Database Matching Under Column Deletions

Serhat Bakirtas, Elza Erkip

New York University

ISIT 2021

Motivation

- A boom in data collection.

Motivation

- A boom in data collection.
- Potentially-sensitive data published or sold after anonymization

Motivation

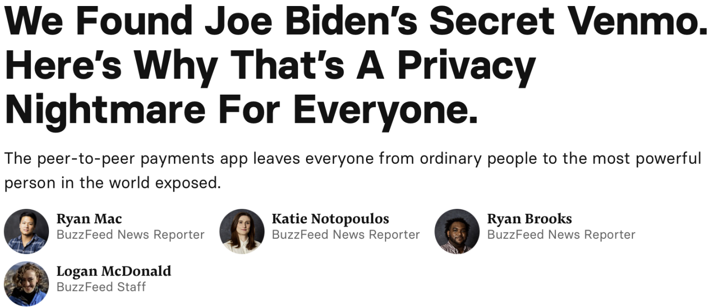
- A boom in data collection.
- Potentially-sensitive data published or sold after anonymization
- Risk of privacy leakage

Motivation

- A boom in data collection.
- Potentially-sensitive data published or sold after anonymization
- Risk of privacy leakage
- Anonymization is not enough on its own!
  - Correlated data → De-anonymization!

Motivation

- A boom in data collection.
- Potentially-sensitive data published or sold after anonymization
- Risk of privacy leakage
- Anonymization is not enough on its own!
  - Correlated data → De-anonymization!



We Found Joe Biden's Secret Venmo. Here's Why That's A Privacy Nightmare For Everyone.

The peer-to-peer payments app leaves everyone from ordinary people to the most powerful person in the world exposed.

Ryan Mac BuzzFeed News Reporter

Katie Notopoulos BuzzFeed News Reporter

Ryan Brooks BuzzFeed News Reporter

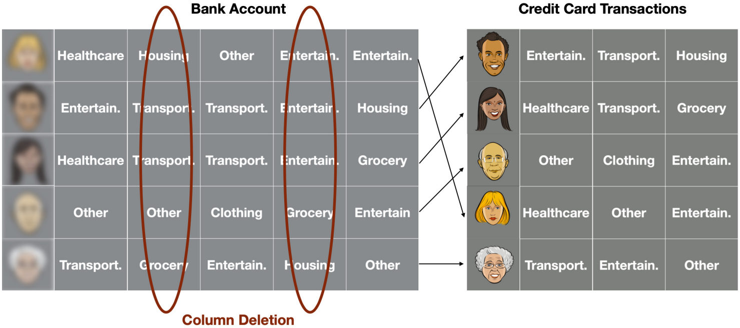
Logan McDonald BuzzFeed Staff

Motivation: Our Work

- Database Matching

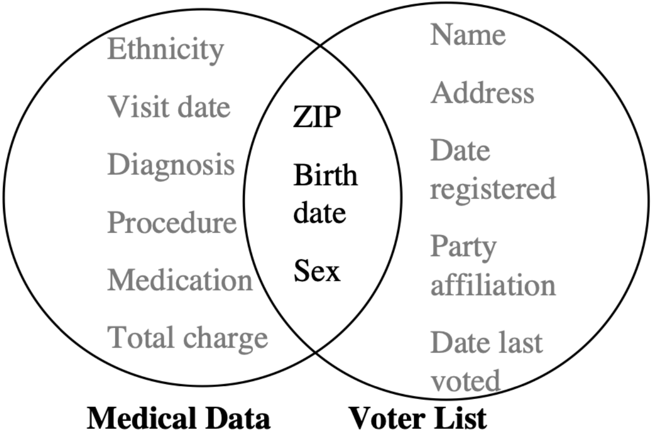
Motivation: Our Work

- Database Matching
- Column deletions
  - Synchronization errors in time-indexed databases.



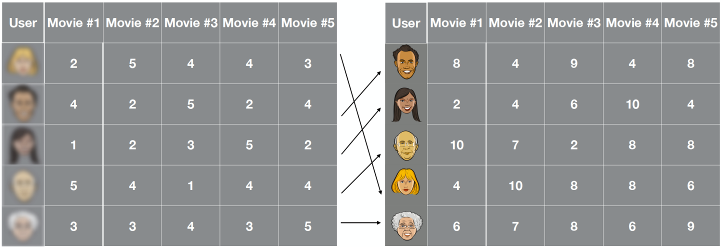
Previous Work: Practical Attacks on Real Data

- [Sweeney, 2002]
  - Deanonymization of MA hospital discharge database using public voter database (worth $20!)



Previous Work: Practical Attacks on Real Data

- [Narayanan and Shmatikov, 2008]
  - Deanonymization of Netflix movie ratings using IMDB reviews



Netflix Prize Dataset

User	Movie #1	Movie #2	Movie #3	Movie #4	Movie #5
	2	5	4	4	3
	4	2	5	2	4
	1	2	3	5	2
	5	4	1	4	4
	3	3	4	3	5

IMDB Reviews

User	Movie #1	Movie #2	Movie #3	Movie #4	Movie #5
	8	4	9	4	8
	2	4	6	10	4
	10	7	2	8	8
	4	10	8	8	6
	6	7	8	6	9

Previous Work: Practical Attacks on Real Data

- [Naini, et al., 2012]
  - User identification from geolocation data

(a) Unlabeled histograms (Day 1)

User	Location		
	Dorm.	Rest.	Lib.
?	75%	15%	10%
?	31%	30%	39%
?	15%	15%	70%
?	15%	65%	20%

(b) Labeled histograms (Day 2)

User	Location		
	Dorm.	Rest.	Lib.
John	33%	33%	34%
Jill	70%	20%	10%
Mary	15%	60%	25%
Mike	15%	20%	65%

Previous Work: Theoretical Limits

[Shirani, Garg, and Erkip, 2019]

$\mathcal{C}^{(1)}$

User ID	Attribute Vector				
$\Theta^{(1)}(1)$	$X^{(1)}_{1,1}$	$X^{(1)}_{1,2}$	•	•	$X^{(1)}_{1,n}$
$\Theta^{(1)}(2)$	$X^{(1)}_{2,1}$	$X^{(1)}_{2,2}$	•	•	$X^{(1)}_{2,n}$
• •	• •	• •	• •	• •	• •
$\Theta^{(1)}(m)$	$X^{(1)}_{m,1}$	$X^{(1)}_{m,2}$	•	•	$X^{(1)}_{m,n}$

$\mathcal{C}^{(2)}$

User ID	Attribute Vector				
$\Theta^{(2)}(1)$	$X^{(2)}_{1,1}$	$X^{(2)}_{1,2}$	•	•	$X^{(2)}_{1,n}$
$\Theta^{(2)}(2)$	$X^{(2)}_{2,1}$	$X^{(2)}_{2,2}$	•	•	$X^{(2)}_{2,n}$
• •	• •	• •	• •	• •	• •
$\Theta^{(2)}(m)$	$X^{(2)}_{m,1}$	$X^{(2)}_{m,2}$	•	•	$X^{(2)}_{m,n}$

- Databases as $m_n \times n$ random matrices
  - Matching rows $\sim f_{X^{(1),n},X^{(2),n}}$

[Shirani, Garg, and Erkip, 2019]

$\mathcal{C}^{(1)}$

User ID	Attribute Vector				
$\Theta^{(1)}(1)$	$X^{(1)}_{1,1}$	$X^{(1)}_{1,2}$	•	•	$X^{(1)}_{1,n}$
$\Theta^{(1)}(2)$	$X^{(1)}_{2,1}$	$X^{(1)}_{2,2}$	•	•	$X^{(1)}_{2,n}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{(1)}(m)$	$X^{(1)}_{m,1}$	$X^{(1)}_{m,2}$	•	•	$X^{(1)}_{m,n}$

$\mathcal{C}^{(2)}$

User ID	Attribute Vector				
$\Theta^{(2)}(1)$	$X^{(2)}_{1,1}$	$X^{(2)}_{1,2}$	•	•	$X^{(2)}_{1,n}$
$\Theta^{(2)}(2)$	$X^{(2)}_{2,1}$	$X^{(2)}_{2,2}$	•	•	$X^{(2)}_{2,n}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{(2)}(m)$	$X^{(2)}_{m,1}$	$X^{(2)}_{m,2}$	•	•	$X^{(2)}_{m,n}$

- Databases as $m_n \times n$ random matrices
  - Matching rows $\sim f_{X^{(1),n},X^{(2),n}}$
- Database growth rate: $R = \lim_{n\to\infty} \frac{1}{n} \log m$

Previous Work: Theoretical Limits

[Shirani, Garg, and Erkip, 2019]

$\mathcal{C}^{(1)}$

User ID	Attribute Vector				
$\Theta^{(1)}(1)$	$X^{(1)}_{1,1}$	$X^{(1)}_{1,2}$	•	•	$X^{(1)}_{1,n}$
$\Theta^{(1)}(2)$	$X^{(1)}_{2,1}$	$X^{(1)}_{2,2}$	•	•	$X^{(1)}_{2,n}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{(1)}(m)$	$X^{(1)}_{m,1}$	$X^{(1)}_{m,2}$	•	•	$X^{(1)}_{m,n}$

$\mathcal{C}^{(2)}$

User ID	Attribute Vector				
$\Theta^{(2)}(1)$	$X^{(2)}_{1,1}$	$X^{(2)}_{1,2}$	•	•	$X^{(2)}_{1,n}$
$\Theta^{(2)}(2)$	$X^{(2)}_{2,1}$	$X^{(2)}_{2,2}$	•	•	$X^{(2)}_{2,n}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{(2)}(m)$	$X^{(2)}_{m,1}$	$X^{(2)}_{m,2}$	•	•	$X^{(2)}_{m,n}$

- Databases as $m_n \times n$ random matrices
  - Matching rows $\sim f_{X^{(1),n},X^{(2),n}}$
- Database growth rate: $R = \lim_{n\to\infty} \frac{1}{n} \log m$
- Successful matching: $P_e \to 0$ as $n \to \infty$
- **Database matching $\Leftrightarrow$ Channel decoding**

This Talk:Database Matching Under Column Deletions

We assume

1. Databases do not have the same number of attributes
   - Random column deletion

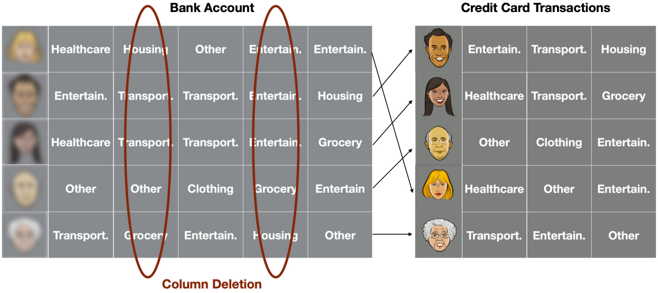
This Talk:Database Matching Under Column Deletions

We assume

1. Databases do not have the same number of attributes
   - Random column deletion
2. The indices of the deleted columns are not known.

We assume

1. Databases do not have the same number of attributes
   - Random column deletion
2. The indices of the deleted columns are not known.
3. Deletion pattern is constant across the rows.



This Talk:Database Matching Under Column Deletions

1. What are the sufficient conditions on the database growth rate for successful de-anonymization?

This Talk:Database Matching Under Column Deletions

1. What are the sufficient conditions on the database growth rate for successful de-anonymization?

2. How does side information on the deletion locations help?

This Talk:Database Matching Under Column Deletions

1. What are the sufficient conditions on the database growth rate for successful de-anonymization?
2. How does side information on the deletion locations help?
3. Can we extract this side information from an already-matched batch of rows, *i.e.* seeds?

This Talk:Database Matching Under Column Deletions

1. What are the sufficient conditions on the database growth rate for successful de-anonymization?

2. How does side information on the deletion locations help?

3. Can we extract this side information from an already-matched batch of rows, *i.e.* seeds?

4. How large this batch should be?

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	• •	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	• •	$X_{2,n-1}$	$X_{2,n}$
•	•	•	• •	•	•
•	•	•	• •	•	•
m	$X_{m,1}$	$X_{m,2}$	• •	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID	Attribute Vector			
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	• •	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	• •	$X_{\Theta^{-1}(2),n-1}$
•	•	•	• •	•
•	•	•	• •	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	• •	$X_{\Theta^{-1}(m),n-1}$

- $\mathcal{C}^{(1)} : (m, n, p_X)$ unlabeled database, *i.i.d.* entries $\sim p_X$ from $\mathfrak{X}$

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID — Attribute Vector

$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- $\mathcal{C}^{(1)} : (m, n, p_X)$ unlabeled database, *i.i.d.* entries $\sim p_X$ from $\mathfrak{X}$
- Columns deleted in $\mathcal{C}^{(2)}$ with probability δ (colored columns)

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID | Attribute Vector

$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- $\mathcal{C}^{(1)} : (m, n, p_X)$ unlabeled database, *i.i.d.* entries $\sim p_X$ from $\mathfrak{X}$
- Columns deleted in $\mathcal{C}^{(2)}$ with probability δ (colored columns)
- Θ: Labeling function

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID	Attribute Vector				
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- $\mathcal{C}^{(1)} : (m, n, p_X)$ unlabeled database, *i.i.d.* entries $\sim p_X$ from $\mathfrak{X}$
- Columns deleted in $\mathcal{C}^{(2)}$ with probability δ (colored columns)
- Θ: Labeling function
- $(\mathcal{C}^{(2)}, \Theta)$: Column deleted labeled database

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID	Attribute Vector				
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- $\mathcal{C}^{(1)} : (m, n, p_X)$ unlabeled database, *i.i.d.* entries $\sim p_X$ from $\mathfrak{X}$
- Columns deleted in $\mathcal{C}^{(2)}$ with probability δ (colored columns)
- Θ: Labeling function
- $(\mathcal{C}^{(2)}, \Theta)$: Column deleted labeled database
- Deleted columns detected with probability α (blue column)

Problem Formulation

$\mathcal{C}^{(1)}$

Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	• •	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	• •	$X_{2,n-1}$	$X_{2,n}$
•	•	•	• •	•	•
•	•	•	• •	•	•
m	$X_{m,1}$	$X_{m,2}$	• •	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID	Attribute Vector			
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	• •	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	• •	$X_{\Theta^{-1}(2),n-1}$
•	•	•	• •	•
•	•	•	• •	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	• •	$X_{\Theta^{-1}(m),n-1}$

- **Successful Matching Scheme**: A mapping $s : (\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) \to \hat{\Theta}$ satisfying

$$P(\Theta(I) = \hat{\Theta}(I)) \to 1 \text{ as } n \to \infty, \quad I \sim unif\{1, m\}$$

Problem Formulation

	$\mathcal{C}^{(1)}$ Attribute Vector					
1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)}, \Theta)$

User ID	Attribute Vector				
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- **Successful Matching Scheme**: A mapping $s : (\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) \to \hat{\Theta}$ satisfying
$$P(\Theta(I) = \hat{\Theta}(I)) \to 1 \text{ as } n \to \infty, \quad I \sim unif\{1, m\}$$
- **Database Growth Rate**: $R = \lim_{n\to\infty} \frac{1}{n} \log_2 m$
  - Relation between #users and #attributes
  - Large R → More users per attributes → More difficult to match

Problem Formulation

$\mathcal{C}^{(1)}$

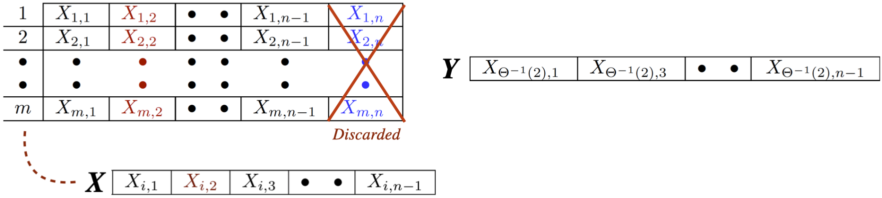
Attribute Vector

1	$X_{1,1}$	$X_{1,2}$	•	•	$X_{1,n-1}$	$X_{1,n}$
2	$X_{2,1}$	$X_{2,2}$	•	•	$X_{2,n-1}$	$X_{2,n}$
•	•	•	•	•	•	•
•	•	•	•	•	•	•
m	$X_{m,1}$	$X_{m,2}$	•	•	$X_{m,n-1}$	$X_{m,n}$

$(\mathcal{C}^{(2)},\Theta)$

User ID	Attribute Vector				
$\Theta^{-1}(1)$	$X_{\Theta^{-1}(1),1}$	$X_{\Theta^{-1}(1),3}$	•	•	$X_{\Theta^{-1}(1),n-1}$
$\Theta^{-1}(2)$	$X_{\Theta^{-1}(2),1}$	$X_{\Theta^{-1}(2),3}$	•	•	$X_{\Theta^{-1}(2),n-1}$
•	•	•	•	•	•
•	•	•	•	•	•
$\Theta^{-1}(m)$	$X_{\Theta^{-1}(m),1}$	$X_{\Theta^{-1}(m),3}$	•	•	$X_{\Theta^{-1}(m),n-1}$

- **Successful Matching Scheme**: A mapping $s:(\mathcal{C}^{(1)},\mathcal{C}^{(2)})\to\hat{\Theta}$ satisfying
$$P(\Theta(I)=\hat{\Theta}(I))\to 1 \text{ as } n\to\infty, \quad I\sim unif\{1,m\}$$
- **Database Growth Rate**: $R=\lim_{n\to\infty}\frac{1}{n}\log_2 m$
  - Relation between #users and #attributes
  - Large R → More users per attributes → More difficult to match
- **Achievable Database Growth Rate**: Given p_X, δ and α, R is achievable if for $(\mathcal{C}^{(1)},\mathcal{C}^{(2)})$, there exists a successful matching scheme.



1. Discard all the detected deleted columns in $\mathcal{C}^{(1)}$.
2. Match a row $\boldsymbol{Y}$ from $\mathcal{C}^{(2)}$ with a row $\boldsymbol{X}$ from $\mathcal{C}^{(1)}$ after discarding if
   - $\boldsymbol{X}$ is typical with respect to p_X.
   - $\boldsymbol{X}$ contains $\boldsymbol{Y}$ as a subsequence.
   - $\boldsymbol{X}$ is the only row of $\mathcal{C}^{(1)}$ satisfying the conditions above.

Achievable Database Growth Rate

Theorem

Given a column deletion probability $\delta < 1 - \frac{1}{|\mathfrak{X}|}$ and a deletion detection probability α, any database growth rate

$$R < \left[(1-\alpha\delta)\left(H(X) - H_b\left(\frac{1-\delta}{1-\alpha\delta}\right)\right) - (1-\alpha)\delta\log(|\mathfrak{X}|-1)\right]^+$$

is achievable, where H, H_b and $[.]^+$ denote the entropy, the binary entropy, and the positive part functions respectively.

Achievable Database Growth Rate

Theorem

Given a column deletion probability $\delta < 1 - \frac{1}{|\mathfrak{X}|}$ and a deletion detection probability α, any database growth rate

$$R < \left[(1-\alpha\delta)\left(H(X) - H_b\left(\frac{1-\delta}{1-\alpha\delta}\right)\right) - (1-\alpha)\delta\log(|\mathfrak{X}|-1)\right]^+$$

is achievable, where H, H_b and $[.]^+$ denote the entropy, the binary entropy, and the positive part functions respectively.

- Higher δ → Lower achievable rates

Achievable Database Growth Rate

Theorem

Given a column deletion probability $\delta < 1 - \frac{1}{|\mathfrak{X}|}$ and a deletion detection probability α, any database growth rate

$$R < \left[(1-\alpha\delta)\left(H(X) - H_b\left(\frac{1-\delta}{1-\alpha\delta}\right)\right) - (1-\alpha)\delta\log(|\mathfrak{X}|-1)\right]^+$$

is achievable, where H, H_b and $[.]^+$ denote the entropy, the binary entropy, and the positive part functions respectively.

- Higher δ → Lower achievable rates
- Higher α → Higher achievable rates

Achievable Database Growth Rate

Theorem

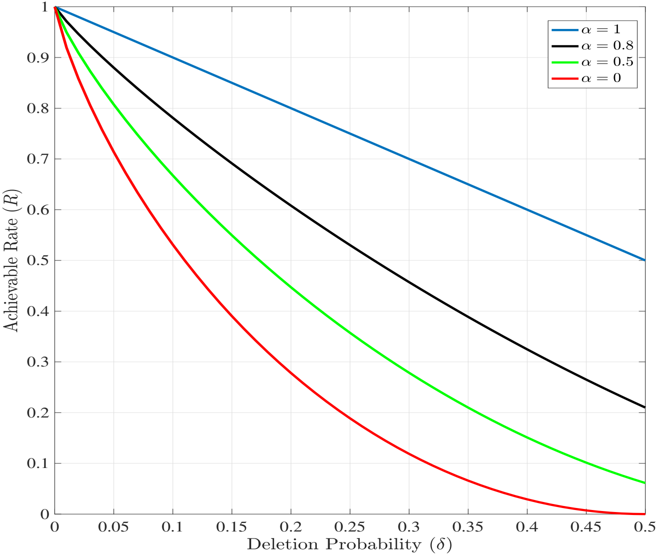
Given a column deletion probability $\delta < 1 - \frac{1}{|\mathfrak{X}|}$ and a deletion detection probability α, any database growth rate

$$R < \left[(1-\alpha\delta)\left(H(X) - H_b\left(\frac{1-\delta}{1-\alpha\delta}\right)\right) - (1-\alpha)\delta\log(|\mathfrak{X}|-1)\right]^+$$

is achievable, where H, H_b and $[.]^+$ denote the entropy, the binary entropy, and the positive part functions respectively.

- Higher δ → Lower achievable rates
- Higher α → Higher achievable rates
- Lower $H(X)$ → Lower achievable rates

Achievable Database Growth Rate



Achievable Rate vs. Deletion Probability, $X \sim Bernoulli(\frac{1}{2})$

Proof: Sketch

1. Bound #potential rows of $\mathcal{C}^{(1)}$ containing a given row $\boldsymbol{Y}$ of $\mathcal{C}^{(2)}$ after discarding detected deleted columns

Proof: Sketch

1. Bound #potential rows of $\mathcal{C}^{(1)}$ containing a given row $\boldsymbol{Y}$ of $\mathcal{C}^{(2)}$ after discarding detected deleted columns

2. Bound the probability of each such row of $\mathcal{C}^{(1)}$
   - Typicality

Proof: Sketch

1. Bound #potential rows of $\mathcal{C}^{(1)}$ containing a given row $\boldsymbol{Y}$ of $\mathcal{C}^{(2)}$ after discarding detected deleted columns
2. Bound the probability of each such row of $\mathcal{C}^{(1)}$
   - Typicality
3. 1 & 2 $\rightarrow$ Pairwise collision probability between 2 rows.

Proof: Sketch

1. Bound #potential rows of $\mathcal{C}^{(1)}$ containing a given row $\boldsymbol{Y}$ of $\mathcal{C}^{(2)}$ after discarding detected deleted columns
2. Bound the probability of each such row of $\mathcal{C}^{(1)}$
   - Typicality
3. 1 & 2 → Pairwise collision probability between 2 rows.
4. Union bound over $m = 2^{nR}$ rows

Observations

Corollary 1: No Deletion Detection

When $\alpha = 0$, we have

$$R < \left[H(X) - H_b(\delta) - \delta \log(|\mathfrak{X}| - 1)\right]^+$$

which is closely related to the deletion channel achievability result from [Diggavi and Grossglauser, 2006].

Corollary 2: Full Deletion Detection

When $\alpha = 1$, we have

$$R < (1 - \delta) H(X)$$

which is related to the erasure channel capacity.

Deletion Detection

- Exploiting known deletion locations helps!

Deletion Detection

- Exploiting known deletion locations helps!
- We've assumed deletion locations are given.

Deletion Detection

- Exploiting known deletion locations helps!
- We've assumed deletion locations are given.
- Instead, one might have access to a batch $(\mathcal{D}^{(1)}, \mathcal{D}^{(2)})$ of correctly-matched rows, *i.e.* seeds.

Deletion Detection

- Exploiting known deletion locations helps!
- We've assumed deletion locations are given.
- Instead, one might have access to a batch $(\mathcal{D}^{(1)}, \mathcal{D}^{(2)})$ of correctly-matched rows, *i.e.* seeds.
- Can we exploit this batch and the identicality of the column deletion pattern to detect the deleted columns?

Deletion Detection Function

$$\mathcal{D}^{(1)} = \begin{bmatrix} 0 & 1 & \mathbf{0} & 1 & 1 \\ 1 & 0 & \mathbf{0} & 1 & 1 \end{bmatrix} \qquad \mathcal{D}^{(2)}) = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

A simple deletion detection $g : \mathfrak{X}^{B\times n} \times \mathfrak{X}^{B\times K} \times [n] \to \{1, \text{inc}\}$ where

$$g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = \begin{cases} 1, & \boldsymbol{D}_j \text{ is not a column of } \mathcal{D}^{(2)} \text{and } \boldsymbol{D}_j \in A_\epsilon^{(B)} \\ \text{inc}, & \textit{otherwise} \end{cases}$$

- $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$: of sizes $B \times n$ and $B \times K$
- $A_\epsilon^{(B)}$: ϵ-typical set associated with p_X with parameter B
- $\boldsymbol{D}_j$: The j^{th} column of $\mathcal{D}^{(1)}$

For example, $g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, 3) = 1$

Performance of Deletion Detection Algorithm

Theorem

Let $\mathcal{D}^{(1)}, \mathcal{D}^{(2)}$ be a batch of correctly-matched B rows of the unlabeled database $\mathcal{C}^{(1)}$, and the corresponding column deleted database $\mathcal{C}^{(2)}$. Then

$$P(g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = 1 | j \in I_D) \geq 1 - \epsilon - n2^{-B(H(X)-\epsilon)}(1-\delta)$$

where I_D is the set of deleted column indices.

Performance of Deletion Detection Algorithm

Theorem

Let $\mathcal{D}^{(1)}, \mathcal{D}^{(2)}$ be a batch of correctly-matched B rows of the unlabeled database $\mathcal{C}^{(1)}$, and the corresponding column deleted database $\mathcal{C}^{(2)}$. Then

$$P(g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = 1 | j \in I_D) \geq 1 - \epsilon - n 2^{-B(H(X)-\epsilon)}(1-\delta)$$

where I_D is the set of deleted column indices.

- Higher B → Higher deletion detection probability

Performance of Deletion Detection Algorithm

Theorem

Let $\mathcal{D}^{(1)}, \mathcal{D}^{(2)}$ be a batch of correctly-matched B rows of the unlabeled database $\mathcal{C}^{(1)}$, and the corresponding column deleted database $\mathcal{C}^{(2)}$. Then

$$P(g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = 1 | j \in I_D) \geq 1 - \epsilon - n2^{-B(H(X)-\epsilon)}(1-\delta)$$

where I_D is the set of deleted column indices.

- Higher B → Higher deletion detection probability
- Lower $H(X)$ → Lower deletion detection probability

Observations

- To guarantee a non-zero deletion detection probability, we need a batch size $B = O(\log n) = O(\log \log m)$, where m is the number of users and n is the number of attributes.

Observations

- To guarantee a non-zero deletion detection probability, we need a batch size $B = O(\log n) = O(\log\log m)$, where m is the number of users and n is the number of attributes.

- $B = \omega(\log n) = \omega(\log\log m)$ guarantees that for large n, we have $P(g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = 1 | j \in I_D) \geq 1 - \epsilon$.

Observations

- To guarantee a non-zero deletion detection probability, we need a batch size $B = O(\log n) = O(\log \log m)$, where m is the number of users and n is the number of attributes.
- $B = \omega(\log n) = \omega(\log \log m)$ guarantees that for large n, we have $P(g(\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, j) = 1 | j \in I_D) \geq 1 - \epsilon$.
- **Remark:** Deletion detection from a batch of seeds does not necessarily lead to an *i.i.d.* deletion detection process.

Conclusion

- A matching scheme

Conclusion

- A matching scheme
- Sufficient conditions for database matching under random column deletions with probabilistic deletion detection.

Conclusion

- A matching scheme
- Sufficient conditions for database matching under random column deletions with probabilistic deletion detection.
- Deletion detection increases the achievable database growth rate
  - upto $\times 20$ when δ is large ($\delta \approx 0.4$).

Conclusion

- A matching scheme
- Sufficient conditions for database matching under random column deletions with probabilistic deletion detection.
- Deletion detection increases the achievable database growth rate
  - upto $\times 20$ when δ is large ($\delta \approx 0.4$).
- An algorithm to detect deleted columns from a batch of seeds.

Conclusion

- A matching scheme
- Sufficient conditions for database matching under random column deletions with probabilistic deletion detection.
- Deletion detection increases the achievable database growth rate
    - upto $\times 20$ when δ is large ($\delta \approx 0.4$).
- An algorithm to detect deleted columns from a batch of seeds.
- $\#$seeds = $O(\log\log \#\text{users})$ is enough to guarantee a non-zero deletion detection probability.

Conclusion

- A matching scheme
- Sufficient conditions for database matching under random column deletions with probabilistic deletion detection.
- Deletion detection increases the achievable database growth rate
    - upto $\times 20$ when δ is large ($\delta \approx 0.4$).
- An algorithm to detect deleted columns from a batch of seeds.
- $\#$seeds = $O(\log \log \#\text{users})$ is enough to guarantee a non-zero deletion detection probability.
- **Ongoing work**: Batchwise matching & Converse results