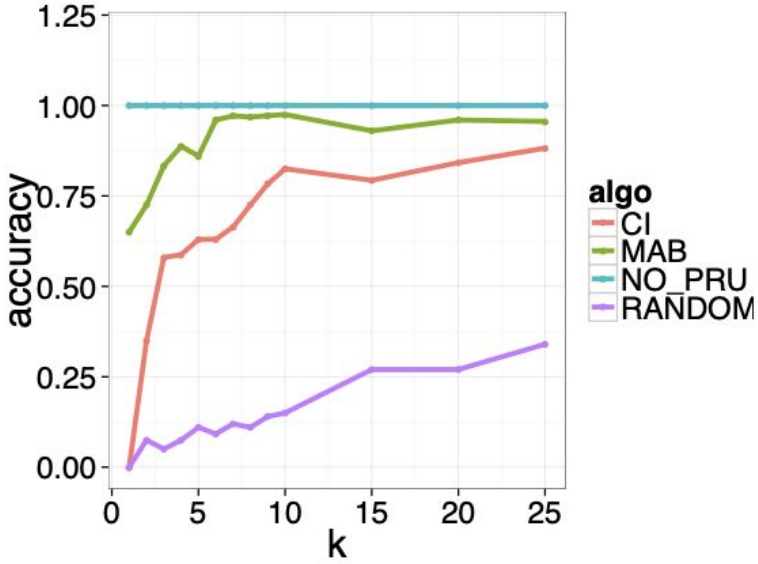
Results - Pruning Accuracy



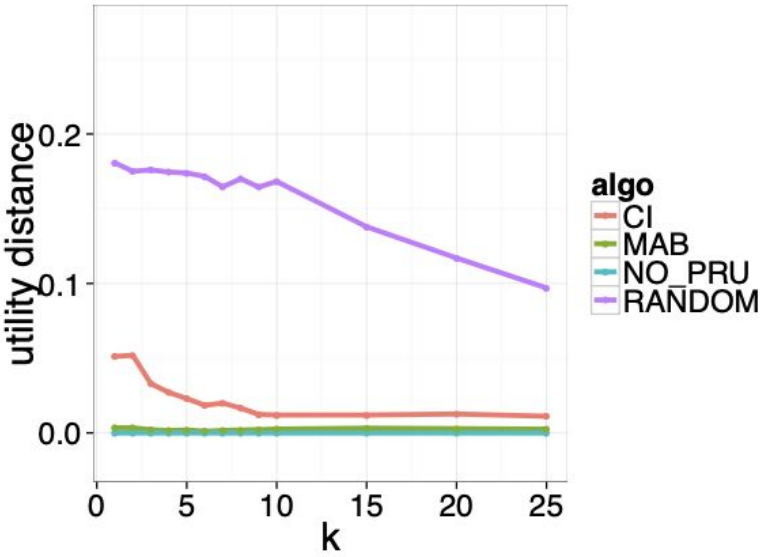


BANK dataset

Results - Pruning Accuracy

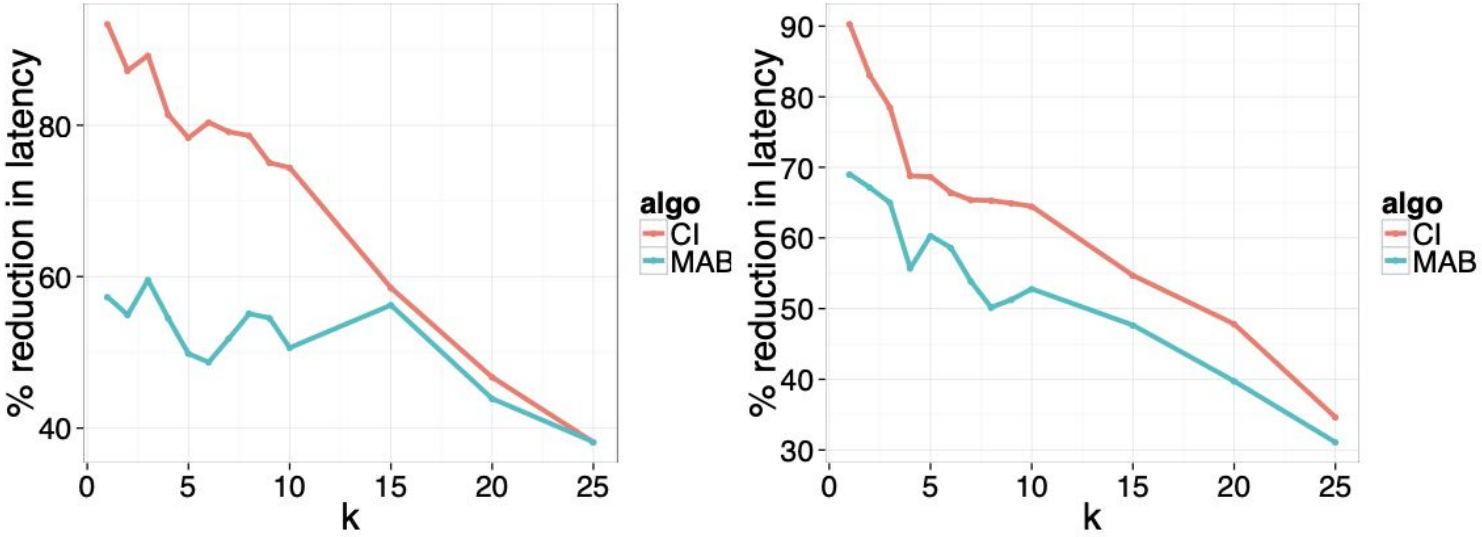


(a) Diabetes Dataset Accuracy



(b) Diabetes Dataset Utility Dist.

Results - Pruning latency improvement



(a) Bank Dataset Latency *(b) Diabetes Dataset Latency*

Figure 13: Latency Across Datasets

User Study

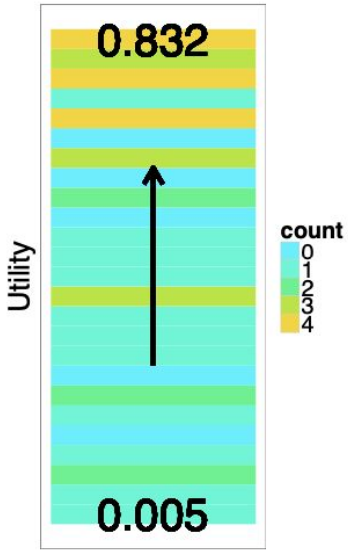
User Study - Deviation Metric Validity

Ground Truth:

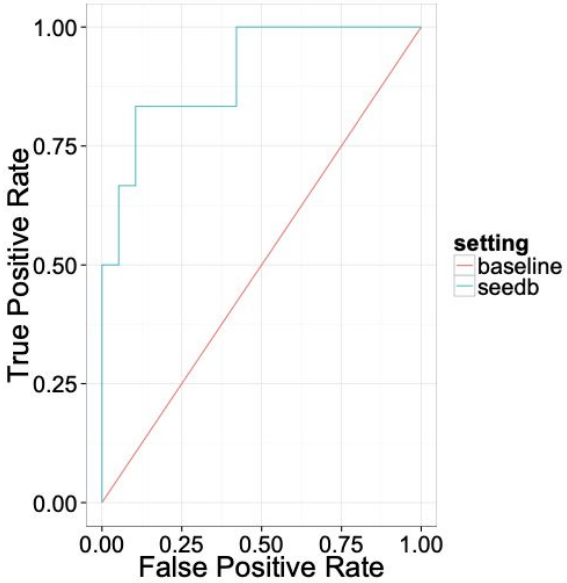
5 data analysis experts are presented with visualizations of the Census dataset and they label them as either interesting or not interesting.

Results - Deviation Metric Validity

Conclusion: SeeDB recommendations have high quality and coverage



(a) Utility Distribution

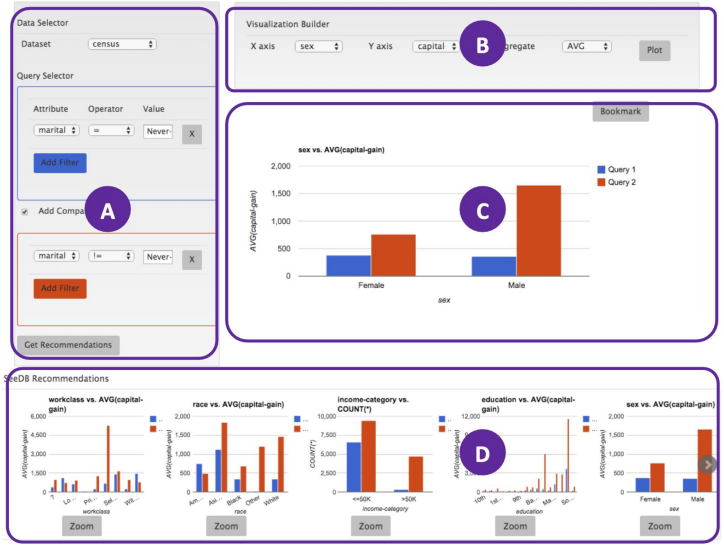


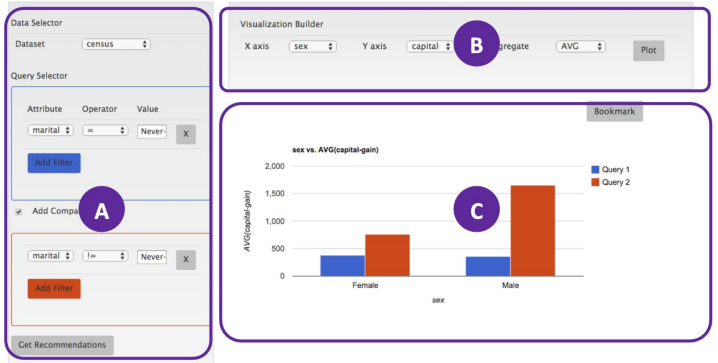
(b) ROC of SeeDB (AUROC = 0.903)

(yellow = popular, blue = not popular)
Might be a typo in the paper. Popular = interesting?

User Study - SeeDB vs Manual

16 participants with prior data analysis experience and visualization are asked to bookmark interesting visualization given an analytic task





Manual

Results - User Study

All participants preferred SeeDB to Manual.

79% of participants found the recommendations “Helpful” or “Very Helpful”

79% of participants indicated that SeeDB visualizations showed unexpected trends

One participant noted that SeeDB was “. . . great tool for proposing a set of initial queries for a dataset”

Results - User Study

	total_viz	num_bookmarks	bookmark_rate
MANUAL	6.3 ± 3.8	1.1 ± 1.45	0.14 ± 0.16
SEEDB	10.8 ± 4.41	3.5 ± 1.35	0.43 ± 0.23

Table 2: Aggregate Visualizations: Bookmarking Behavior Overview

Significant effect of tool on the number of bookmarks, $F(1,1) = 18.609$, $p < 0.001$

Significant effect of tool on bookmark rate, $F(1,1) = 10.034$, $p < 0.01$

No significant effect of dataset on number of bookmarks nor bookmark rate

Thank you!

SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics

Practitioner: Ashmita

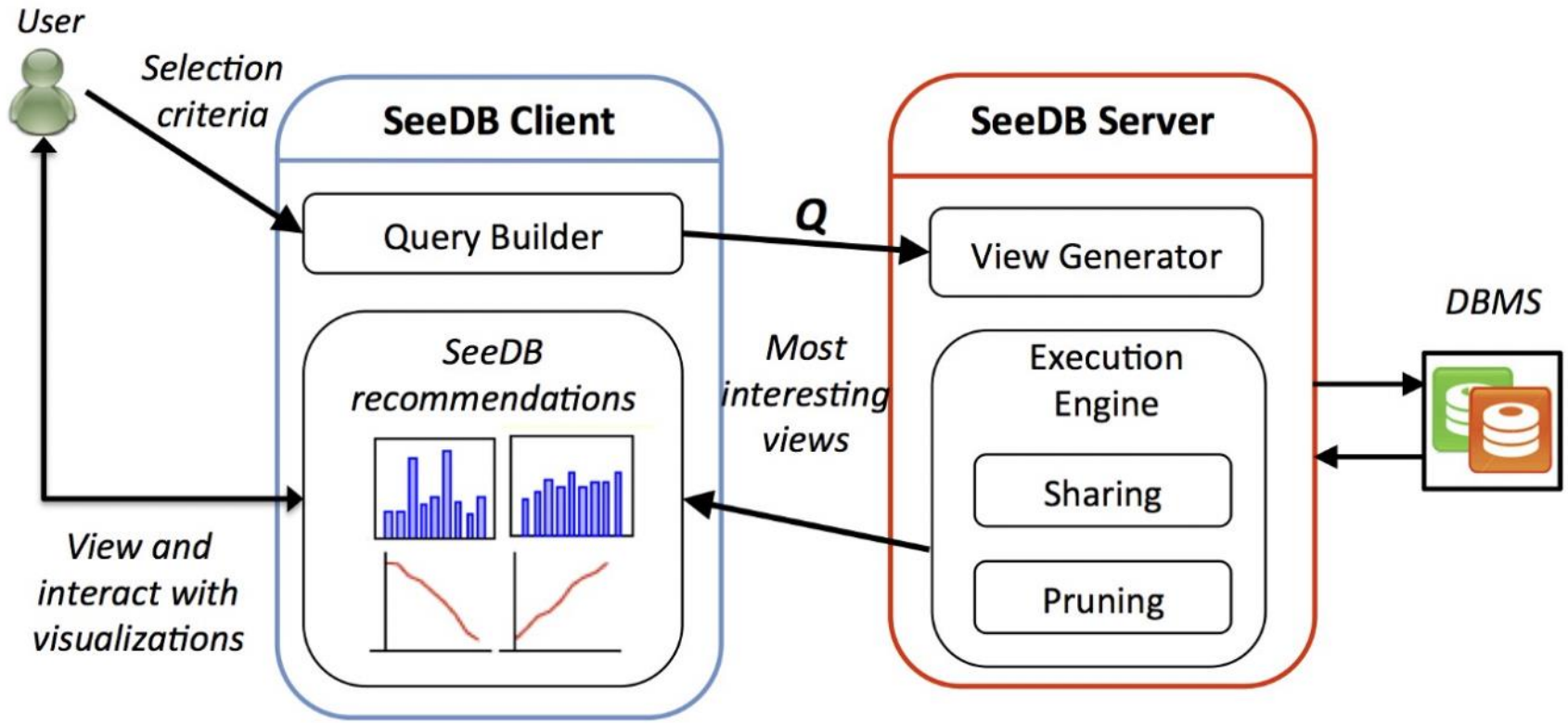
Summary

- User inputs a query which indicates the subset of data of interest.
- SEEDB automatically identifies and highlights the most interesting views of the query results using methods based on deviation.
- Optimizations to share computation and prune computation are implemented
- Middleware on top of any DB.

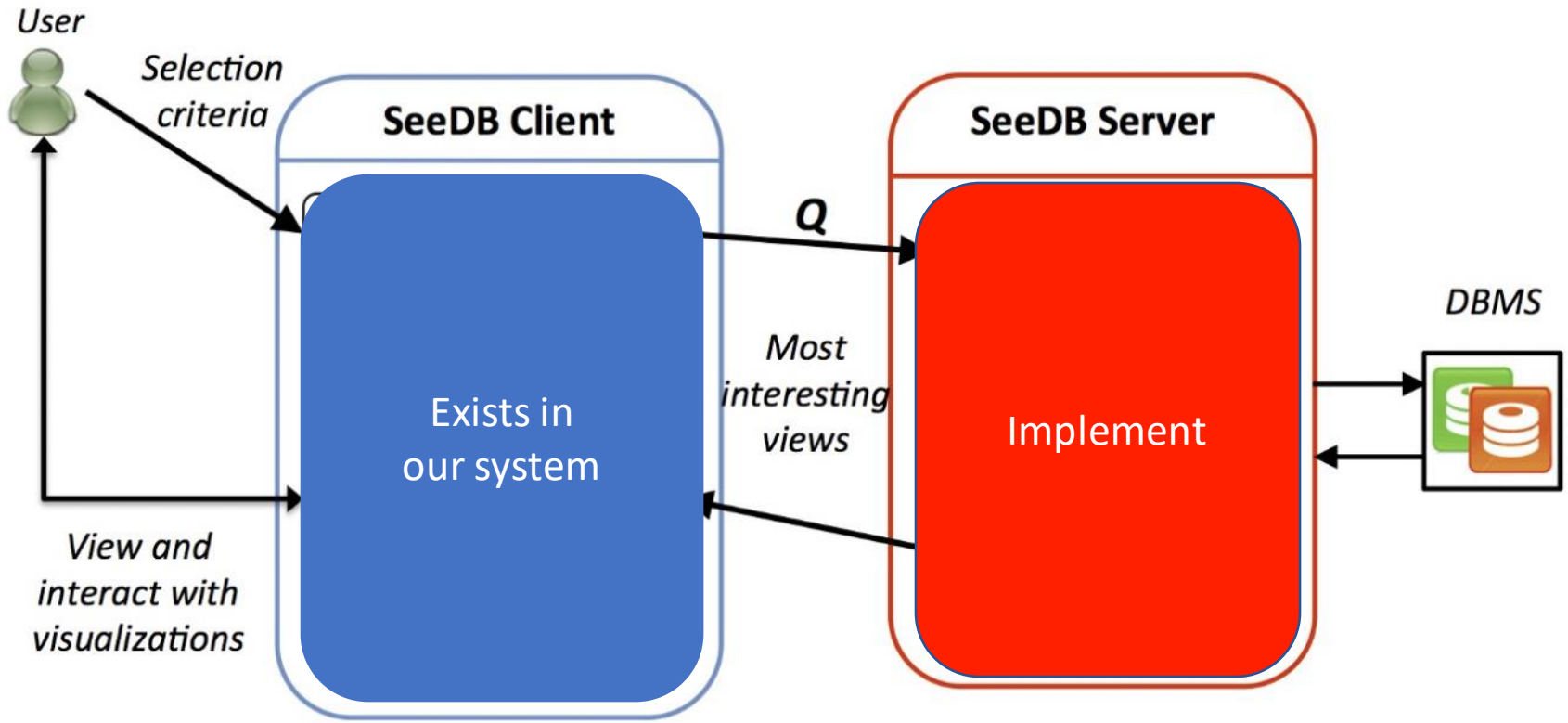
Integrating SeeDB

- Product - Interactive visualization tool like Tableau
- Implement SeeDB as it is advertised
- Middleware between the UI and the backend DB
- Technical Report contains more algorithmic details

Architecture



Architecture



Should we integrate?

- Improve the recommendations provided by our system
- The results presented in the paper are compelling
- User study indicates that users will benefit from this type of visualization recommendation
- Not much increase in compute/memory because of the optimizations
- **Negatives:**
  - Needs time and effort as it is not open sourced
  - Doesn’t support all types of charts – only bar and line, but should be extendible to any type

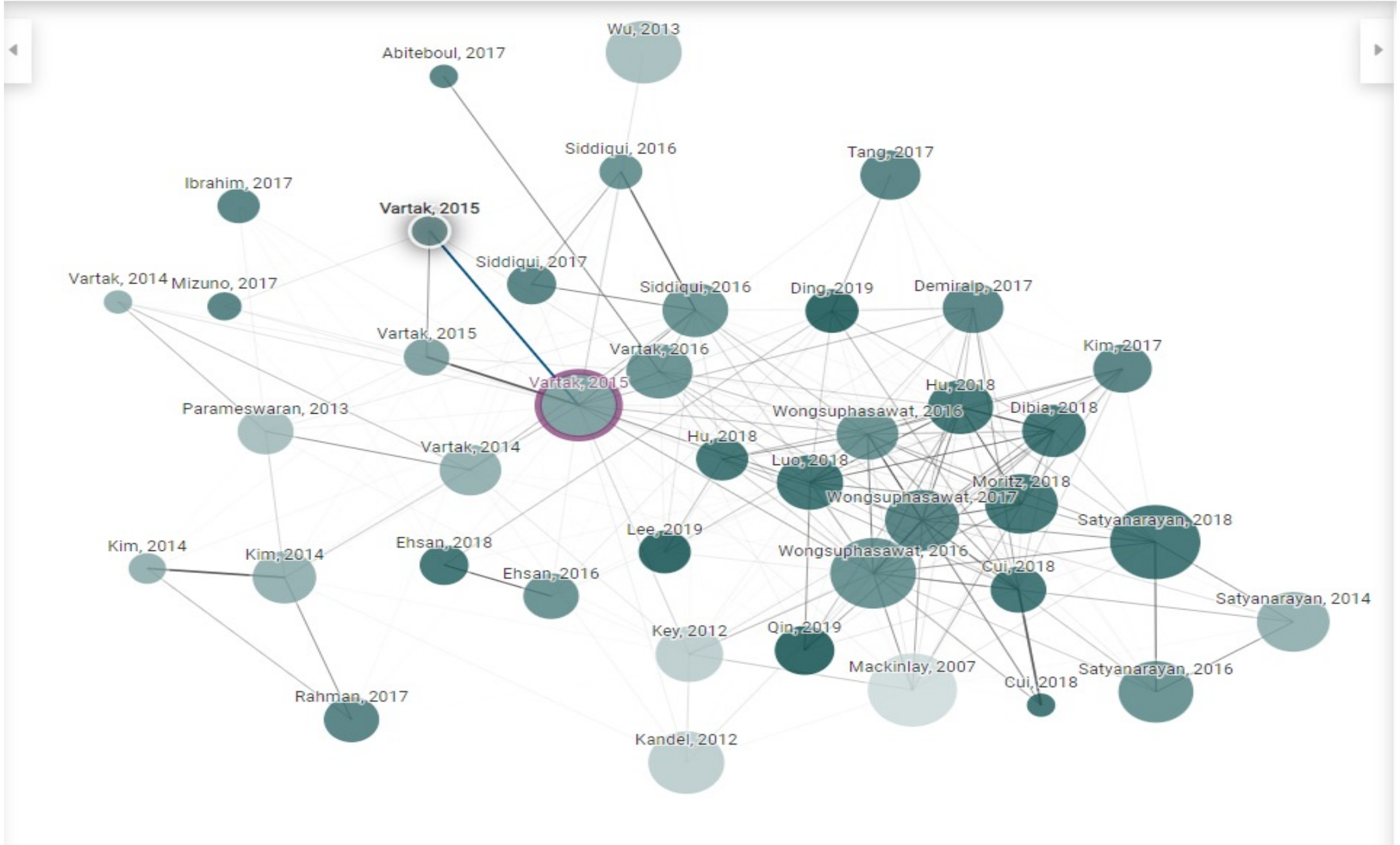
Thank You!

SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics

Archaeologist: Cangdi Li

Nov 3

ConnectedPapers

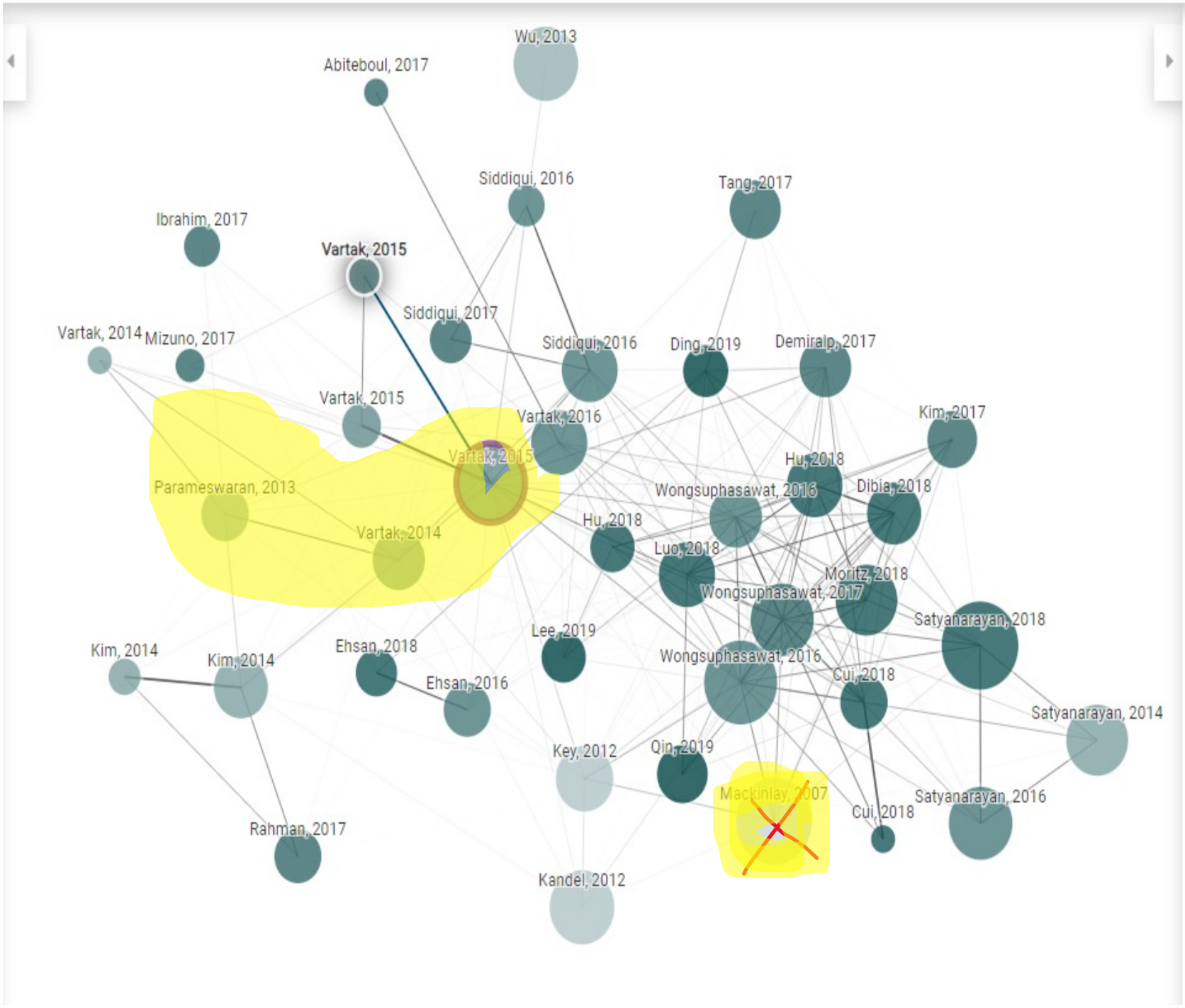


• Prior work

- Although the range is shown as 2007-2015, but the papers that have direct impact on this paper is only since 2012, and most of them are previous version of the SeeDB series.

- The SeeDB series:

- SeeDB: visualizing database queries efficiently Aditya G. Parameswaran, Neoklis Polyzotis, H. Garcia-Molina 2013, VLDB 2013
- SEEDB: Automatically Generating Query Visualizations Manasi Vartak, S. Madden, Aditya G. Parameswaran, Neoklis Polyzotis 2014, Proc. VLDB Endow
- SeeDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics Manasi Vartak, Sajjadur Rahman, S. Madden, Aditya G. Parameswaran, Neoklis Polyzotis 2015, Proc. VLDB Endow.



Growth of SeeDB

- **SeeDB: visualizing database queries efficiently** 2013
- A 4-page paper with 50 citations
- It propose a initial DBMS design that partially automates the task of finding the visualizations for a query, and it also gives recommendation of potentially "interesting" visualizations, with only Multi-Query Optimization (sharing).

- **SeeDB: Automatically Generating Query Visualizations** 2014
- A 4-page paper with 82 citations
- On top of the previous version, it proposes more optimization methods on pruning, and designed the structure of the tool to frontend and backend.

- **SeeDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics** 2015
- Same optimizations, and it was proposed as a middleware layer that can run on top of any DBMS. Completely implemented, evaluated with user study.

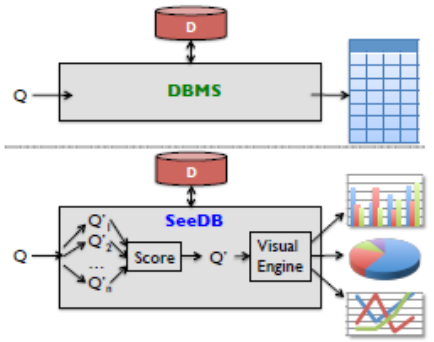


Figure 2: SeeDB Comparison with regular DBMS (The workflow for SeeDB is only conceptual and need not happen in that order.)

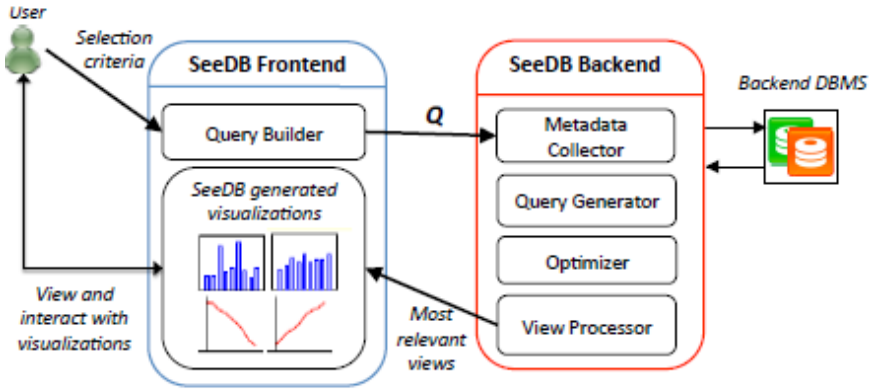
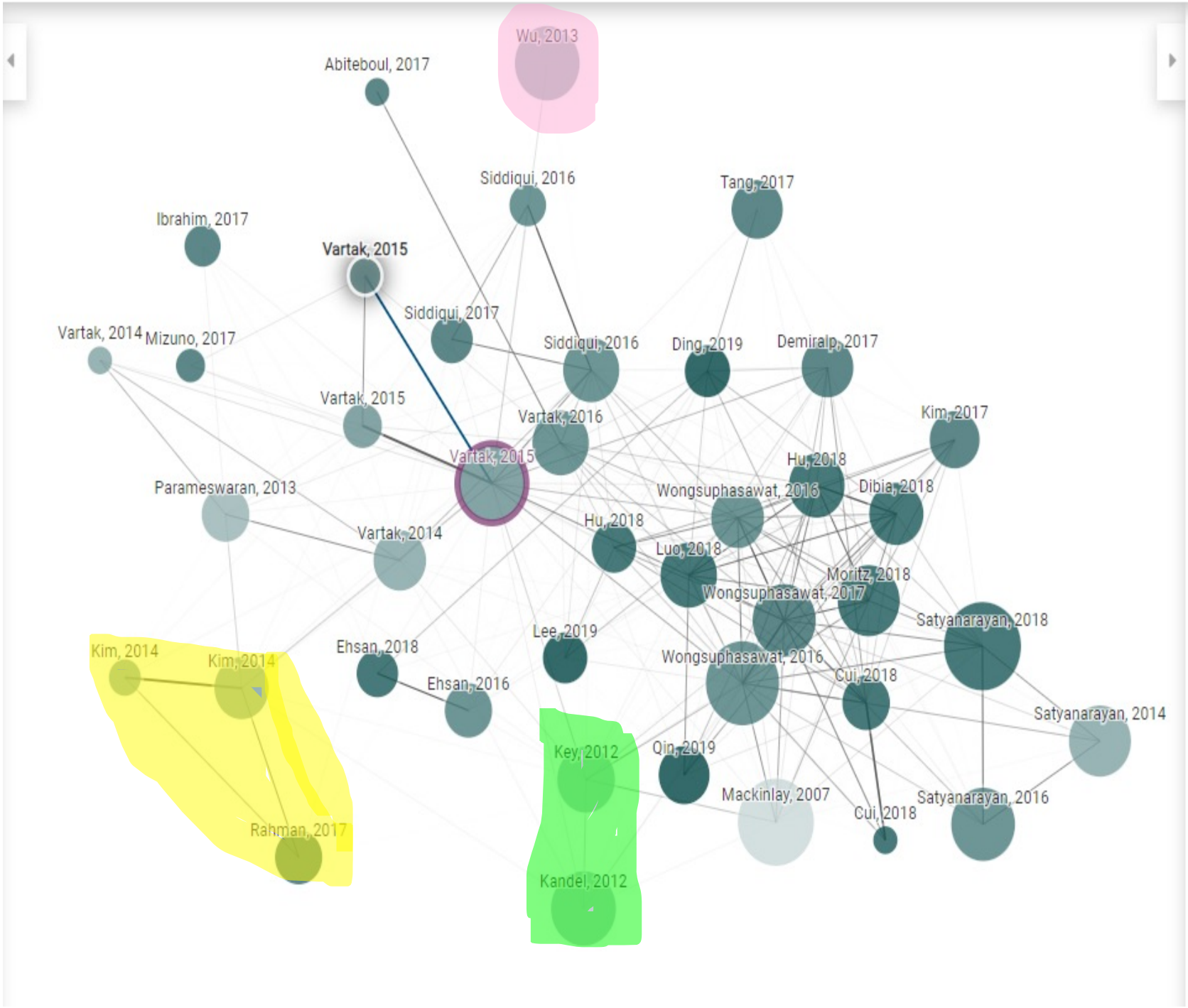


Figure 4: SeeDB Architecture

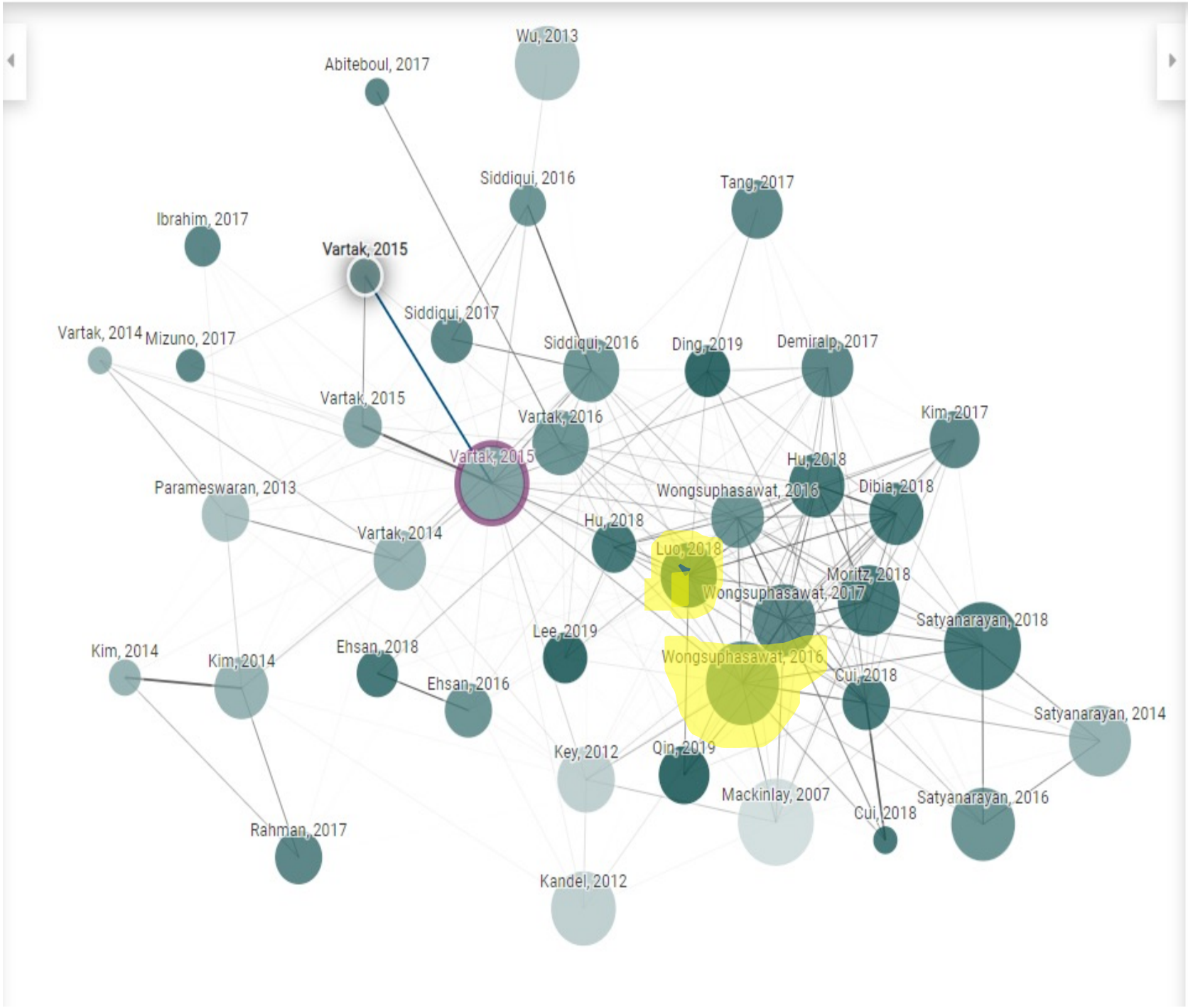
• Prior work

- Most of the impactful previous works focus on:
  - **Automated visualization reccomendation**
  - **sampling algorithm optimization.**
  - **explain outlier**
  - **Plenty of other papers that focus on visualization technique of multi-dimensional data.**



• Later work

- There’ s 2 major later works with big citation numbers:
- **DeepEye: Towards Automatic Data Visualization** Yuyu Luo, et al 2018 IEEE
- **Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations** Kanit Wongsuphasawat etal 2016, IEEE



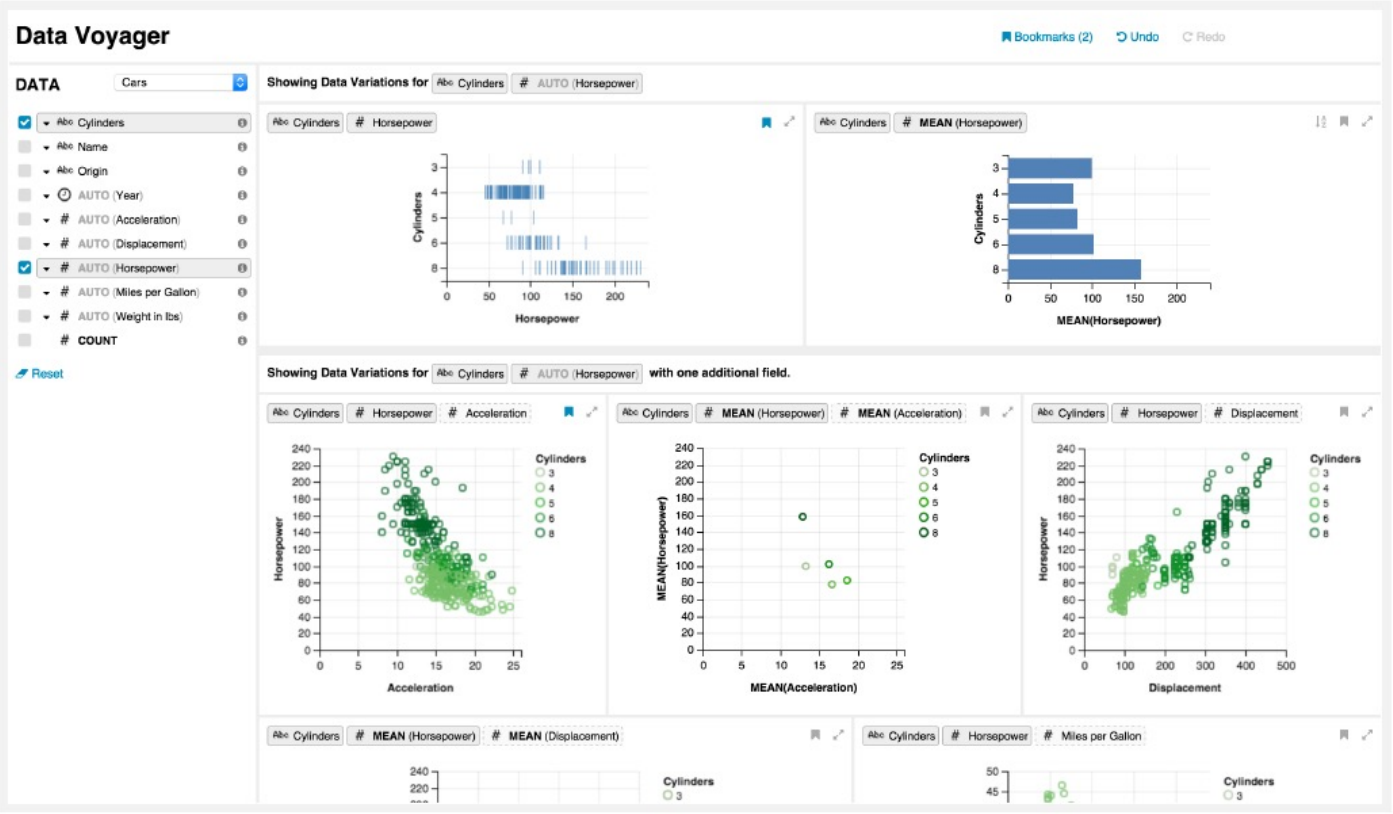
DeepEye: Towards Automatic Data Visualization Yuyu Luo, et al 2018 IEEE

- DeepEye is a novel system for automatic data visualization that tackles three problems:
- (1) Visualization recognition: given a visualization, is it “good” or “bad”?
- (2) Visualization ranking: given two visualizations, which one is “better”?
- (3) Visualization selection: given a dataset, how to find top-k visualizations?

- DEEPEYE addresses
- (1) by training a binary classifier to decide whether a particular visualization is good or bad.
- (2) from two perspectives:
  - (i) Machine learning: it uses a supervised learning-to-rank model to rank visualizations;
  - (ii) Expert rules: it relies on experts’ knowledge to specify partial orders as rules.
  - Moreover, a “boring” dataset may become interesting after data transformations
- (3) Extensive experiments verify the effectiveness of DEEPEYE.

Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations

- This paper seeks to complement manual chart construction with interactive navigation of a gallery of automatically-generated visualizations. It presents Voyager, a mixed-initiative system that supports faceted browsing of recommended charts chosen according to statistical and perceptual measures.



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

Peer Review

Shen En Chen

Summary of Contribution

- Built a **recommendation** system for **top-k most interesting visualizations**
- Leveraged **data deviation as cue** for “interestingness” and developed a **utility metric** to quantify it
- Used **share computation** across visualizations to avoid repeated scans on the same data
- Adopted **pruning techniques** to avoid computations on low-utility visualization
- Demonstrated the **accuracy** of utility estimation and **speeds** of recommendation on real and synthetic data.
- Conducted a **user study** to validates the usefulness of SeeDB in analytical workflows.

Strong Points

- **Auto-Identification and Recommendation**: Avoid human-in-the-loop trial-and-error process for finding interesting visualizations.
- **Highly Extensible**: Supports any SQL-compliant database systems, different utility metrics, different distance functions for the proposed deviation-based utility metrics.
- **Interactive Speeds**: Achieves 100x speedup overall.
- **Theoretical Guarantees**: Supports the design with strong/weak theoretical guarantees when possible.
- **Ablation Study**: Evaluations the speedups brought by different optimization components.
- **Supported by User Study**: Validates (1) the accuracy of the utility metric, (2) the capability to recommend interesting visualizations, and (3) the speedup on visual analysis.
- **Two Types of Accuracy**: (1) Accuracy against true utility scores and (2) Accuracy against user-voted interestingness
- **Detailed Controlled Settings and Analysis for User Study**: Conducted user studies with detailed protocols (e.g., same interface w/ vs w/o recommendation) and post-study analysis.

Opportunities for Improvement

- Better Prioritization and Summarization of Content: be more selective on the technical content presented and include the uncovered content in the technical report.
- Clarity on FIgures: improve the clarity of the figures with more annotations.
- Larger Sample Size for User Study: carry out user study with a larger sample size and more balanced demographics.
- Incorporation of Other Optimization Techniques (future direction): improve interactivity further by employing other techniques such as in-memory caching, sampling, and prefetching.
- Recommendations of Different Visualization (future direction): support non-bar chart visualizations.

ACCEPT

SeeDB

Review

Harshal Gajjar

8803-MDS Fall 22

Prof. Kexin Rong

Summary of contribution

- **What is the "utility" function** that makes a particular visualization important/informative.
- **Problems in naïve implementation** to find top-k informative visualizations which can make the interactive system unusable by increasing latency beyond 100s.
- **Optimizations in SeeDB server's execution engine** that reduces the number of queries, and the number of complete-db scans by the DBMS; this include sharing of data across queries and pruning of the number of queries (by removing underwhelming visualisations (as defined by the utility function))
- This is followed by **benchmarking on 7 datasets and a user study** on another 3 datasets to find qualitative and quantitive proofs for the usefulness of SeeDB.

Strong Points

"No existing system that we are aware of makes use of **variation from a reference** to recommend visualizations."
∵ Novel idea

Execution engine which iterates over phases while pruning options between phases and sharing data within phases.
∵ Generic idea and can be applied in several scenarios where there is an incoming stream of options and limited storage space

We note that this is the first time that bandit strategies have been applied to the problem of identifying interesting visualizations.
∵ Novel idea

Improvements

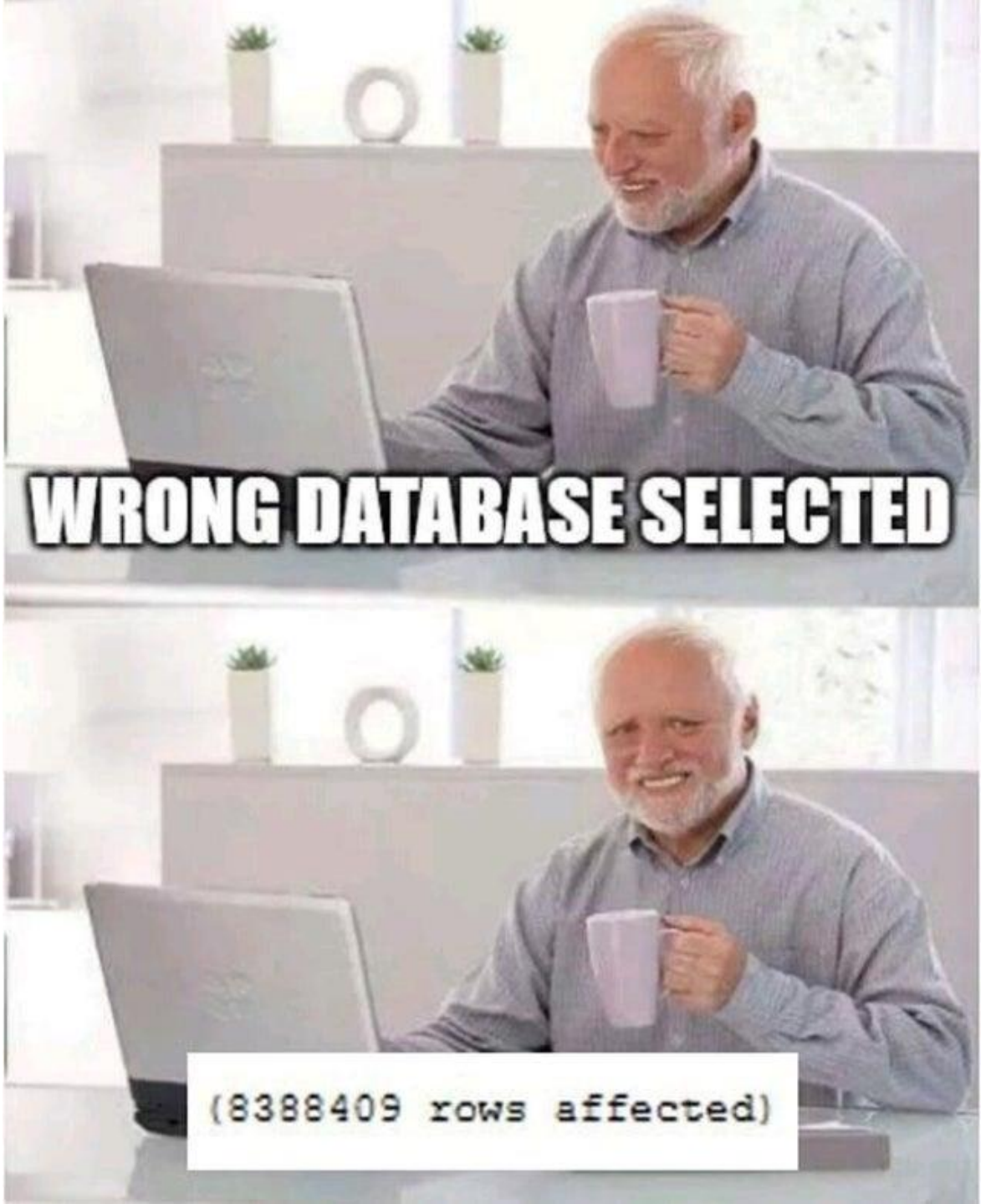
The current system only works with bar charts; that limitation was not explicitly stated in the paper.

The definition of problem (Problem 2.1 in the paper) was a bit non-intuitive, it should be self-sufficient, but it lacked definition of the terms a, m, f, and lacks reasoning for requiring Q.

The authors do not explore any other solutions to Multi-Armed bandit problem / explore-exploit problem or explain reasoning behind their algorithm

- **Epsilon-greedy strategy**:[31] The best lever is selected for a proportion $1 - \epsilon$ of the trials, and a lever is selected at random (with uniform probability) for a proportion ϵ. A typical parameter value might be $\epsilon = 0.1$, but this can vary widely depending on circumstances and predilections.
- **Epsilon-first strategy**[*citation needed*]: A pure exploration phase is followed by a pure exploitation phase. For N trials in total, the exploration phase occupies ϵN trials and the exploitation phase $(1 - \epsilon)N$ trials. During the exploration phase, a lever is randomly selected (with uniform probability); during the exploitation phase, the best lever is always selected.
- **Epsilon-decreasing strategy**[*citation needed*]: Similar to the epsilon-greedy strategy, except that the value of ϵ decreases as the experiment progresses, resulting in highly explorative behaviour at the start and highly exploitative behaviour at the finish.
- **Adaptive epsilon-greedy strategy based on value differences (VDBE)**: Similar to the epsilon-decreasing strategy, except that epsilon is reduced on basis of the learning progress instead of manual tuning (Tokic, 2010).[32] High fluctuations in the value estimates lead to a high epsilon (high exploration, low exploitation); low fluctuations to a low epsilon (low exploration, high exploitation). Further improvements can be achieved by a softmax-weighted action selection in case of exploratory actions (Tokic & Palm, 2011).[33]
- **Adaptive epsilon-greedy strategy based on Bayesian ensembles (Epsilon-BMC)**: An adaptive epsilon adaptation strategy for reinforcement learning similar to VBDE, with monotone convergence guarantees. In this framework, the epsilon parameter is viewed as the expectation of a posterior distribution weighting a greedy agent (that fully trusts the learned reward) and uniform learning agent (that distrusts the learned reward). This posterior is approximated using a suitable Beta distribution under the assumption of normality of observed rewards. In order to address the possible risk of decreasing epsilon too quickly, uncertainty in the variance of the learned reward is also modeled and updated using a normal-gamma model. (Gimelfarb et al., 2019).[34]
- **Contextual-Epsilon-greedy strategy**: Similar to the epsilon-greedy strategy, except that the value of ϵ is computed regarding the situation in experiment processes, which lets the algorithm be Context-Aware. It is based on dynamic exploration/exploitation and can adaptively balance the two aspects by deciding which situation is most relevant for exploration or exploitation, resulting in highly explorative behavior when the situation is not critical and highly exploitative behavior at critical situation.[35]

Accept 🎉



Discussion

How are recommendation criteria similar/different in visual recommendation systems and traditional recommendation systems

- Relevance
- Novelty
- Non-obviousness
- Diversity
- Coverage

Discussion

What are opportunities to speed up visualization recommendation computations?

- Offline
- Online
- Approximation

Next class

Lux: Always-on Visualization Recommendations for Exploratory Dataframe Workflows

Authors: Sahil, Gaurav

Reviewer: Bojun

Archaeologist: Gaurav

Practioner: Cuong