Medical Data Privacy and Ethics in the Age of Artificial Intelligence

Lecture 17: Federated Learning and Synthetic Data Generation

Zhiyu Wan, PhD (wanzhy@shanghaitech.edu.cn)

Assistant Professor of Biomedical Engineering

ShanghaiTech University

May 14, 2025

Due: on or before 3:00pm, May 28, 2025 (Wednesday)

HW2

- Question 1 (50 pts)

For this question, all datasets consist of Boolean attributes, where 0 and 1 both generalize to the value *. Assume that all attributes are equal in their modification (i.e., generalization/suppression) costs.

Q1a (10 points). Protect the following dataset using k-anonymization with $k = 2$. You can generalize cells and/or suppress records in your solution. Please make as few modifications as possible.

ID	Attribute A	Attribute B	Attribute C
1	1	0	1
2	0	0	0

Q1b (10 points). Protect the following dataset using k-anonymization with $k = 3$. You can generalize cells and/or suppress records in your solution. Please make as few modifications as possible.

ID	Attribute A	Attribute B	Attribute C
1	1	0	1
2	0	0	0

HW2

- Question 1 (50 pts)

For Q1c – Q1e, use the following dataset:

ID	Attribute A	Attribute B	Attribute C
1	1	1	1
2	1	1	1

Q1c (10 points). Protect the dataset using *k*-anonymization with *k* = 3. You can generalize cells and/or suppress records in your solution. Please make as few modifications as possible.

Q1d (10 points). Protect the dataset using *k*-anonymization with *k* = 3. You can generalize cells and/or suppress records in your solution. However, there is one additional condition: All cells in each attribute (column) need to be generalized to the same level if they are generalized. This is so-called **Full-domain generalization with suppression**. Please make as few modifications as possible.

Q1e (10 points). Protect the dataset using *k*-ambiguation with *k* = 3. You can generalize cells and/or suppress records in your solution. Please make as few modifications as possible.

Feel free to use the Datafly algorithm [1] or ARX Data Anonymization Tool [2,3] to help you achieve *k*-Anonymization. However, these tools are not required.

[1] Sweeney L. **Datafly: A system for providing anonymity in medical data**. *Database Security XI: Status and Prospects*. 1998:356-81. https://dataprivacylab.org/datafly/paper2.pdf

[2] Prasser F, Kohlmayer F, Lautenschläger R, Kuhn KA. **Arx-a comprehensive tool for anonymizing biomedical data**. In *AMIA Annual Symposium Proceedings* 2014 Nov 14 (Vol. 2014, p. 984). https://pmc.ncbi.nlm.nih.gov/articles/PMC4419984/pdf/1984395.pdf

[3] **ARX Data Anonymization Tool.** https://arx.deidentifier.org/

HW2

- Question 2 (50 pts)

Problem Description:

In this problem, you are asked to decode a string that was encoded using a variation of the Caesar cipher.

Example:

```
>>> decode_caesar_with_key("EQJ5586 ltpibrvp!", [3, 4, 5])
' BME2133 homework!'
```

HW2

- Question 2 (50 pts)

Submit your source code in *.cpp file(s) or *.py / *.ipynb file(s).

Datafly (Sweeney ‘97 & ‘02)

- Input:

Table T

Quasi-ID = $\{A_1, \ldots, A_p\}$

k protection parameter

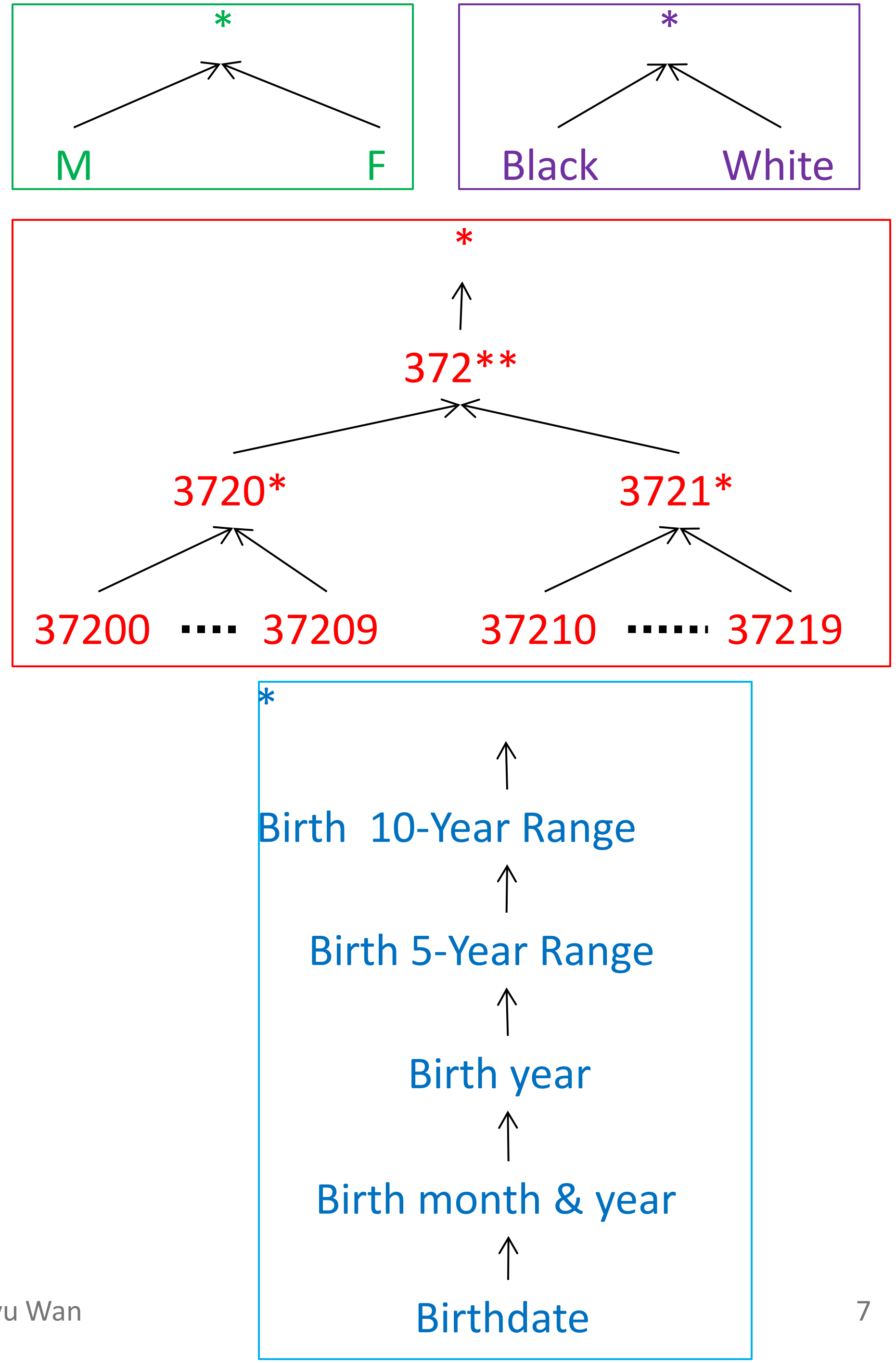
Domain Generalization Hierarchies DGH_{Ai}

L. Sweeney. Guaranteeing anonymity when sharing medical data, the Datafly system. Proc AMIA Symp. 1997: 51-55.

L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness, & Knowledge-based Systems. 2002; 10(5): 571-588.

Example k = 2

Record	Race	Birthdate	Sex	Zip
r_1	Black	9/20/65	M	37203
r_2	Black	2/14/65	M	37203
r_3	Black	10/23/65	F	37215
r_4	Black	8/24/65	F	37215
r_5	Black	11/7/65	F	37215
r_6	Black	12/1/64	F	37215
r_7	White	10/23/64	M	37215
r_8	White	3/15/64	F	37217
r_9	White	8/13/64	M	37217
r_{10}	White	5/5/64	M	37217
r_{11}	White	2/13/67	M	37215
r_{12}	White	3/21/67	M	37215



Adapted from Brad Malin's slides

Datafly (Sweeney)

1. FREQ ← list of quasi-id value frequencies from table
2. While the set of quasi-ids in FREQ with count < k account for > k records
    1. Choose attribute A_i with greatest number of distinct values in FREQ
    2. Generalize all quasi-ids according to the DGH_{Ai}
3. Suppress quasi-ids from FREQ with < k records
4. If 0 < (# of suppressed records) < k
    1. Suppress k – (# of suppressed records) records
5. Return protected table ← built from FREQ

1. FREQ ← list of quasi-id value frequencies

Race	Birthdate	Sex	Zip
Black	9/20/65	M	37203
Black	2/14/65	M	37203
Black	10/23/65	F	37215
Black	8/24/65	F	37215
Black	11/7/64	F	37215
Black	12/1/64	F	37215
White	10/23/64	M	37215
White	3/15/64	F	37217
White	8/13/64	M	37217
White	5/5/64	M	37217
White	2/13/67	M	37215
White	3/21/67	M	37215

1. FREQ ← list of quasi-id value frequencies

Race	Birthdate	Sex	Zip
Black	9/20/65	M	37203
Black	2/14/65	M	37203
Black	10/23/65	F	37215
Black	8/24/65	F	37215
Black	11/7/64	F	37215
Black	12/1/64	F	37215
White	10/23/64	M	37215
White	3/15/64	F	37217
White	8/13/64	M	37217
White	5/5/64	M	37217
White	2/13/67	M	37215
White	3/21/67	M	37215

Count	Records
1	r_1
1	r_2
1	r_3
1	r_4
1	r_5
1	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}

Race	Birthdate	Sex	Zip
Black	9/20/65	M	37203
Black	2/14/65	M	37203
White	10/23/64	M	37215
White	3/15/64	F	37217
White	8/13/64	M	37217
White	5/5/64	M	37217
White	2/13/67	M	37215
White	3/21/67	M	37215

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{Ai}

Count	Records
1	r_1
1	r_2
	r_3
	r_4
	r_5
	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}

2. While the set of quasi-ids in FREQ with count < k account for > k records
    1. Choose attribute A_i with greatest number of distinct values in FREQ
    2. Generalize all quasi-ids according to DGH_{Ai}

Race	Birthdate	Sex	Zip
Black	9/20/65	M	37203
Black	2/14/65	M	37203
White	10/23/64	M	37215
White	3/15/64	F	37217
White	8/13/64	M	37217
White	5/5/64	M	37217
White	2/13/67	M	37215
White	3/21/67	M	37215

Count	Records
1	r_1
1	r_2
	r_3
	r_4
	r_5
	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}
12	

Race	Birthdate	Sex	Zip
Black	9/20/65	M	37203
Black	2/14/65	M	37203
White	10/23/64	M	37215
White	3/15/64	F	37217
White	8/13/64	M	37217
White	5/5/64	M	37217
White	2/13/67	M	37215
White	3/21/67	M	37215

COUNT	Records
1	r_1
1	r_2
	r_3
	r_4
	r_5
	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{A_i}

	Race	Birthdate	Sex	Zip
	Black	9/20/65	M	37203
	Black	2/14/65	M	37203
	Black	10/23/65	F	37215
	Black	8/24/65	F	37215
	Black	11/7/64	F	37215
	Black	12/1/64	F	37215
	White	10/23/64	M	37215
	White	3/15/64	F	37217
	White	8/13/64	M	37217
	White	5/5/64	M	37217
	White	2/13/67	M	37215
	White	3/21/67	M	37215
# of Values	**2**	**12**	**2**	**3**

Greatest Number of Values

COUNT	Records
1	r_1
1	r_2
1	r_3
1	r_4
1	r_5
1	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}

	Race	Birthdate	Sex	Zip
	Black	9/20/65	M	37203
	Black	2/14/65	M	37203
	White	10/23/64	M	37215
	White	3/15/64	F	37217
	White	8/13/64	M	37217
	White	5/5/64	M	37217
	White	2/13/67	M	37215
	White	3/21/67	M	37215
# of Values	**2**	**12**	**2**	**3**

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{A_i}

Greatest Number of Values

*

↑

Birth 10-year Range

↑

Birth 5-year Range

↑

Birth year

↑

Birth month & year

↑

Birthdate

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{A_i}

	Race	Birthdate	Sex	Zip
	Black	9/65	M	37203
	Black	2/65	M	37203
	White	10/64	M	37215
	White	3/64	F	37217
	White	8/64	M	37217
	White	5/64	M	37217
	White	2/67	M	37215
	White	3/67	M	37215
# of Values	**2**	**12**	**2**	**3**

FREQ	Records
1	r_1
1	r_2
1	r_3
1	r_4
1	r_5
1	r_6
1	r_7
1	r_8
1	r_9
1	r_{10}
1	r_{11}
1	r_{12}
12	

	Race	Birthdate	Sex	Zip
	Black	9/65	M	37203
	Black	2/65	M	37203
	White	10/64	M	37215
	White	3/64	F	37217
	White	8/64	M	37217
	White	5/64	M	37217
	White	2/67	M	37215
	White	3/67	M	37215
# of Values	**2**	**12**	**2**	**3**

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{A_i}

Race	Birthdate	Sex	Zip
Black	9/65	M	37203
Black	2/65	M	37203
White	10/64	M	37215
White	3/64	F	37217
White	8/64	M	37217
White	5/64	M	37217
White	2/67	M	37215
White	3/67	M	37215
# of Values	**2**	**12**	**2**	**3**

2. While the set of quasi-ids in FREQ with count < k account for > k records
    1. Choose attribute A_i with greatest number of distinct values in FREQ
    2. Generalize all quasi-ids according to DGH_{A_i}

*

↑

Birth 10-year Range

↑

Birth 5-year Range

↑

Birth year

↑

Birth month & year

↑

Birthdate

Greatest Number of Values

	Race	Birthdate	Sex	Zip
	Black	1965	M	37203
	Black	1965	F	37215
	Black	1964	F	37215
	White	1964	M	37215
	White	1964	F	37217
	White	1964	M	37217
	White	1967	M	37215
# of Values	**2**	**3**	**2**	**3**

Count	Records
2	r_1, r_2
2	r_3, r_4
2	r_5, r_6
1	**r_7**
1	**r_8**
2	r_9, r_{10}
2	r_{11}, r_{12}
2	**←TOTAL**

2. While the set of quasi-ids in FREQ with count < k account for > k records
   1. Choose attribute A_i with greatest number of distinct values in FREQ
   2. Generalize all quasi-ids according to DGH_{A_i}

	Race	Birthdate	Sex	Zip
	Black	1965	M	37203
	Black	1965	F	37215
	Black	1964	F	37215
	White	1964	M	37215
	White	1964	F	37217
	White	1964	M	37217
	White	1967	M	37215
# of Values	**2**	**3**	**2**	**3**

Count	Records
2	r_1, r_2
2	r_3, r_4
2	r_5, r_6
1	r_7
1	r_8
2	
2	
2	

Only 2 records

2. While the set of quasi-ids in FREQ with count < k account for > k records
    1. Choose attribute A_i with greatest number of distinct values in FREQ
    2. Generalize all quasi-ids according to DGH_{A_i}

	Race	Birthdate	Sex	Zip
	Black	1965	M	37203
	Black	1965	F	37215
	Black	1964	F	37215
	White	1964	M	37215
	White	1964	F	37217
	White	1964	M	37217
	White	1967	M	37215
# of Values	**2**	**3**	**2**	**3**

Count	Records
2	r_1, r_2
2	r_3, r_4
2	r_5, r_6
1	r_7
1	r_8
2	r_9, r_{10}
2	r_{11}, r_{12}
2	

3. Remove quasi-ids from FREQ with < k records

	Race	Birthdate	Sex	Zip
	Black	1965	M	37203
	Black	1965	F	37215
	Black	1964	F	37215
	White	1964	M	37217
	White	1967	M	37215
# of Values	**2**	**3**	**2**	**3**

Count	Records
2	r_1, r_2
2	r_3, r_4
2	r_5, r_6
2	r_9, r_{10}
2	r_{11}, r_{12}

3. Remove quasi-ids from FREQ with < k records

	Race	Birthdate	Sex	Zip
	Black	1965	M	37203
	Black	1965	F	37215
	Black	1964	F	37215
	White	1964	M	37217
	White	1967	M	37215
# of Values	**2**	**3**	**2**	**3**

Count	Records
2	r_1, r_2
2	r_3, r_4
2	r_5, r_6
2	r_9, r_{10}
2	r_{11}, r_{12}
2	**# Suppressed**

Suppressed = k = 2 records ← done

4. If 0 < # of suppressed records < k
   1. Suppress k – (# of suppressed records) records

Race	Birthdate	Sex	Zip
Black	1965	M	37203
Black	1965	M	37203
Black	1965	F	37215
Black	1965	F	37215
Black	1964	F	37215
Black	1964	F	37215
White	1964	M	37217
White	1964	M	37217
White	1967	M	37215
White	1967	M	37215

5. Return T ← built from FREQ

FREQ				
Race	Birthdate	Sex	Zip	Count
Black	1965	M	37203	2
Black	1965	F	37215	2
Black	1964	F	37215	2
White	1964	M	37217	2
White	1967	M	37215	2

Record	Race	Birthdate	Sex	Zip
r_1	Black	9/20/65	M	37203
r_2	Black	2/14/65	M	37203
r_3	Black	10/23/65	F	37215
r_4	Black	8/24/65	F	37215
r_5	Black	11/7/65	F	37215
r_6	Black	12/1/64	F	37215
r_7	White	10/23/64	M	37215
r_8	White	3/15/64	F	37217
r_9	White	8/13/64	M	37217
r_{10}	White	5/5/64	M	37217
r_{11}	White	2/13/67	M	37215
r_{12}	White	3/21/67	M	37215

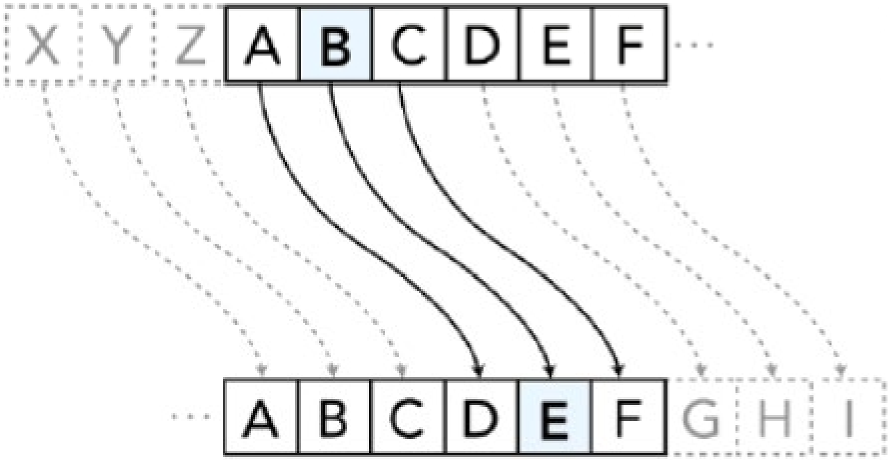
Original

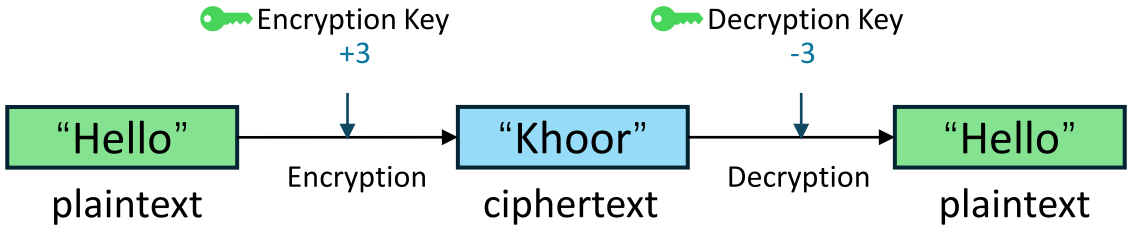
Race	Birthdate	Sex	Zip
Black	1965	M	37203
Black	1965	M	37203
Black	1965	F	37215
Black	1965	F	37215
Black	1964	F	37215
Black	1964	F	37215
*	*	*	*
*	*	*	*
White	1964	M	37217
White	1964	M	37217
White	1967	M	37215
White	1967	M	37215

2-anonymous

Caesar Cipher

Shift=3





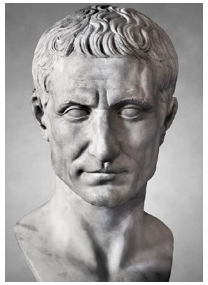
Ciphertext=(Plaintext + Shift) mod 26

Plaintext =(Ciphertext - Shift) mod 26

Credit: Yuhang Guo

Caesar Cipher

N fr f ljsnzx!



已深度思考（用时 27 秒）

好的，我现在需要帮用户解码这个凯撒密码的密文："N fr f ljsnzx!"。首先，用户可能已经知道凯撒密码的基本原理，但可能需要具体的步骤来解码。让我先回顾一下凯撒密码的工作方式。凯撒密码通过将字母表中的每个字母移动固定的偏移量来加密，解码时则需要反向移动相同的偏移量。

首先，我需要确定用户提供的密文是否有任何明显的特征可以帮助确定偏移量。比如，常见的英文单词或结构。密文是"N fr f ljsnzx!"。看起来有几个单字母单词，比如"f"，这可能对应英文中的"I"或"a"，但需要进一步分析。

接下来，考虑到用户可能不知道偏移量，我需要尝试不同的偏移量进行暴力破解。凯撒密码通常有25种可能的偏移量（1到25），因为偏移0或26不会改变原文。用户可能希望快速得到结果，所以我可以列出所有可能的解码结果，或者找到最可能的偏移量。

让我先尝试一些常见的偏移量，比如凯撒常用的3，或者ROT13（偏移13）。不过先看看密文中的单字母词"f"。在英语中，单字母词通常是"I"或"a"。假设"f"对应"a"，那么偏移量是5（因为a到f是+5）。如果"f"对应"I"，则偏移量是-5（即+21）。我需要检查这两种可能性。

首先尝试偏移量5的解码：

- N -> I（N是第14个字母，14-5=9，对应I）

- f -> a（f是第6个字母，6-5=1，对应a）

- r -> m（r是18-5=13，对应M）

所以前两个词"N fr"解码为"I am"，这看起来合理。接着继续：

- l -> g（l是12-5=7，G）

- j -> e（j是10-5=5，E）

- s -> n（s是19-5=14，N）

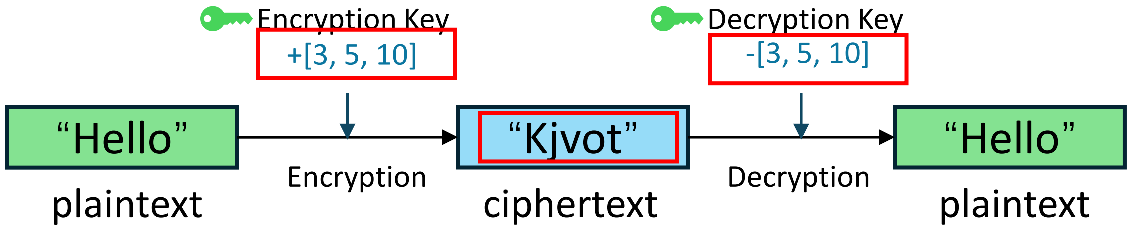
- n -> i（n是14-5=9，I）

- z -> u（z是26-5=21，U）

- x -> s（x是24-5=19，S）

所以"ljsnzx"解码为"genius"，整个句子就是"I am a genius!"，这完全合理。

不过，我需要验证其他可能的偏移量是否也能得到有意义的结果，但根据经验，偏移量5在这里显然正确。用户可能希望得到明确的答案，而不仅仅是可能的列表，所以直接给出偏移量5的解码结果即可。同时，用户可能需要知道如何验证这个过程，所以解释每个字母的转换步骤也是有必要的。



Ciphertext=(Plaintext + Shift) mod 26

Plaintext =(Ciphertext - Shift) mod 26

Credit: Yuhang Guo

Learning Objectives of This Lecture

After this lecture, students should be able to:

- Know the concept of federated learning (FL)
  - Types of FL
  - FedAvg
- Know the concept of Synthetic Data Generation (SDG)
  - GAN

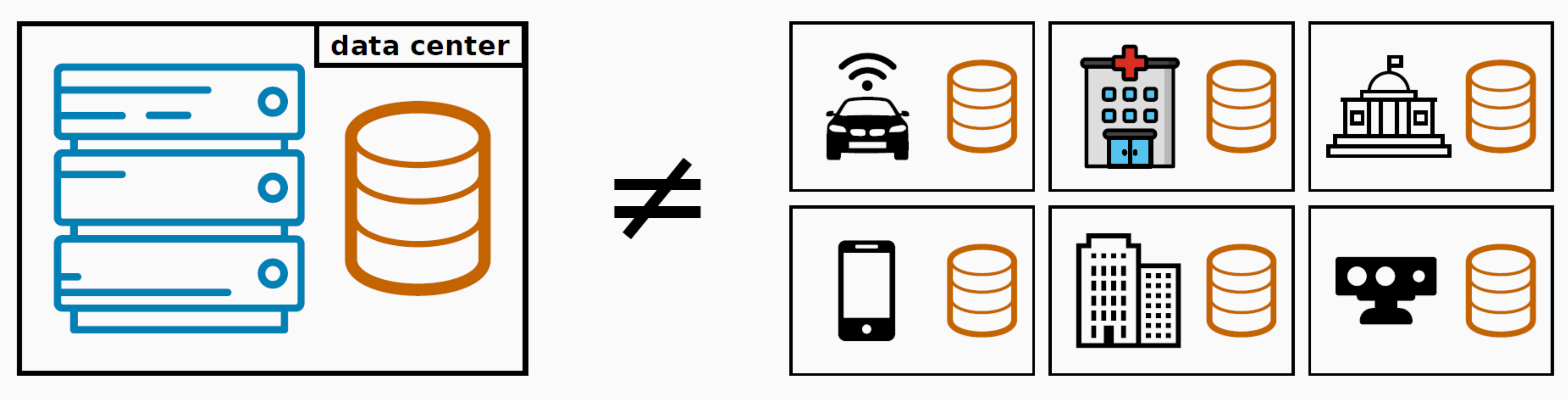
Outline

- What is Federated Learning (FL)
  - Types of FL
  - A baseline algorithm: FedAvg
  - Challenges of FL
- What is Synthetic Data Generation
  - GAN

Federated Learning

From Centralized to Decentralized Data

- The standard setting in Machine Learning considers a **centralized dataset processed in a tightly integrated system**
- But in the real world, **data is often decentralized across many parties**



Adapted from Aurelien Bellet's slides

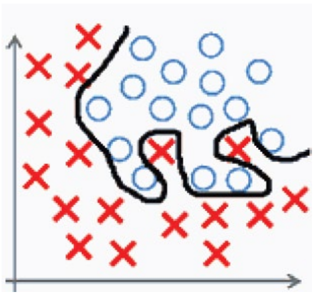
Why can not we just centralize the data?

- Sending the data may be **too costly**
  - Self-driving cars are expected to generate several TBs of data a day
  - Some wireless devices have limited bandwidth/power

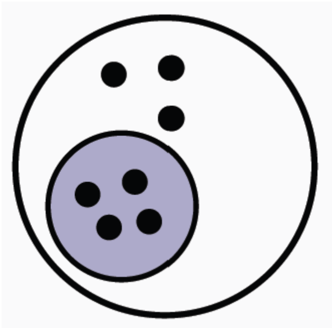
- Data may be considered **too sensitive**
  - We see a growing public awareness and regulations on data privacy
  - Keeping control of data can give a competitive advantage in business and research

How about each party learning on its own?

- The local dataset may be **too small**
  - Poor predictive performance (e.g., due to overfitting)
  - Non-statistically significant results (e.g., medical studies)

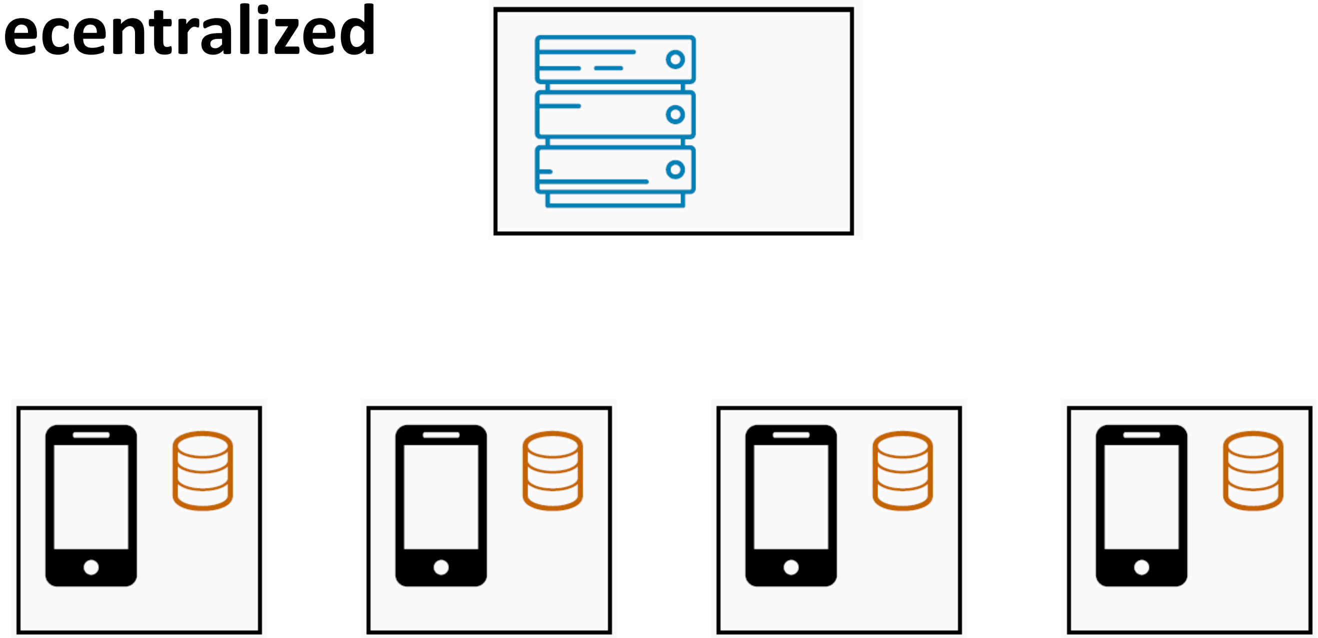


- The local dataset may be **biased**
  - Not representative of the target distribution



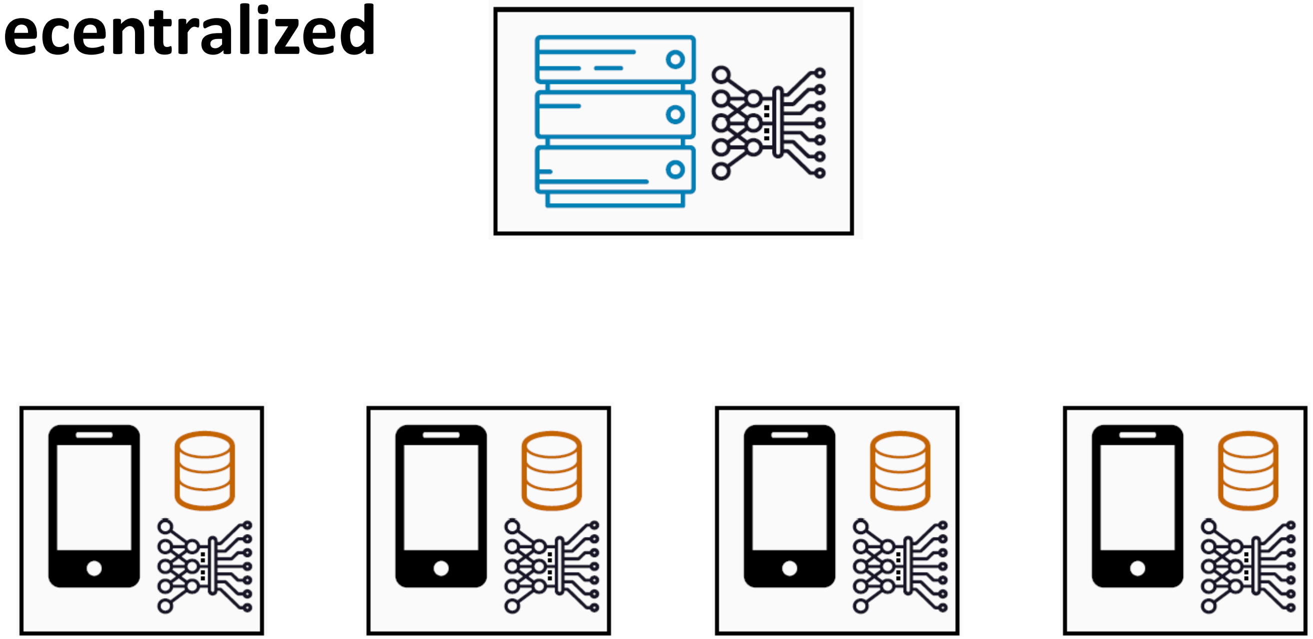
Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**



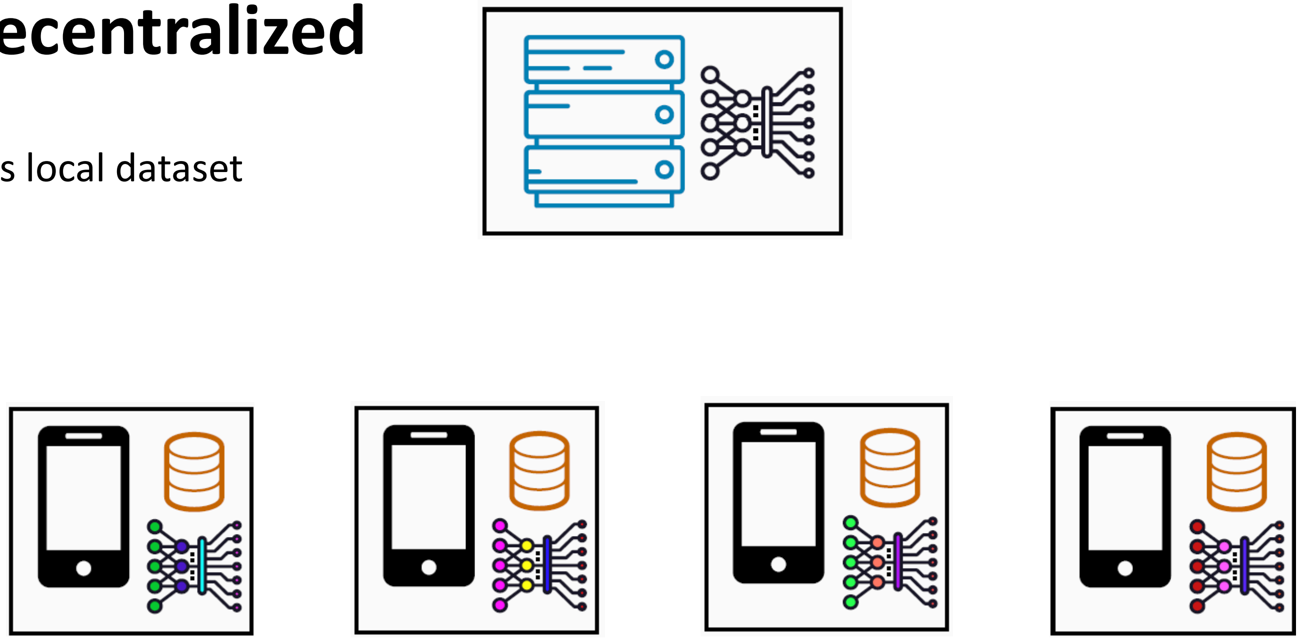
Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**
    1. Initialize model



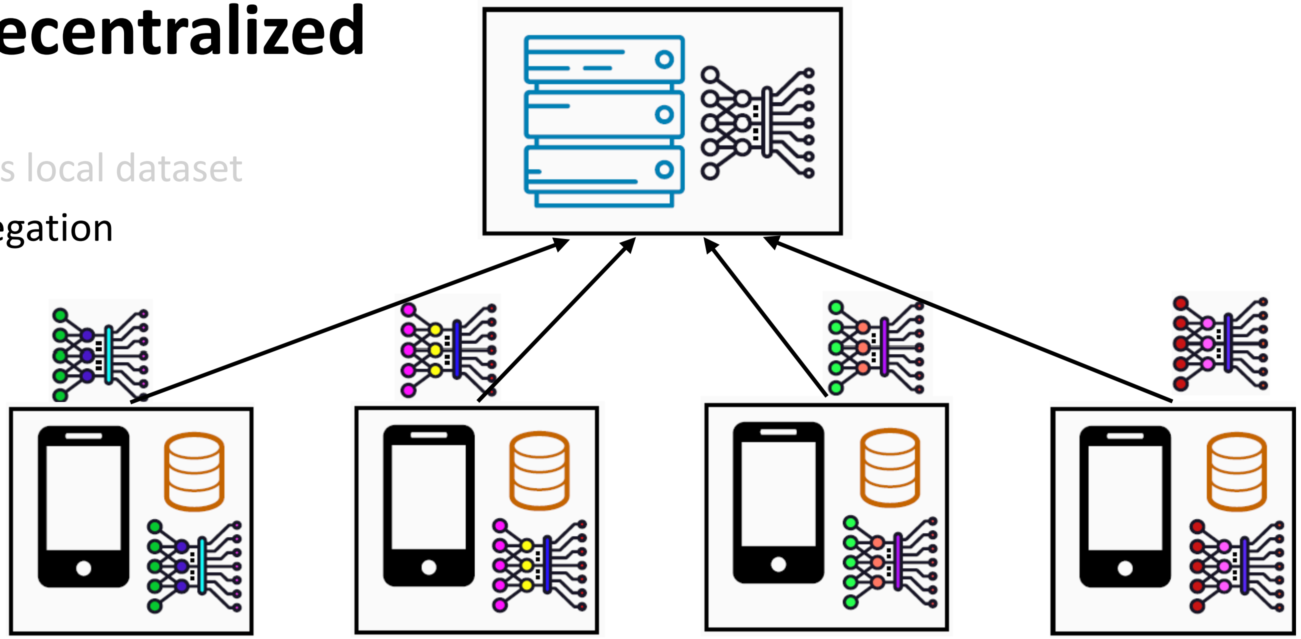
Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**
  1. Initialize model
  2. Each party makes an update using its local dataset



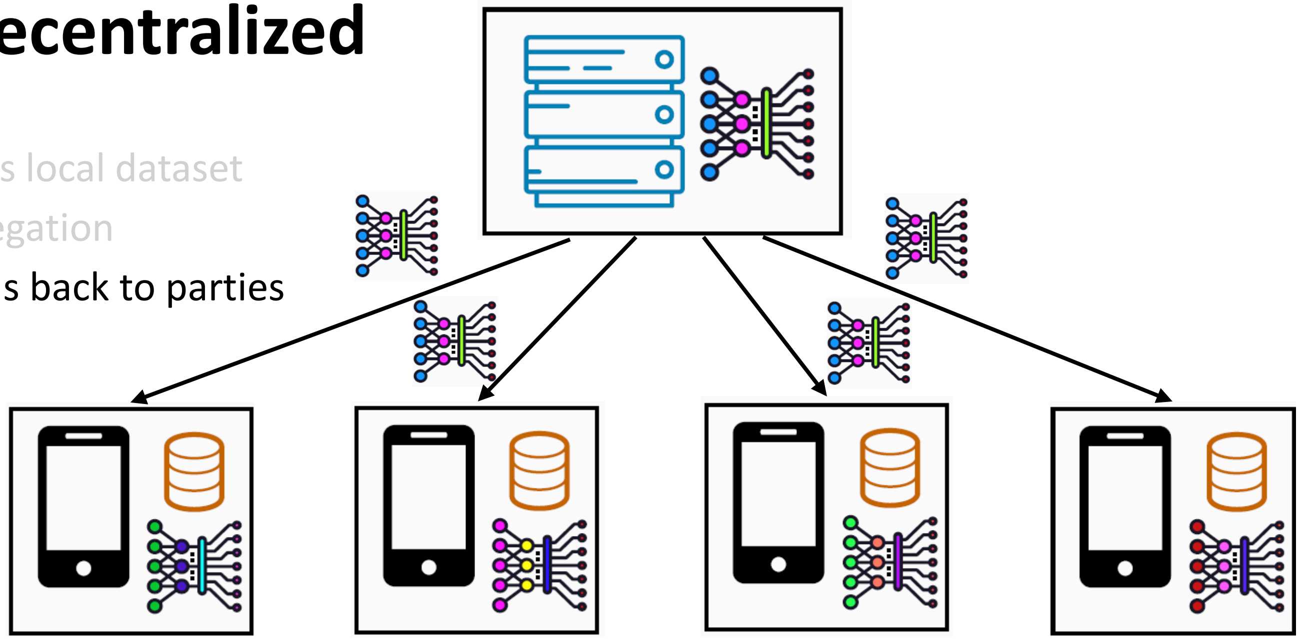
Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**
    1. Initialize model
    2. Each party makes an update using its local dataset
    3. Parties share local updates for aggregation



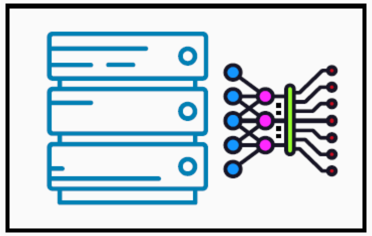
Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**
    1. Initialize model
    2. Each party makes an update using its local dataset
    3. Parties share local updates for aggregation
    4. Server aggregates updates and sends back to parties

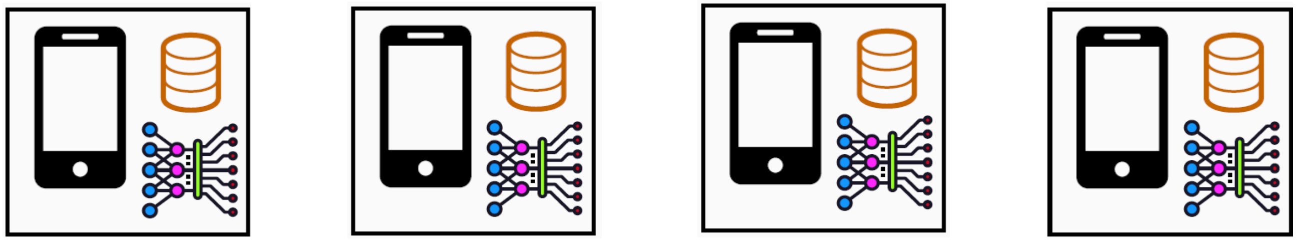


Broad Definition of Federated Learning

- Federated Learning (FL) aims to **collaboratively train a ML model** while **keeping the data decentralized**



  1. Initialize model
  2. Each party makes an update using its local dataset
  3. Parties share local updates for aggregation
  4. Server aggregates updates and sends back to parties
  5. Parties update their model and iterate



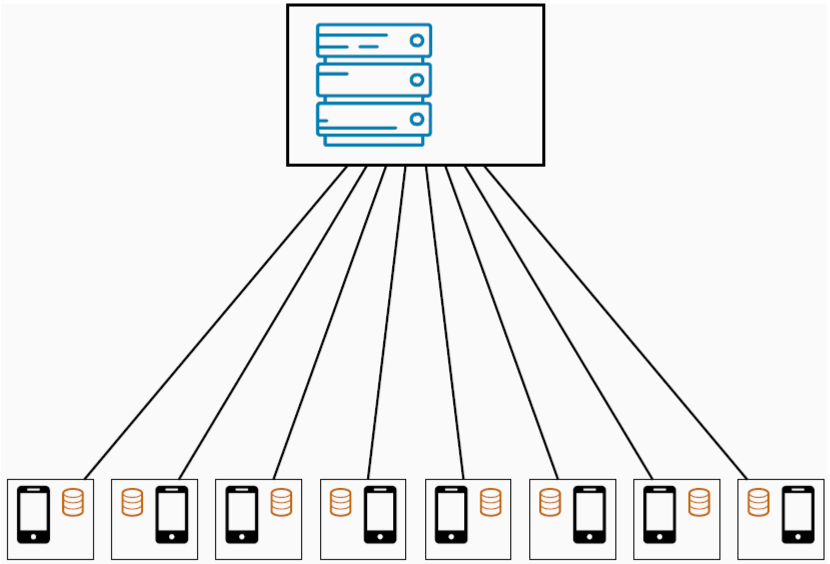
We would like the final model to be **as good as the centralized solution** (ideally), or at least **better than what each party can learn on its own**

Key Differences with Distributed Learning

- Data distribution
  - In distributed learning, **data is centrally stored** (e.g., in a data center)
    - The main goal is just to **train faster**
    - We control how data is distributed across workers: usually, it is **distributed uniformly at random** across worker
  - In FL, **data is naturally distributed and generated locally**
    - Data is not independent and identically distributed (**non-i.i.d.**), and it is **imbalanced**
- Additional challenges that arise in FL
  - Enforcing privacy constraints
  - Dealing with the possibly limited reliability/ availability of participants
  - Achieving robustness against malicious parties

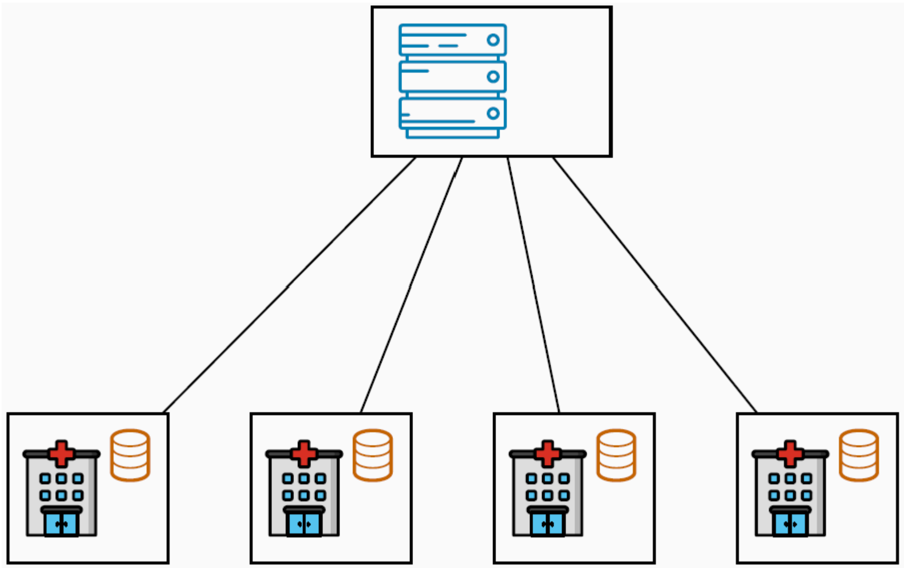
Cross-Device VS. Cross-Silo FL

- Cross-device FL



- Massive number of parties (up to 10^{10})
- Small dataset per party (could be 1)
- Limited availability and reliability
- Some parties may be malicious

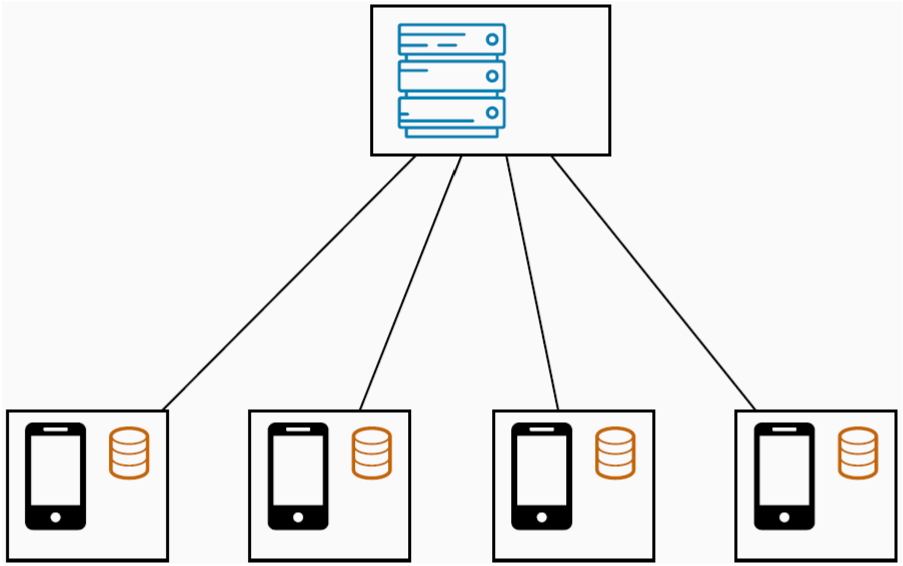
- Cross-silo FL



- 2-100 parties
- Medium to large dataset per party
- Reliable parties, almost always available
- Parties are typically honest

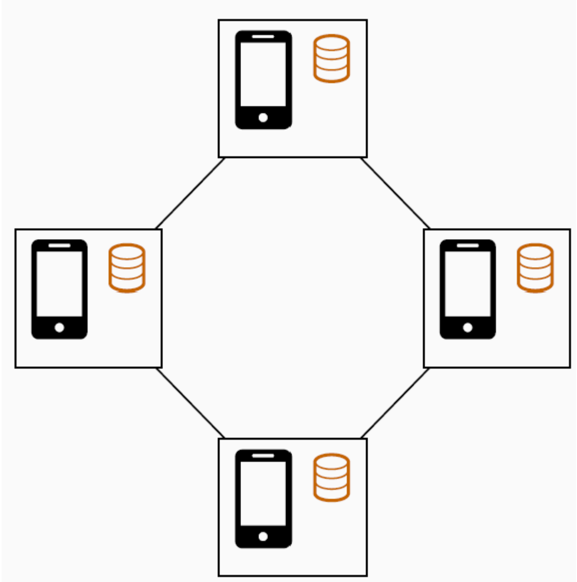
Server Orchestrated VS. Fully Decentralized FL

- Server-orchestrated FL



  - Server-client communication
  - Global coordination, global aggregation
  - Server is a single point of failure and may become a bottleneck

- Fully decentralized FL



  - Device-to-device communication
  - No global coordination, local aggregation
  - Naturally scales to a large number of devices

Categorized based on feature overlap in client datasets

- Horizontal Federated Learning (HFL)
- Vertical Federated Learning (VFL)
- Federated Transfer Learning (FTL)

Horizontal Federated Learning

- **Shared features, different users**: Clients have the same set of features.
- **Focus**: Leveraging the diversity of users with the same data structure to enhance model accuracy and generalization.
- **Example**: Multiple banks training a fraud detection model using transaction data (shared features) from different customers (different users).

Vertical Federated Learning

- **Different features, overlapping users**: Clients have different feature sets but some features might overlap.
- **Focus**: Combining data from participants with complementary information while protecting sensitive features.
- **Example**: Hospitals and insurance companies collaborating on healthcare predictions using medical records (Hospital data) and policy data (Insurance data) with overlapping features like patient IDs.

Federated Transfer Learning

- **Leveraging pre-trained knowledge**: Uses a pre-trained model to guide learning on a new task or data with different characteristics.
- **Focus**: Accelerating learning on new tasks or data with limited resources, especially when privacy concerns restrict model sharing.
- **Example**: Using a sentiment analysis model trained on public product reviews to personalize recommendations within a specific e-commerce domain.

Key Differences

Features	Horizontal FL	Vertical FL	Transfer FL
Feature overlap	High	Low/Partial	No
User overlap	Low	High	Varies
Focus	Data diversity, accuracy	Shared information, privacy	Knowledge transfer

Synchronous Federated Learning

- The Server updates the shared central model after “all the devices send their model updates”.
- Eg: Federated Averaging.
- This approach offers several advantages:
  - **Faster convergence**: Synchronization leads to quicker convergence towards a more accurate global model.
  - **Better accuracy**: The coordinated updates can result in higher model accuracy compared to asynchronous methods.
  - **Reduced staleness**: Updates are always fresh, mitigating the issue of outdated gradients.

Synchronous Federated Learning

- However, synchronous federated learning also faces some challenges:
  - **Increased communication overhead**: All devices need to communicate with the server at every step, leading to higher bandwidth requirements.
  - **Higher synchronization latency**: Waiting for the slowest device can introduce delays in the training process.

Asynchronous Federated Learning

- The Server updates the shared central model "as the new updates keep coming in".
- Eg: SMPC Aggregation, Secure Aggregation with Trusted Execution Environment(TEE).
- This approach offers several advantages:
  - **Relaxed communication requirements**: Devices can update the model whenever convenient, reducing communication overhead.
  - **Improved scalability**: Asynchronous learning can handle a large number of devices more efficiently.
  - **Fault tolerance**: The system is more resilient to device failures or intermittent connections

Asynchronous Federated Learning

- However, asynchronous federated learning also faces some challenges:
  - **Stale gradients**: Updates from devices may become outdated before reaching the server, impacting accuracy.
  - **Slower convergence**: The lack of synchronization can slow down the overall training process.
  - **Potential for divergence**: Individual models on devices may diverge significantly from the global model.

History of Federated Learning

- 2016: the term FL is first coined by Google researchers;
- 2020: more than **1,000 papers on FL in the first half of the year** (compared to just 180 in 2018)
- We have already seen some **real-world deployments** by companies and researchers
- Several **open-source libraries** are under development: PySyft, TensorFlow Federated, FATE, Flower, Substra...
- FL is highly **multidisciplinary**: it involves machine learning, numerical optimization, privacy & security, networks, systems, hardware...

https://www.forbes.com/sites/robtoews/2020/10/12/the-next-generation-of-artificial-intelligence/

What Is Aggregation in FL?

- Aggregation methods vary, each with unique advantages and challenges.
  - Beyond model updates, aggregate statistical indicators (loss, accuracy).
  - Hierarchical aggregation for large-scale FL systems.
- Aggregation algorithms are crucial for FL success.
  - Determine model training effectiveness.
  - Impact practical usability of the global model.

Different Approaches of Aggregation

- 2017-2019
  - FedAvg, RFA, 1 unnamed
- 2020
  - FedProx, LAQ, SAFA, FedBoost, SACFFOLD FedMA, 3unnamed
- 2021
  - FedDist, FEDHQ, FAIR, FedPSO, SecureD-FL, LEGATO, SEAR, MHAT
- After 2022
  - EPPDA, FedBuff, HeteroSAg, LightSecAgg

Different Approaches of Aggregation

- Average Aggregation
- Clipped Average Aggregation
- Secure Aggregation
- Differential Privacy Average Aggregation
- Momentum Aggregation
- Weighted Aggregation
- Bayesian Aggregation
- Adversarial Aggregation
- Quantization
- Hierarchical Aggregation
- Personalized Aggregation
- Ensemble-based Aggregation

Model Aggregation Techniques

- Federated Averaging:
  - Each device sends its model updates.
  - The updates are averaged to create a better global model.
- Federated Stochastic Gradient Descent (FedSGD):
  - Devices send gradients (directions to improve the model).
  - The global model adjusts based on these gradients.

A Baseline Algorithm: FedAvg

- We consider a set of $\boldsymbol{K}$ **parties (clients)**
- Each party k holds a **dataset $\boldsymbol{D_k}$ of $\boldsymbol{n_k}$ points**
- Let $D = D_1 \cup \ldots \cup D_k$ be the joint dataset and $n = \sum_k n_k$ the total number of points
- We want to solve problems of the form $\min\limits_{\boldsymbol{\theta} \in \mathbb{R}^{\boldsymbol{p}}} \boldsymbol{F}(\boldsymbol{\theta}; \boldsymbol{D})$ where:
- $\boldsymbol{F}(\boldsymbol{\theta}; \boldsymbol{D}) = \sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{k}} \frac{\boldsymbol{n_k}}{\boldsymbol{n}} \boldsymbol{F_k}(\boldsymbol{\theta}; \boldsymbol{D_k})$ and $\boldsymbol{F_k}(\boldsymbol{\theta}; \boldsymbol{D_k}) = \sum_{\boldsymbol{d} \in \boldsymbol{D_k}} \boldsymbol{f}(\boldsymbol{\theta}; \boldsymbol{d})$
- $\boldsymbol{\theta} \in \mathbb{R}^{\boldsymbol{p}}$ **are model parameters** (e.g., weights of a logistic regression or neural network)
- This **covers a broad class of ML problems** formulated as **empirical risk minimization**.

FedAvg (a.k.a. Local SGD)

Algorithm FedAvg (server-side)

Parameters: client sampling rate ρ

initialize θ
for each round $t = 0, 1, \ldots$ **do**
  $\mathcal{S}_t \leftarrow$ random set of $m = \lceil \rho K \rceil$ clients
  for each client $k \in \mathcal{S}_t$ in parallel **do**
    $\theta_k \leftarrow \text{ClientUpdate}(k, \theta)$
  $\theta \leftarrow \sum_{k \in \mathcal{S}_t} \frac{n_k}{n} \theta_k$

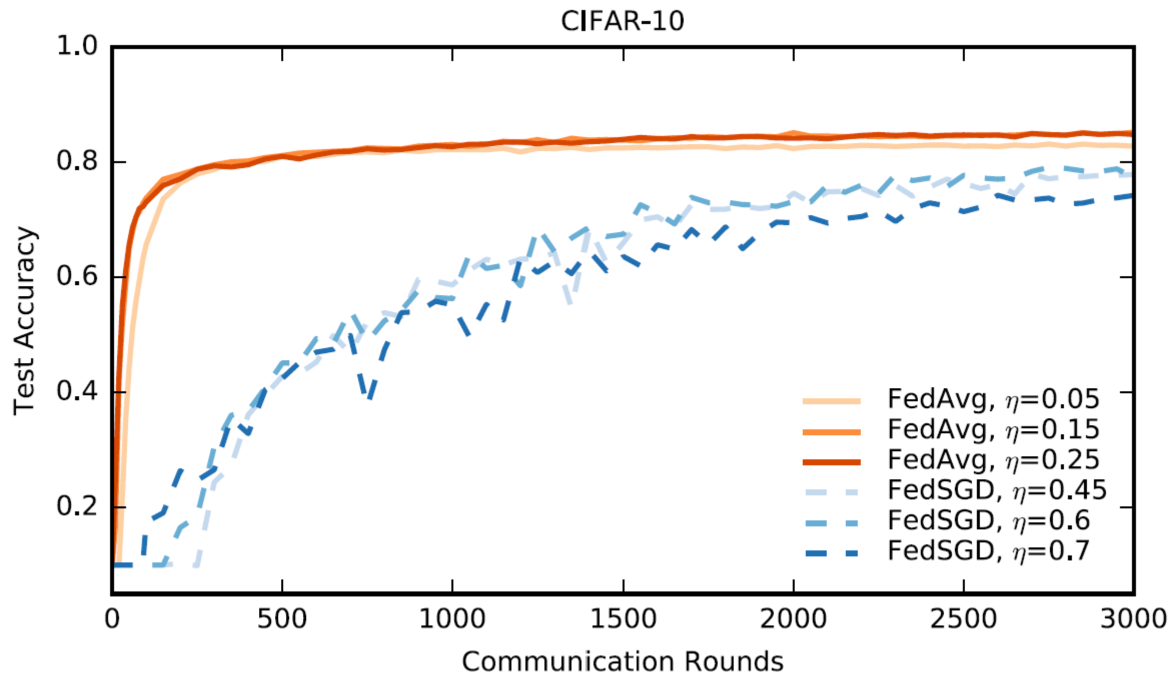
Algorithm ClientUpdate(k, θ)

Parameters: batch size B, number of local steps L, learning rate η

for each local step $1, \ldots, L$ **do**
  $\mathcal{B} \leftarrow$ mini-batch of B examples from $\mathcal{D}_k$
  $\theta \leftarrow \theta - \frac{n_k}{B} \eta \sum_{d \in \mathcal{B}} \nabla f(\theta; d)$
send θ to server

- For $L = 1$ and $p = 1$, it is equivalent to classic **parallel SGD**: updates are aggregated and the model synchronized at each step
- For $L > 1$: each client performs **multiple local SGD steps** before communicating

FedAvg (a.k.a. Local SGD)



- **FedAvg with $L > 1$ allows to reduce the number of communication rounds**, which is often the bottleneck in FL (especially in the cross-device setting)
- It empirically achieves better generalization than parallel SGD with large mini-batch
- Convergence to the optimal model can be guaranteed for i.i.d. data [Stich, 2019] [Woodworth et al., 2020] but **issues arise in strongly non-i.i.d. case** (more on this later)

Fully Decentralized Setting

- We can derive algorithms similar to FedAvg for the **fully decentralized setting**, where parties do not rely on a server for aggregating updates
- Let $G = (\{1, \ldots, K\}, E)$ be a connected undirected graph where nodes are parties and an edge $\{k, l\} \in E$ indicates that k and l can exchange messages
- Let $W \in [0,1]^{K \times K}$ be a symmetric, doubly stochastic matrix such that $W_{k,l} = 0$ if and only if $\{k, l\} \notin E$
- Given models $\Theta = [\theta_1, \ldots, \theta_K]$ for each party, $W\Theta$ corresponds to a **weighted aggregation among neighboring nodes** in G:

$$[W\Theta]_k = \sum_{l \in \mathcal{N}_k} W_{k,l}\theta_l, \quad \text{where } \mathcal{N}_k = \{l : \{k, l\} \in E\}$$

Fully Decentralized Setting

Algorithm Fully decentralized SGD (run by party k)

Parameters: batch size B, learning rate η, sequence of matrices $W^{(t)}$

initialize $\theta_k^{(0)}$

for each round $t = 0, 1, \ldots$ **do**

$\mathcal{B} \leftarrow$ mini-batch of B examples from $\mathcal{D}_k$

$\theta_k^{(t+\frac{1}{2})} \leftarrow \theta_k^{(t)} - \frac{n_k}{B}\eta \sum_{d\in\mathcal{B}} \nabla f(\theta_k^{(t)}; d)$

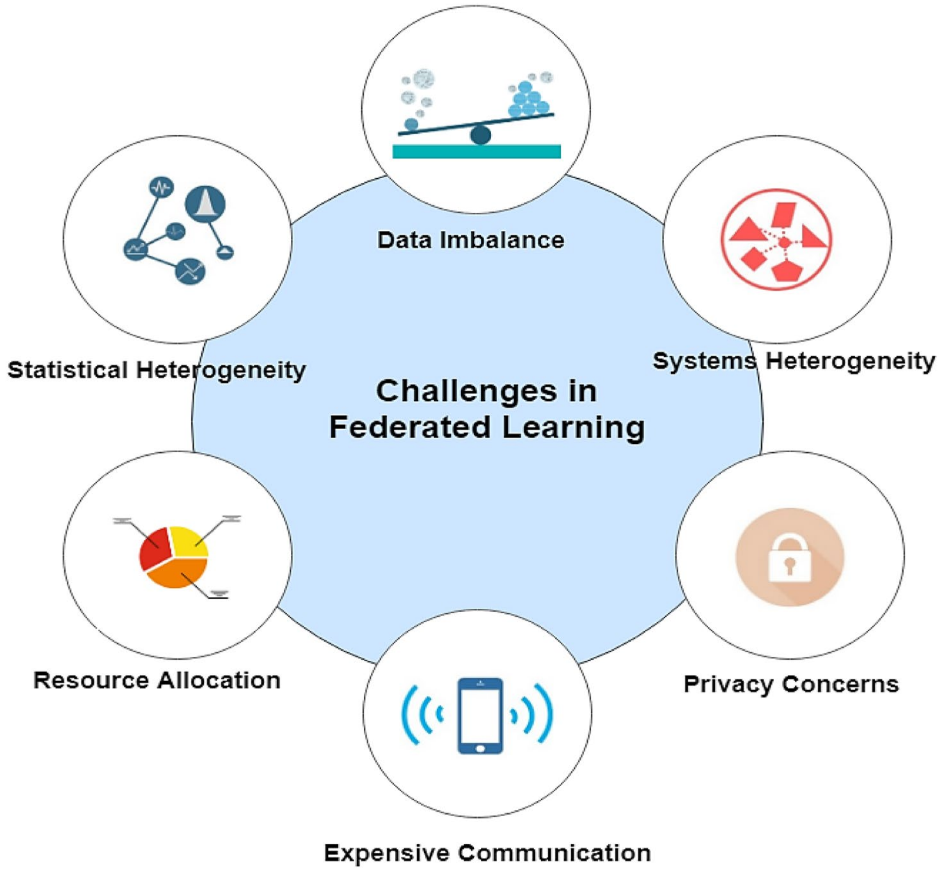
$\theta_k^{(t+1)} \leftarrow \sum_{l\in\mathcal{N}_k^{(t)}} W_{k,l}^{(t)} \theta_l^{(t+\frac{1}{2})}$

- Decentralized SGD alternates between **local updates** and **local aggregation**
- Doing multiple local steps is equivalent to choosing $W^{(t)} = I_n$ in some of the rounds
- **The convergence rate depends on the topology** (the more connected, the faster)

Challenges

- **Communication Overhead**: Federated Learning can sometimes slow down due to communication between devices and servers.

- **Data Differences**: Devices may have different types or amounts of data, making it tricky to combine their updates seamlessly.



Use Cases

- Healthcare
- Telecommunications
- Finance
- Smart Grid Optimization
- Manufacturing and Industry 4.0
- Autonomous Vehicles
- Agriculture and Precision Farming

Implementation

➢ **Choosing Frameworks**
- Select frameworks like TensorFlow Federated or PySyft for implementing Federated Learning.
- TensorFlow Federated (TFF)
  - Developed by Google helps define Federated Learning tasks and manage communication.
- PySyft
  - Built on PyTorch, it ensures privacy in computations using techniques like differential privacy
- These tools provide the necessary resources to manage training across devices.

➢ **Scaling Up**
- Ensure scalability by optimizing communication and aggregation processes.
- Balancing loads, managing resources, and maintaining reliability are vital for efficient operation.

Top Federated learning frameworks

FATE

TensorFlow federated

NVIDIA FLARE

Flower

OpenFL intel

PySyft

FedML

Federated Learning

Adapted from Miloudi Amara's slides

TensorFlow Federated

- **TensorFlow Federated (TFF)**: Building Blocks for Distributed Learning
  - **Open-source** and flexible framework by Google AI
  - **High-level API** for defining federated computations and algorithms
  - **Supports various** machine learning models and distributed architectures

Pysyft

- **PySyft**: Secure and Private Federated Learning with Python
  - **Secure** enclaves for data privacy and computation
  - **Focus** on secure aggregation and model poisoning prevention
  - **Easy integration** with existing Python libraries and tools.

Flower

- **Flower**: Orchestrating Federated Learning Workflows
  - **Lightweight** and flexible framework for managing federated training
  - **Focus** on orchestration, communication, and resource management
  - **Agnostic** to underlying machine learning libraries and frameworks.

Synthetic Data Generation

Challenges with Real Datasets

- Coverage
- It may not be feasible to get samples for all categories
- Lighting conditions
- Modifications (Glasses/No glasses, Moustache/ No Moustache etc.)
- Positions

Acne

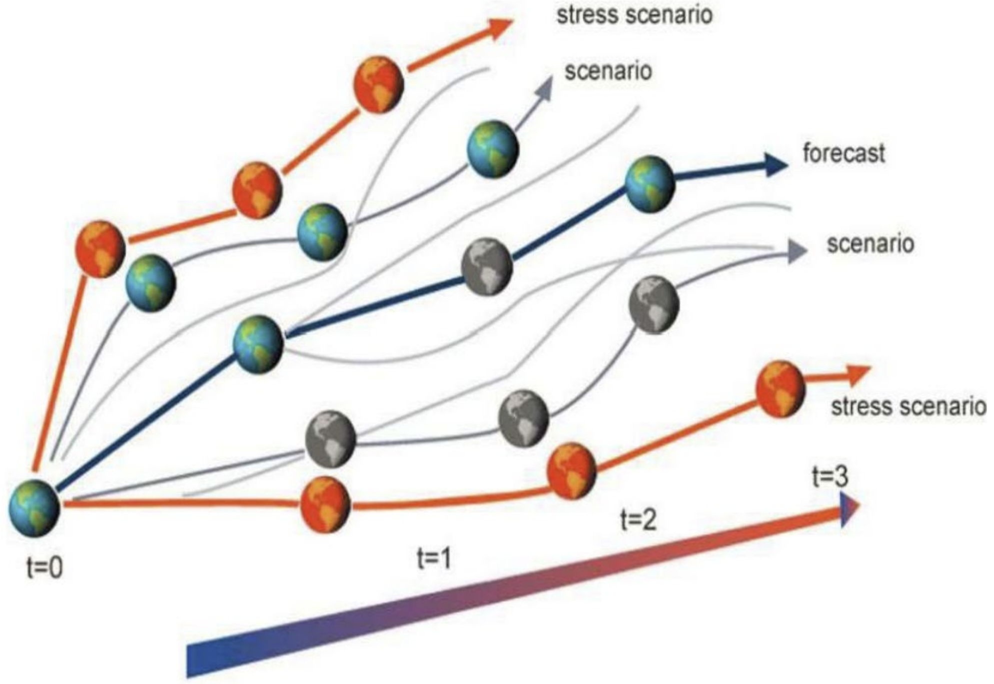
Skin redness

Bags under the eyes

https://huggingface.co/datasets/TrainingDataPro/dermatology-dataset-acne-redness-and-bags-under-the-eyes

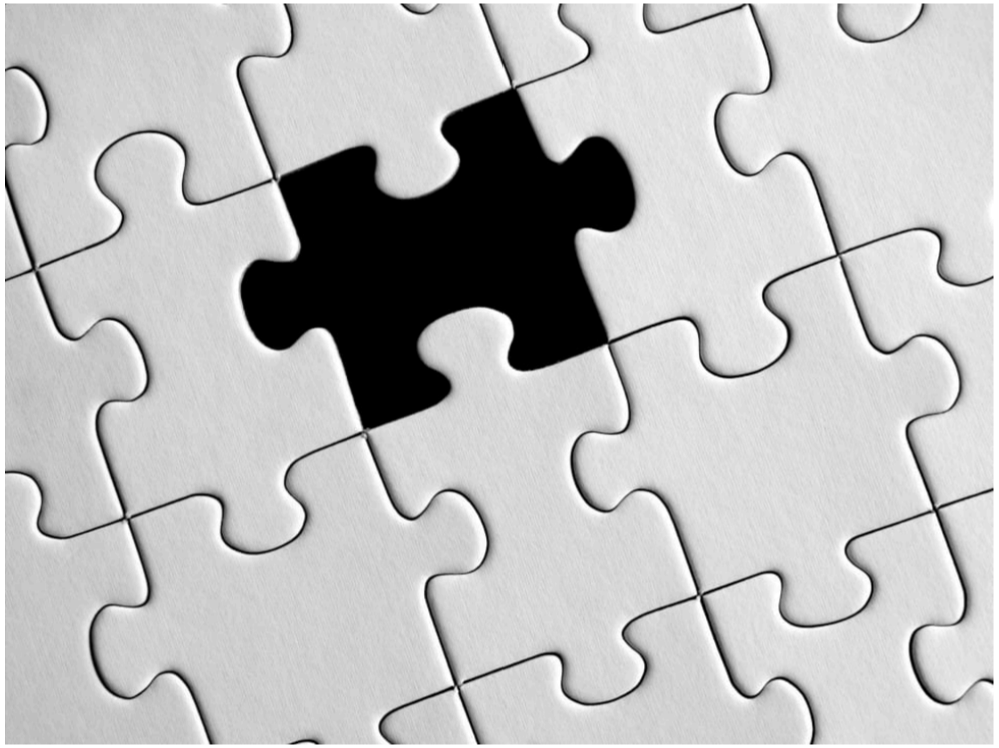
Challenges with Real Datasets

- All scenarios haven't played out
  - Stress scenarios
  - What-if scenarios



Challenges with Real Datasets

- Missing values
  - Missing at random
  - Missing sequences
  - Need data to fill frames



Challenges with Real Datasets

- Access
  - Hard to find
  - Rare class problems
  - Privacy concerns making it difficult to share

Challenges with Real Datasets

- Imbalanced
  - Need more samples of rare class
  - Need proxies for data points that were not observed or recorded

Challenges with Real Datasets

- Labels
  - Human labeling is hard
  - Synthetic label generators

Open-source Tools

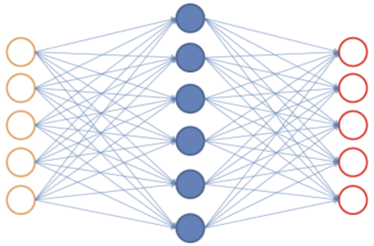
- Faker
- Synthetic Data Vault
- Data Synthesizer
- Synthpop
- VAE
- GAN
- WGAN

Generative Adversarial Nets (GANs) [GPM+14]

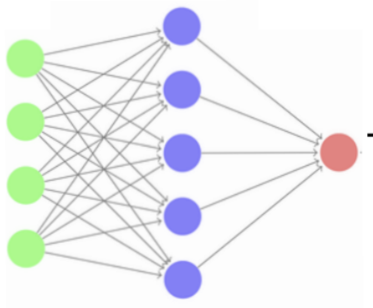
2-Player Zero-Sum Game

Generator G:
mimic the real data

Noise
z

→

→ Synthetic data
x

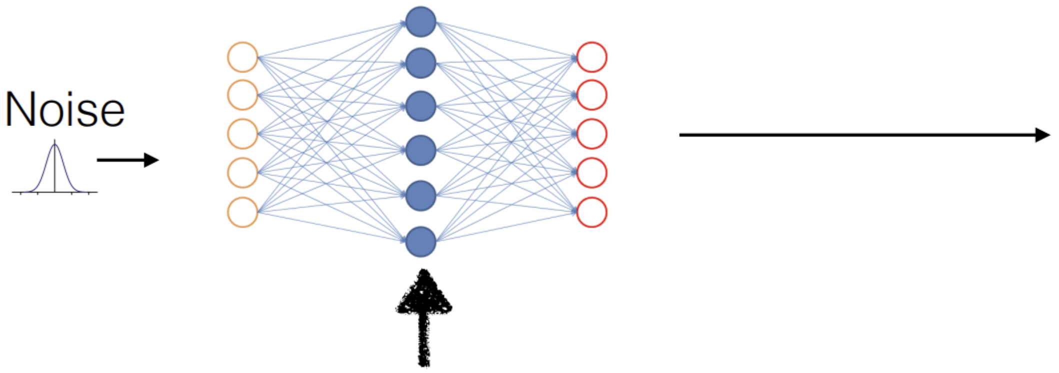
Discriminator D:
distinguish real and fake data

Real/Fake
Examples
x
→

→ Probability of
"real"

Wasserstein GAN [ACB17]

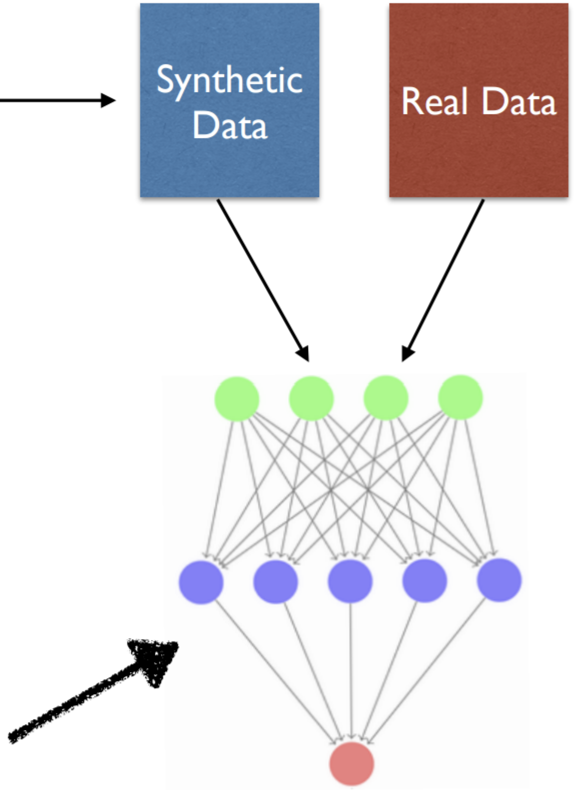
$$\min_{G} \max_{D} \mathbb{E}_{x \sim p_X}[D(x)] + \mathbb{E}_{z \sim p_z}[1 - D(G(z))]$$

Private GAN Training



Training Generator:

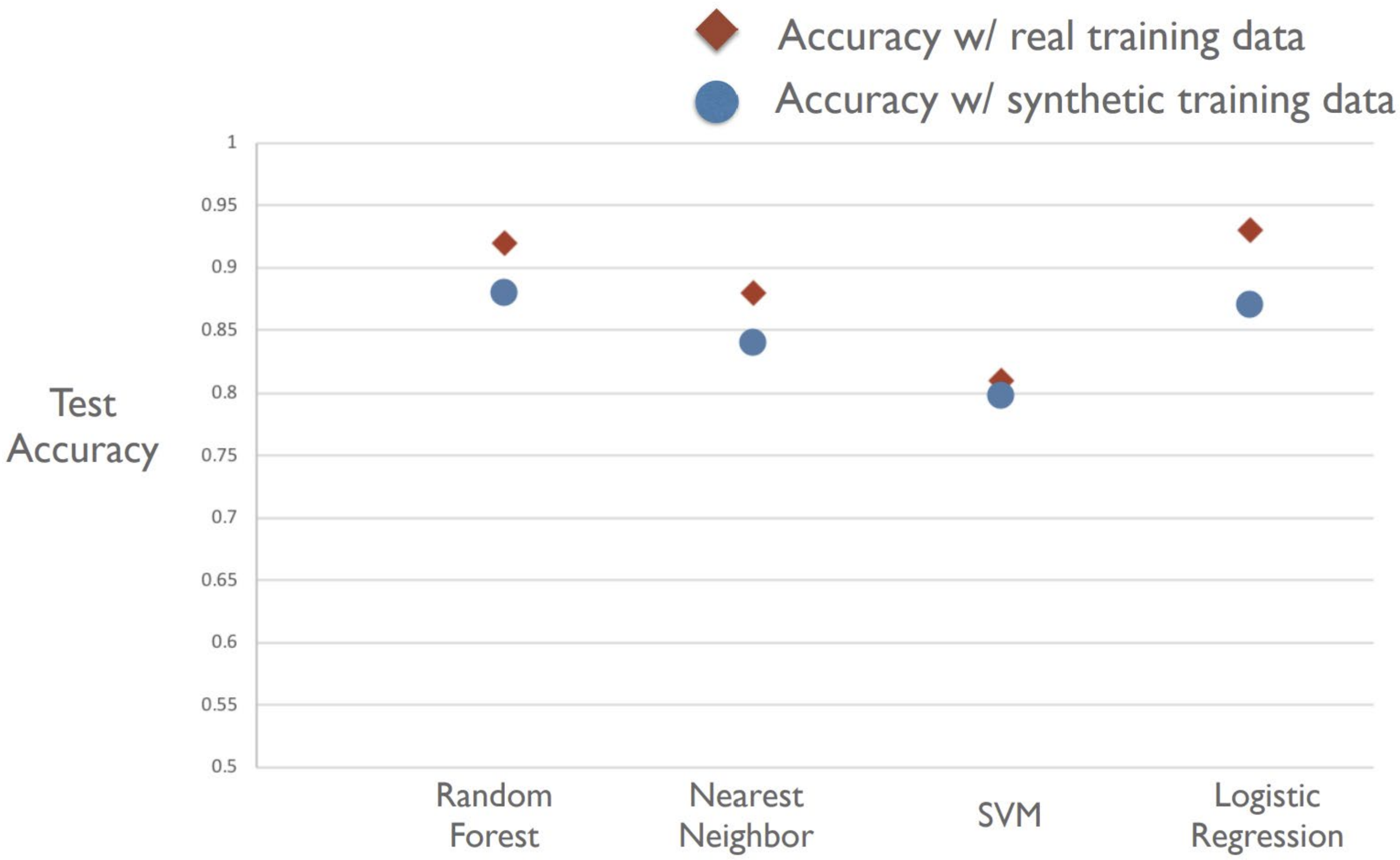
- Does not directly interact with real data
- Train using standard (non-private) methods (e.g., SGD)



Privately Training Discriminator:

- Interacts with real data
- Train using DP method such as DP-SGD

Models Trained on Synthetic v.s. Real Data



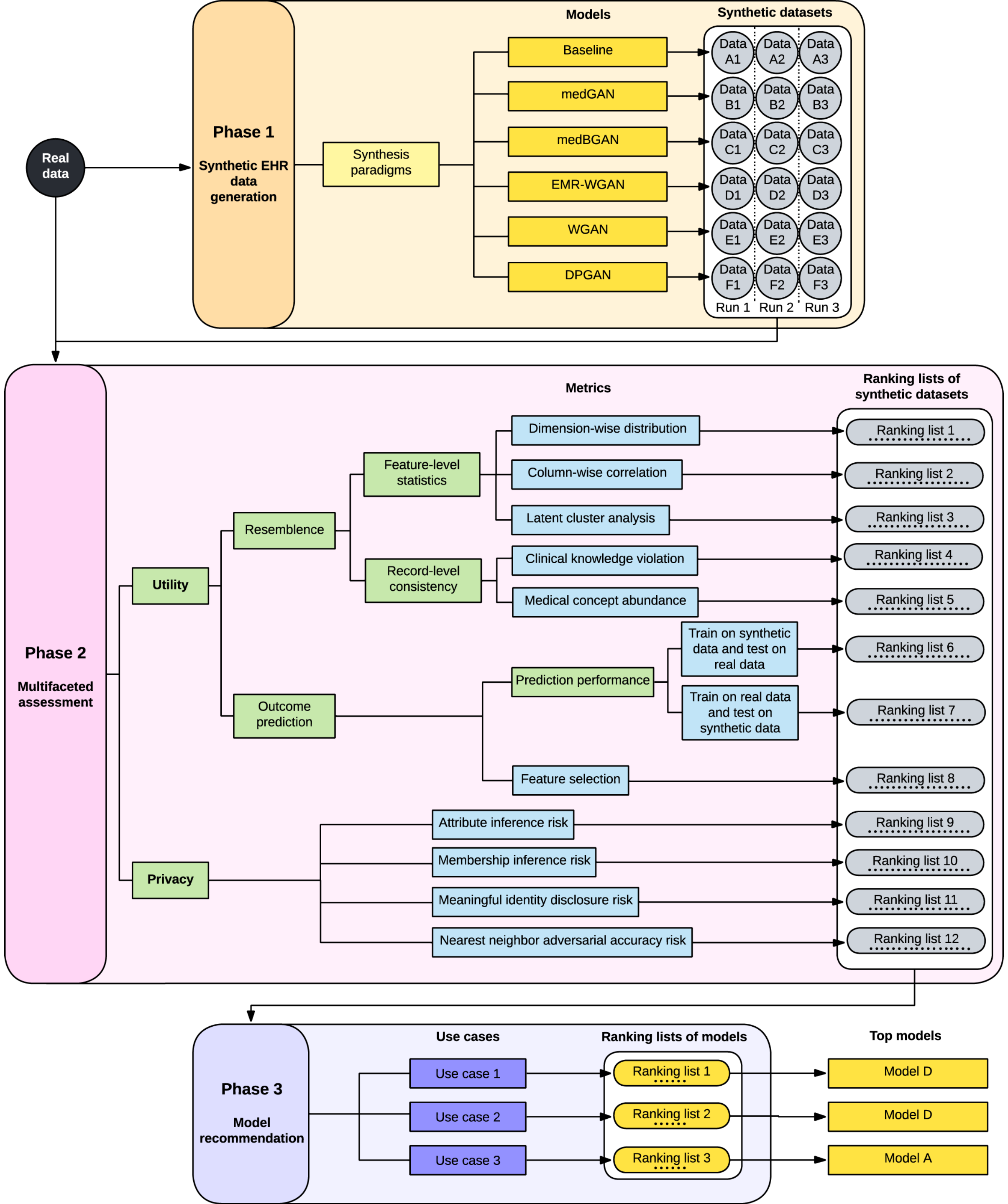
Difficult to Reach Convergence

- Training procedures a sequence (generator, discriminator)
- The last generator often gives poor synthetic data distribution
- But mixture of generators can provide good synthetic data

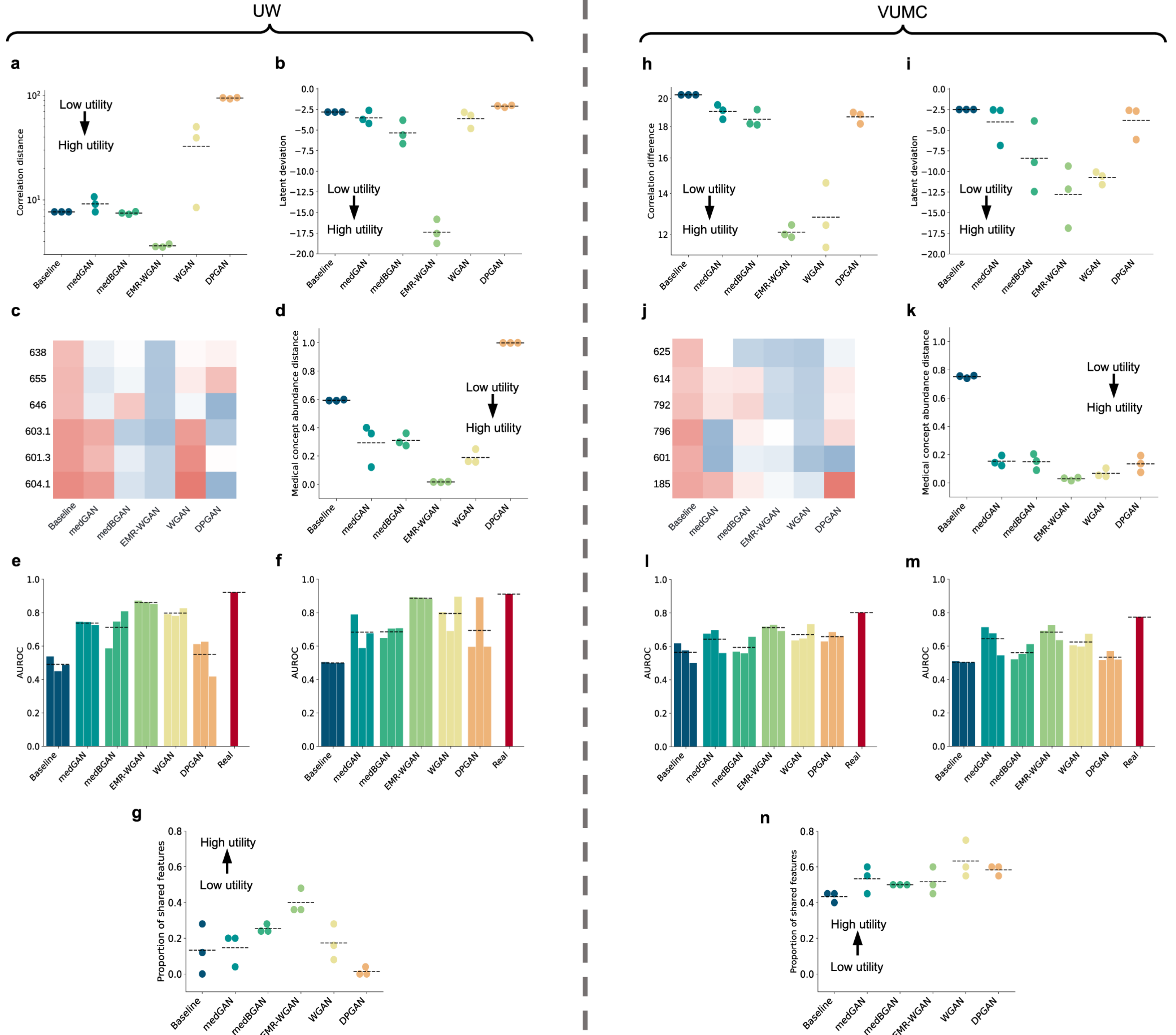
Synthetic Data Release

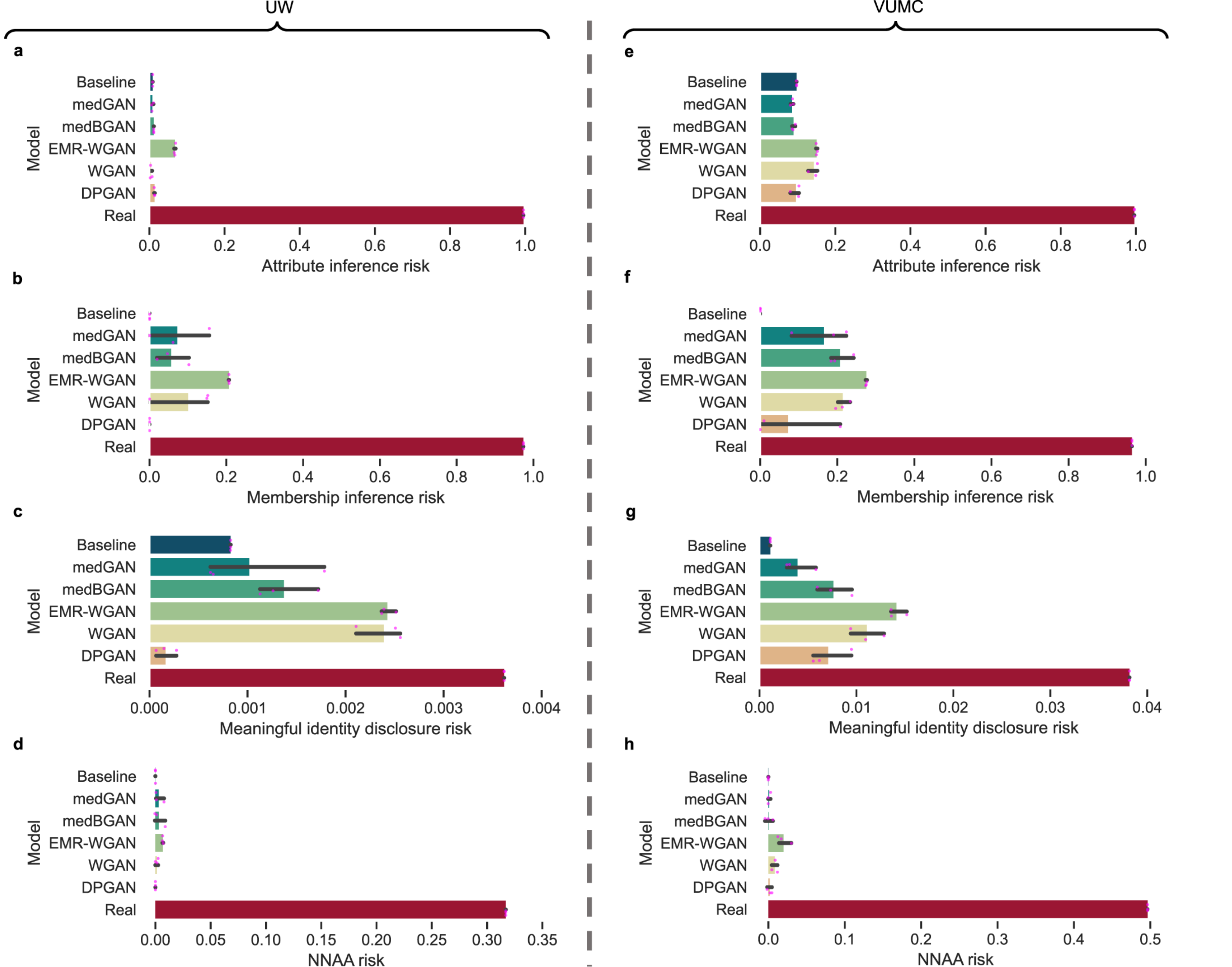
- Synthetic data for query/statistics release
  - A large collection of statistics in mind
- 2. General-purpose synthetic data
  - Exploratory data analysis
  - Training ML models

Evaluation



Yan C*, Yan Y*, **Wan Z***, Zhang Z, Omberg L, Guinney J, Mooney SD, Malin BA. **A multifaceted benchmarking of synthetic electronic health record generation models**. *Nature communications*. 2022 Dec 9;13(1):7609.


UW
VUMC
a
b
c
d
e
f
g
h
i
j
k
l
m
n
Low utility
High utility
Correlation distance
Latent deviation
Medical concept abundance distance
AUROC
Proportion of shared features
Correlation difference
Baseline
medGAN
medBGAN
EMR-WGAN
WGAN
DPGAN
Real
638
655
646
603.1
601.3
604.1
625
614
792
796
601
185


UW
VUMC
a
b
c
d
e
f
g
h
Model
Baseline
medGAN
medBGAN
EMR-WGAN
WGAN
DPGAN
Real
Attribute inference risk
Membership inference risk
Meaningful identity disclosure risk
NNAA risk

Readings for the Next Week

- 1. N/A

- Optional
  - N/A

Feedback Survey

- One thing you learned or felt was valuable from today’s class & reading

- Muddiest point: what, if anything, feels unclear, confusing or “muddy”

- https://www.wjx.cn/vm/hX0mIro.aspx