Speeding Up Data Science:
From a Data Management Perspective

JIANNAN WANG
SIMON FRASER UNIVERSITY

Database Group @ UBC
July 24, 2017

Our Lab's Mission

Speeding Up Data Science

Computer Science vs. Data Science

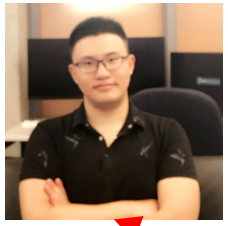
What	When	Who	Goal
Computer Science	1950-	Software Engineer	Write software to make computers work

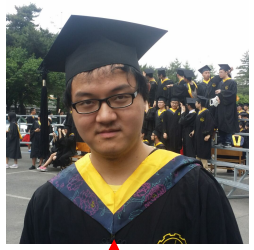
Plan → Design → Develop → Test → Deploy → Maintain

What	When	Who	Goal
Data Science	2010-	Data Scientist	Extract insights from data to answer questions

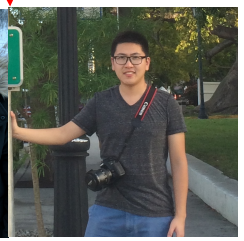
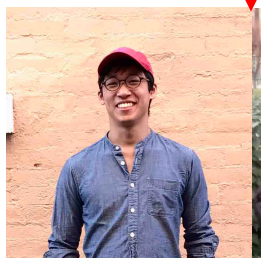
Collect → Clean → Integrate → Analyze → Visualize → Communicate

Lab Members

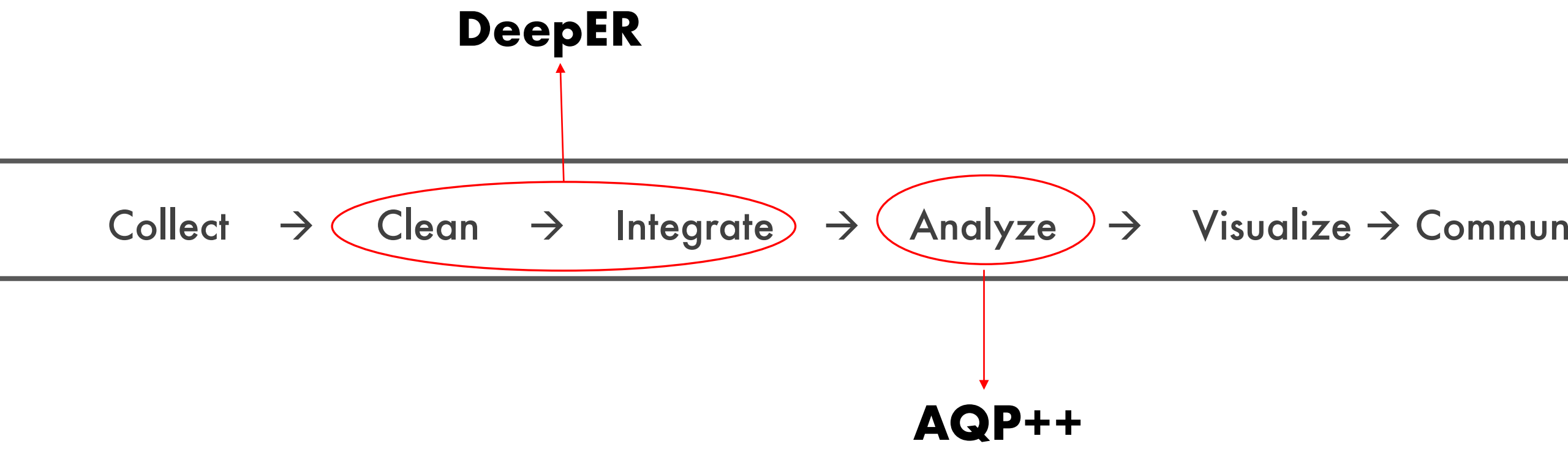


Collect → Clean → Integrate → Analyze → Visualize → Commun

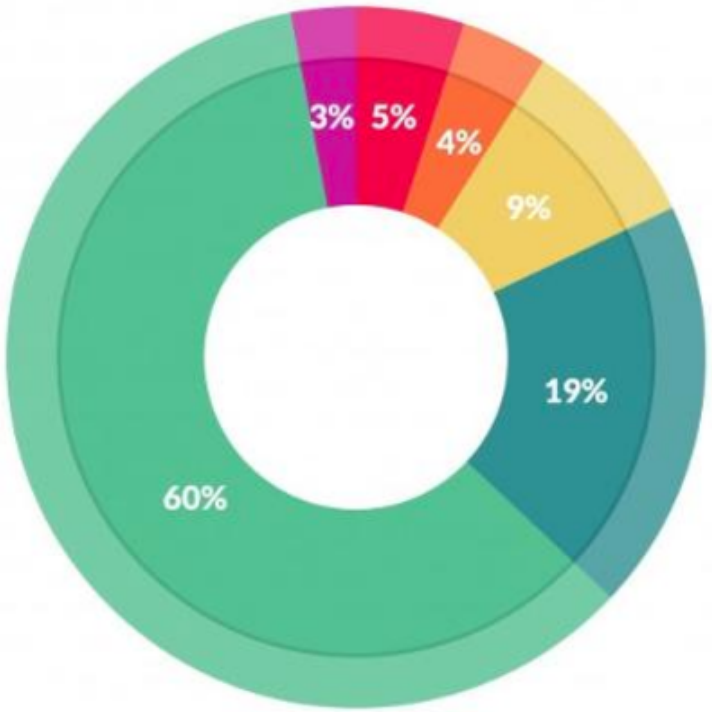




Today's Talk


DeepER
Collect
Clean
Integrate
Analyze
Visualize
Commun
AQP++

Where is the bottleneck?

Data scientists spend **60%** of their time on cleaning and organizing data.



What data scientists spend the most time doing

- *Building training sets: 3%*
- *Cleaning and organizing data: 60%*
- *Collecting data sets; 19%*
- *Mining data for patterns: 9%*
- *Refining algorithms: 4%*
- *Other: 5%*

(Source: Cloudera)

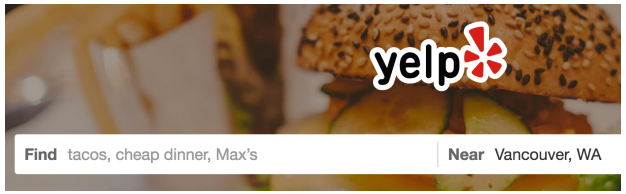
DeepER's Key Idea

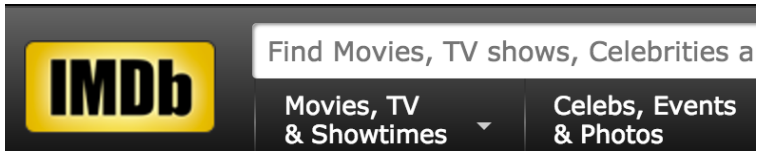
Leveraging **Deep Web** To Speed Up

Data Cleaning and **Data Enrichment**

Deep Web

Hidden Database





Invaluable External Resource

- Big: Consisting of a substantial number of entities
- Rich: Having rich Information about each entity
- High-quality. Being trustful and up-to-date

A real-world example

Customer Location Data

User ID	Location	Zip Code	Frequency
U12345	Lotus of Siam	891004	20 visits

1. Data Enrichment

User ID	Location	Zip Code	Frequency	Category
U12345	Lotus of Siam	891004	20 visits	Thai, Wine Bars

2. Data Cleaning

User ID	Location	Zip Code	Frequency
U12345	Lotus of Siam	89104	20 visits

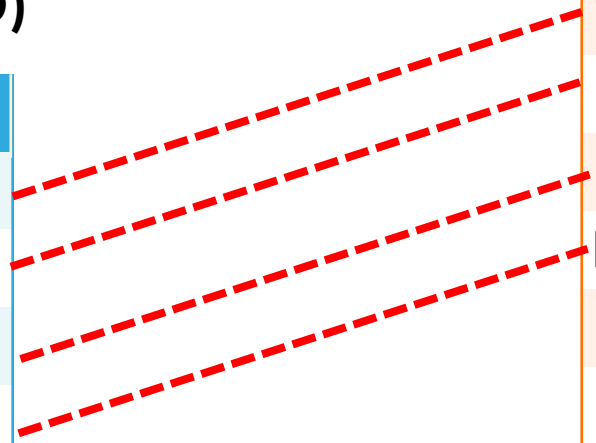
Entity Resolution

Hidden Database (H)

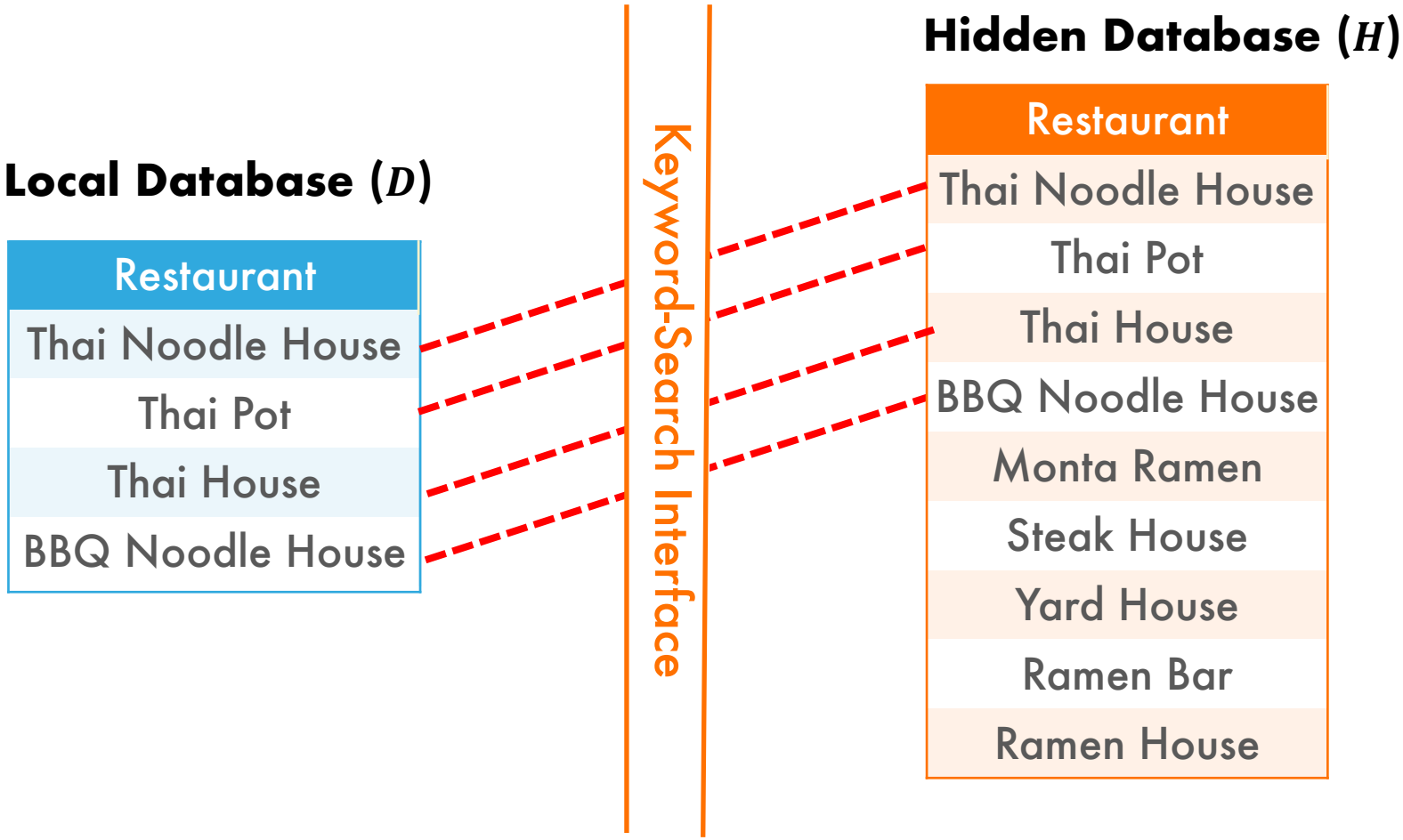
Restaurant
Thai Noodle House
Thai Pot
Thai House
BBQ Noodle House
Monta Ramen
Steak House
Yard House
Ramen Bar
Ramen House

Local Database (D)

Restaurant
Thai Noodle House
Thai Pot
Thai House
BBQ Noodle House



Deep Entity Resolution



Keyword Search

1. Conjunctive Query
2. Top-k Constraint
3. Deterministic Query Processing

Local and Hidden DBs

1. D has no duplicate record
2. H has no duplicate record

New Challenges

Limited Query Budget

- Yelp API is restricted to 25,000 free requests per day
- Goolge Maps API only allows 2,500 free requests per day

Top-k Constraint

- Return top-k results based on an unknown ranking function

NaiveCrawl

Enumerate each record in D and then generate a query to cover it

Limitation

- Cover one record at a time

Restaurant
Thai Noodle House
Thai Pot
Thai House
BBQ Noodle House

Keyword Queries

q_1 = "Thai Noodle House"
q_2 = "Thai Pot"
q_3 = "Thai House"
q_4 = "BBQ Noodle House"

FullCrawl

1. Try to crawl the entire hidden database $H_{crawled}$
2. Perform entity resolution between D and $H_{crawled}$

Limitation

- Not aware of the existence of a local database

SmartCrawl

Insight 1. Query Sharing

Restaurant
Thai Noodle House
Thai Pot
Thai House
BBQ Noodle House

Keyword Queries

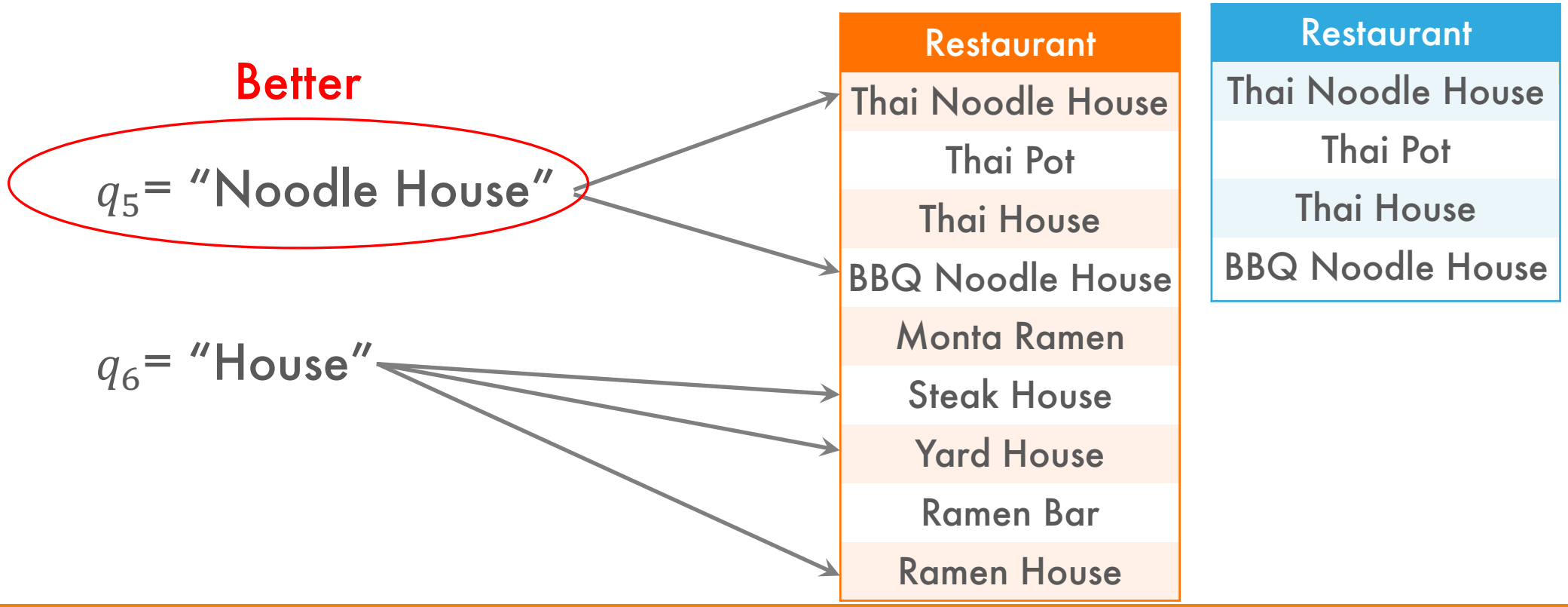
q_1 = "Thai Noodle House"
q_2 = "Thai Pot"
q_3 = "Thai House"
q_4 = "BBQ Noodle House"

q_5 = "Noodle House"
q_6 = "House"
q_7 = "Thai"

Cover multiple records at a time

SmartCrawl

Insight 2. Local-database-aware crawling



SmartCrawl Framework

1. Generate a query pool Q
2. Select at most b queries from Q such that $|H_{crawled} \cap D|$ is maximized
3. Perform entity resolution between $H_{crawled}$ and D

Query Pool Generation

Basic Idea

- Only need to consider the queries in D

Restaurant
Thai Noodle House
Thai Pot
Thai House
BBQ Noodle House

q = "Sushi"

Keyword Queries

q_1 = "Thai Noodle House"
q_2 = "Thai Pot"
q_3 = "Thai House"
q_4 = "BBQ Noodle House"
q_5 = "Noodle House"
q_6 = "House"
q_7 = "Thai"

SmartCrawl Framework

1. Generate a query pool Q
2. Select at most b queries from Q such that $|H_{crawled} \cap D|$ is maximized
3. Perform entity resolution between $H_{crawled}$ and D

Query Selection

NP-Hard Problem

- Can be proved by a reduction from the maximum coverage problem

Greedy Algorithm

- Suffers from a chicken-and-egg problem

Sampling and Estimation

Deep Web Sampling [Zhang et al. SIGMOD 2011]

- H_s is a random sample of H
- θ is the sampling ratio

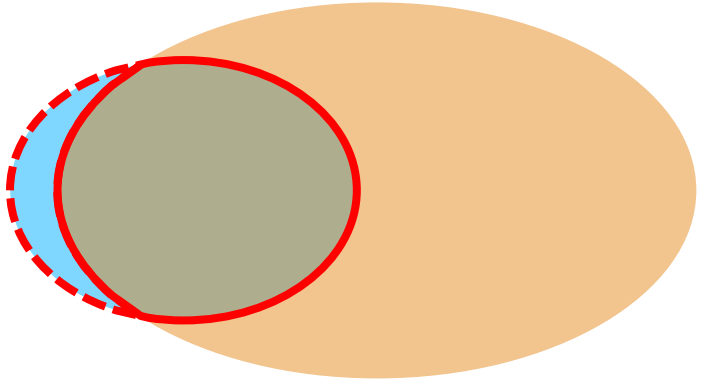
Two classes of queries

- Solid Query
- Overflowing Query

IF $\frac{|q(H_s)|}{\theta} \leq k$ **THEN**
    q is a solid query
ELSE
    q is an overflowing query
END

Solid Query

How to estimate $|q(D) \cap q(H)|$**?**



Unbiased Estimator: $\frac{|q(D) \cap q(H_s)|}{\theta}$

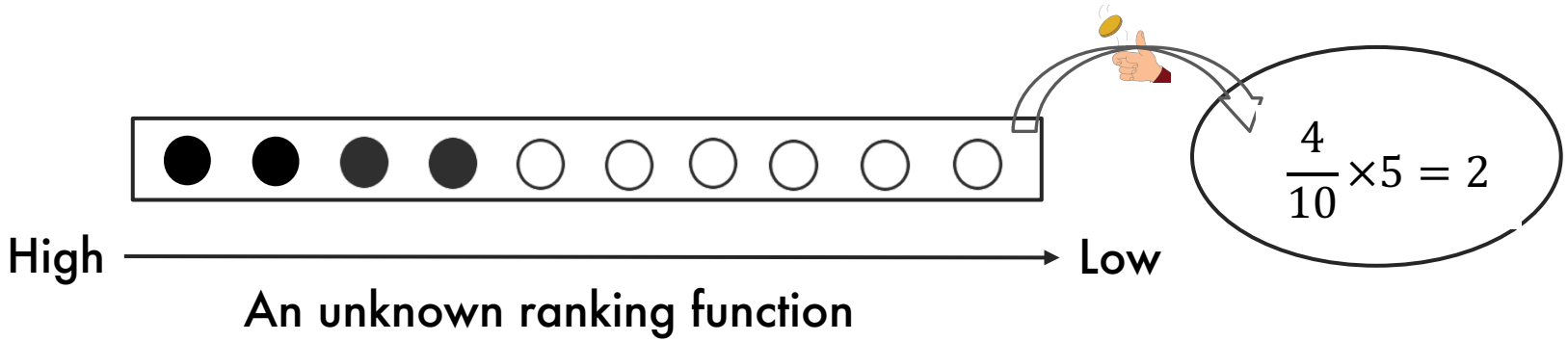
Key Observation: $|q(D) - q(H)|$ ***is small***

Biased Estimator: $|q(D)|$

Overflowing Query

How to estimate $|q(D) \cap q(H)_{\boldsymbol{k}}|$**?**

Basic Idea



How to estimate $\frac{k}{|q(H)|} \times |q(D) \cap q(H)|$**?**

A Summary of Estimators

	Unbiased	Biased (w/ small biases)
Solid	$\frac{\lvert q(\mathcal{D}) \cap q(\mathcal{H}_s) \rvert}{\theta}$	$\lvert q(\mathcal{D}) \rvert$
Overflowing	$\lvert q(\mathcal{D}) \cap q(\mathcal{H}_s) \rvert \cdot \frac{k}{\lvert q(\mathcal{H}_s) \rvert}$	$\lvert q(\mathcal{D}) \rvert \cdot \frac{k\theta}{\lvert q(\mathcal{H}_s) \rvert}$

Other Contributions

1. Theoretical Analysis

2. Efficient Implementations

3. Inadequate Sample Size

4. Fuzzy Matching

Experimental Settings

Simulation

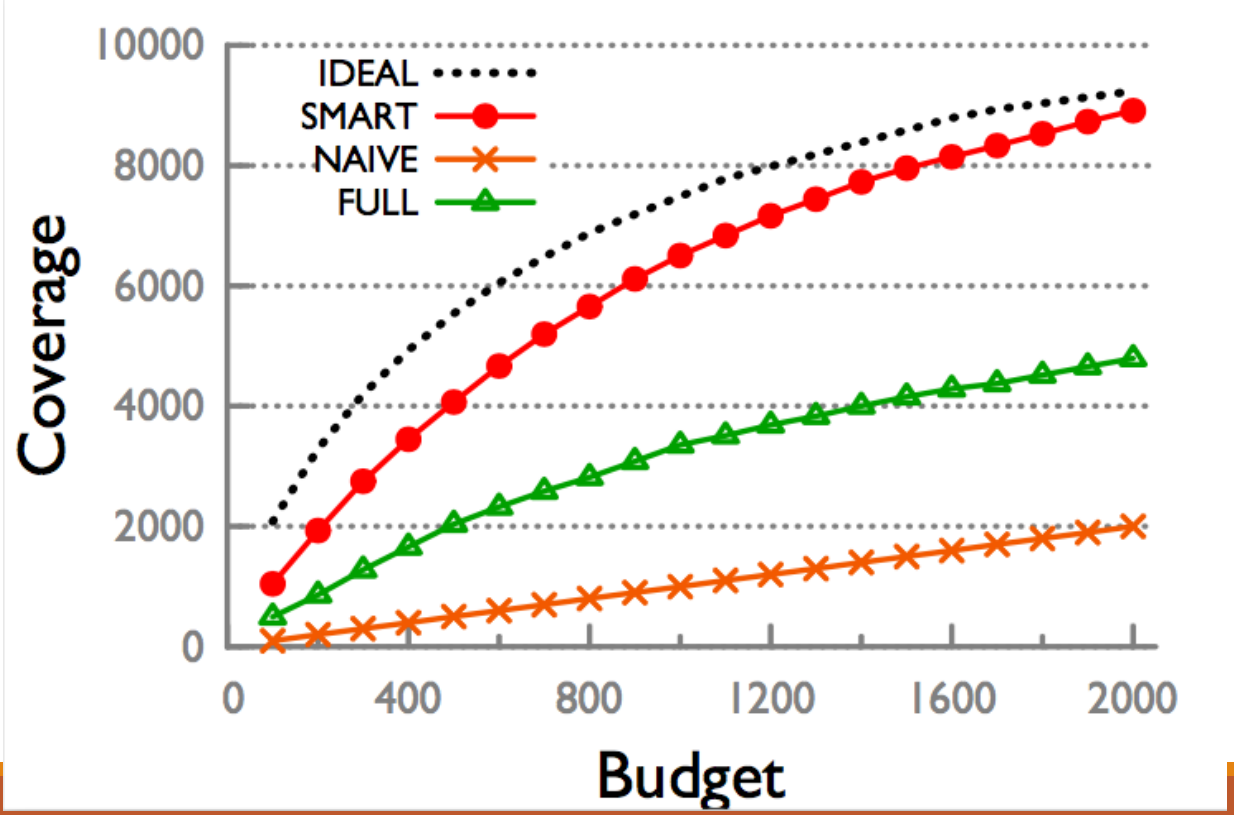
- Hidden Database: DBLP
- Local Database: Database Researchers' publications

Real-world

- Hidden Database: Yelp
- Local Database: 3000 restaurants in AZ
- Ground-Truth: Manually Labeled

DBLP

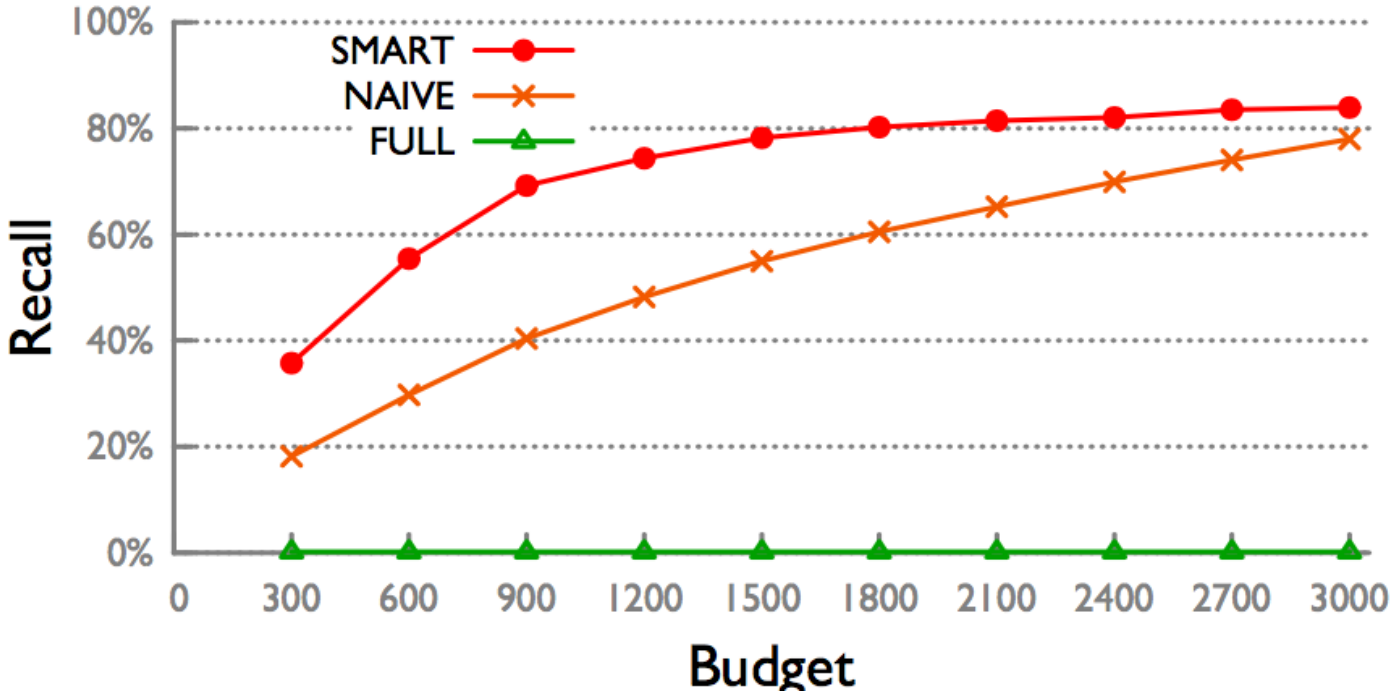
$$|D| = 10{,}000, |H| = 100{,}000, K = 100, \theta = 0.2\%$$



1. SmartCrawl performed very well with a small sampling ratio
2. SmartCrawl outperforms straightforward solutions

Yelp

$$|D| = 3{,}000, |H| \approx 250{,}000, K = 50, \theta = 0.2\%$$



1. SmartCrawl outperformed straightforward solutions
2. SmartCrawl was more robust to the fuzzy-matching situation than NaiveCrawl

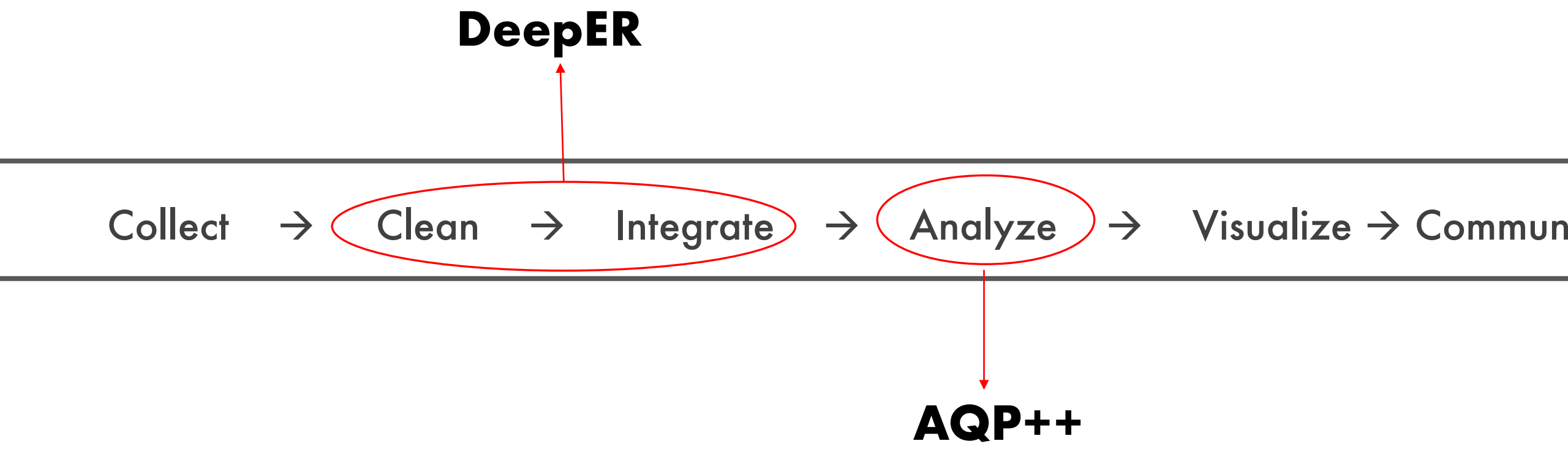
DeepER Conclusion

We are the first to study the DeepER problem

SmartCrawl outperforms NaiveCrawl and FullCrawl by a factor of $2 - 7\times$

SmartCrawl is more robust to the fuzzy-matching situation than NaiveCrawl

Today's Talk


DeepER
Collect → Clean → Integrate → Analyze → Visualize → Commun
AQP++

Interactive Analytics

tableau®

jupyter

Power BI

databricks

Apache Zeppelin

Two Separate Ideas

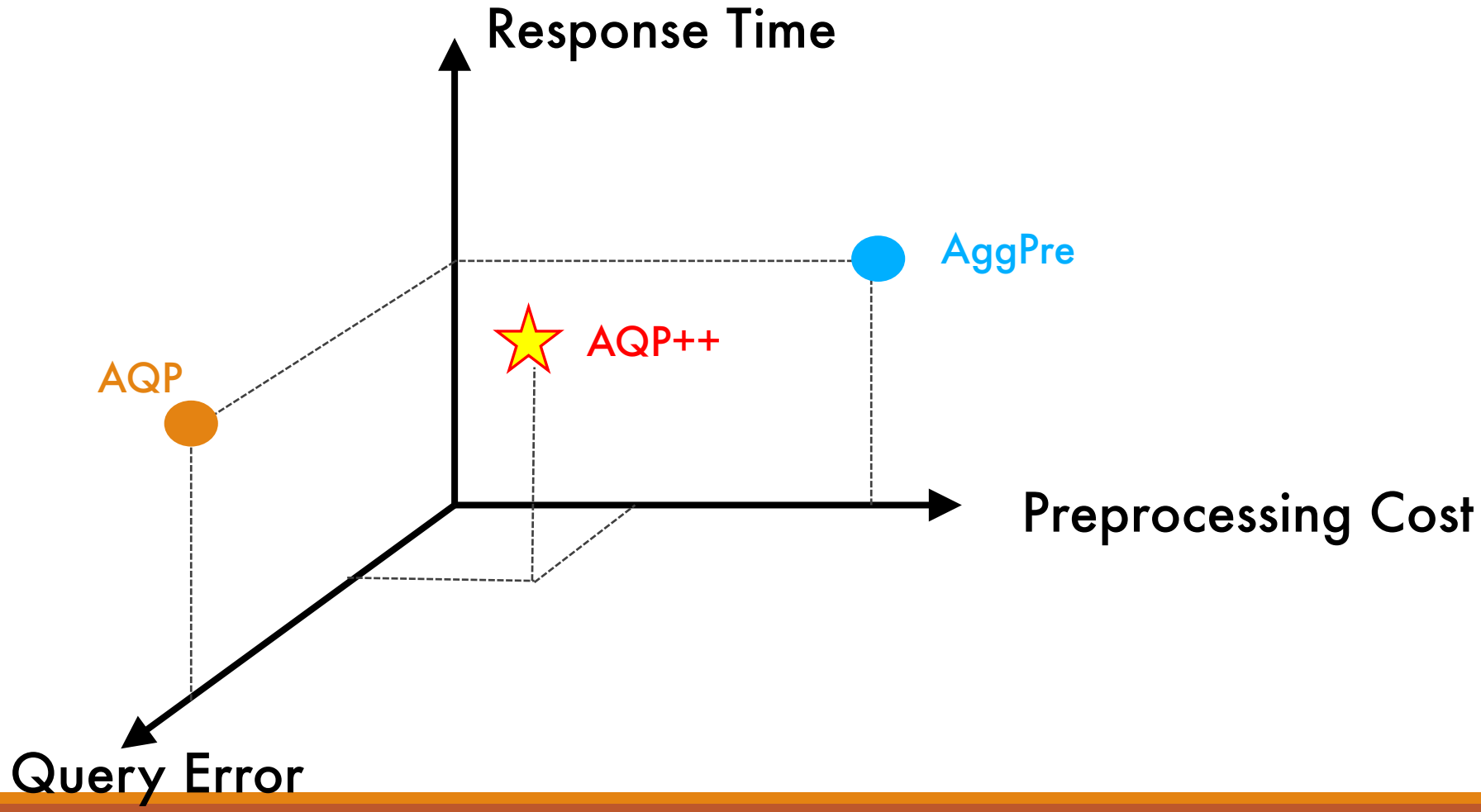
Approximate Query Processing (AQP)

- Trade answer quality for interactive response time

Aggregate Precomputation (AggPre)

- Trade preprocessing cost for interactive response time

AQP++: Connecting AQP with AggPre



Experimental Result

TPCD-Skew (10GB, z = 2, 0.3% sample)

	Preprocessing Cost		Response Time	Answer Quality	
	Space	Time		Avg Err.	Mdn Err.
AQP	30.72 MB	1.71 min	0.11 sec	3.28%	2.89%
AggPre	10.91 TB	> 1 day	< 0.01 sec	0.00%	0.00%
AQP++	30.74 MB	2.97 min	0.12 sec	0.41%	0.28%

On-going Projects

Students	Stages	Projects
Pei & Yongjun	Clean & Integrate	DeepER: Deep Entity Resolution
Jinglin Peng	Analyze & Visualize	AQP++: Connecting AQP with AggPre
Mathew & Mohamad	Clean & Analyze	Data Cleaning Advisor for ML
Changbo & Ruochen	Collect & Clean	Live Video Highlight Detection using Crowdsourced User Comments
Nathan Yan	Clean	Data Cleaning with Statistical Constraints
Young Woo	Analyze	ML Explanation and Debugging

Two Take-away Messages

Data scientists waste a lot of time on data processing

Collect → Clean → Integrate → Analyze → Visualize → Communicate

Database researchers play a central role to speed up data science