Transparent Decisions: Selective Information Disclosure To Generate Synthetic Data

Carlos Gavidia-Calderon[1], Steve Harris[2],[3], Markus Hauru[1], Florimond Houssiau[1], Carsten Maple[1,4], Iain Stenson[1] and May Yong[1]

[1]The Alan Turing Institute
[2]University College London
[3]NIHR University College London Hospitals BRC
[4]University of Warwick

Abstract

The UK government and the public wish to see the National Health Service (NHS) use data and Artificial Intelligence for public good [13][16]. However, there is a major challenge in making health data available for research whilst respecting patient privacy. Synthetic data generation is an emerging technique that enables access to data that, in some way, shares the characteristics of the original data. In this paper we introduce SqlSynthGen (SSG), a method for generating synthetic relational datasets. SSG offers a human-readable, risk-guided approach to refining data fidelity while managing disclosure risk. This paper presents SSG, specifically focusing on its application for generating synthetic data from NHS hospitals.

1 Introduction

Hospitals electronic health record systems are typically built using relational databases containing millions of records. While hospital staff access this data for their clinical duties, other professional communities— scientists, software engineers and educators — rightly must follow lengthy processes to be granted access. Controls are in place to ensure patient data —which is both sensitive and valuable [28]— is accessed for only legitimate reasons. Current practices involve preparing employee contracts, implementing de-identification or anonymisation mechanisms to remove personal information, and accessing data only via Trusted Research Environments [14].

While protecting patient privacy is of utmost importance, these processes impede collaboration and engagement, and introduce delays to researchers already working to arduous grant deadlines. For instance, researchers can use data to improve diagnostic accuracy, refine our understanding of diseases, or develop personalised treatments [30]. Patient data can be used to train the next generation of healthcare practitioners and researchers. Synthetic data is an accelerator: it can provide a simulcrum with the characteristics of patient data that can be shared onwardly. This can be used to support education and training, to quality control applications and code, and to test reproducible analytical pipelines in the open. This will accelerate academic progress for patient benefit.

In order to both protect user privacy and control access, current techniques employ mechanisms including data agreements, de-identification or anonymisation, aggregation over the original data, and provision of trusted

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

research environments (TRE) for access by third parties. While these techniques provide an extra layer of protection, they are not immune to vulnerabilities [21]. For example, de-identified data releases are still susceptible to linkage attacks. Aggregation requires releasing only aggregate population metrics, such as counts or averages, but outliers remain vulnerable to identification [30]. Instead of releasing real patient data —either partial or aggregate— an option is to release synthetic patient data.

Synthetic data is data that is manufactured, as opposed to real data that is collected from real-life events and people. Synthetic data generators (SDG) use algorithms to produce synthetic data entries while preserving statistical properties of the real dataset. There are multiple SDG approaches in the literature, each one targeting a specific data type, such as tabular data or time-series data [17]. SDGs can, when appropriately constructed, offer mathematical guarantees of the preservation of user privacy [19, 8] by incorporating differential privacy.

In this paper, we describe our work on developing a new SDG approach at the University College London Hospitals (UCLH) NHS Foundation Trust. Each year, UCLH admits 100,000 patients and stores their data in a relational database. Broadly, we discover that these are their requirements regarding their utilisation:

- **REQ-1:** The synthetic datasets should be in the form of relational datasets for any given relational schema
- **REQ-2:** The generator can manufacture synthetic data by utilising aggregates and statistical properties extracted from real patients
- **REQ-3:** Ensure that information disclosed about real patients are easily understandable by humans.

Listing 1: Requirements for Synthetic Data Generation at UCLH Trust

We developed SQLSYNTHGEN [12] to meet the requirements in Listing 1. SQLSYNTHGEN is an open-source Python package that can replicate the database schema of a relational database. Once the replica is in place, SQLSYNTHGEN can generate synthetic samples at different levels of fidelity: from low-fidelity random values compliant with the database schema, to high-fidelity samples from probability distributions learned from real data.

SQLSYNTHGEN uses a white-box approach where information extraction from real data are expressed as SQL queries in human-readable format, rather than black-box approaches, such as deep generative models with thousands of parameters [6]. For ensuring patient privacy, SQLSYNTHGEN supports differential privacy (DP)[10] to add quantifiable noise to the information extracted from the real data.

2 Sharing Patient Data

This section starts by enumerating motivations for sharing patient data. An understanding of motivations is important because these determine the requirements of appropriate data sharing mechanisms. The reasoning for sharing data dictates what minimum data needs to be shared, and this in turn defines the requirements to be met if the data is to be shared reasonably safely.

We then survey the current privacy preservation practices currently adopted by hospitals to enable collaborators controlled access to hospital data. We show that these are a) linked to inadequate privacy protection measures [21, 30], or b) a cause of unnecessary friction to analysis [23]. While synthetic data is considered a potential solution to overcome the above challenges, many patient datasets are organised as relational databases. Current synthetic data generators have limitations: a) they do not address the unique challenges of the relational structures [22][32]; b) they require users to specify dataset schemas [29]; or c) they can achieve differentially private, explainable, high-fidelity synthetic data for relational databases but currently face limitations in scalability. [8].

2.1 On the Benefits of Sharing Patient Data

Enhancing Research Quality and Innovation: Collaboration can lead to more comprehensive research studies, allowing healthcare practitioners and researchers to test hypotheses or observe trends across a broader dataset than is available internally. How well a dataset represents the true distribution matters more than simply dataset size[2]. In the medical domain, where lack of data is a common occurrence, the amalgamation of diverse datasets has a better chance of representing true underlying distributions.

Access to Specialised Expertise: External collaborators bring specialised knowledge and skills that complement the in-house capabilities of a hospital. For example, collaborations with methodology researchers can lead to state-of-the-art data analysis and interpretation, thereby improving both method development and treatment outcomes. Software engineers and machine learning operations engineers can build customised cyber-physical infrastructure to support analysis of patient data in real time[14].

Accelerating Medical Discoveries: By pooling resources and data between hospitals, research can proceed at a faster pace[2], potentially leading to quicker discoveries in disease mechanisms, treatment effectiveness, and development of new therapies or medical technologies. Sharing patient data can facilitate the recruitment of participants for clinical trials, ensuring a diverse and adequate sample size. This can be crucial in studying rare diseases or sub-types of common diseases, especially in hospitals that offer specialisations not commonly offered elsewhere in the world.

Expanding Research Funding Opportunities: Collaborative research often has better chances of securing funding[31]. Funding bodies frequently encourage or require collaboration across institutions as a criterion for grants, viewing it as a way to maximise the impact of their investment.

Bench-marking and Quality Improvement: Comparing data across institutions can help identify best practices and areas for improvement in patient care and management. This bench-marking is used to drive quality improvement initiatives within a hospital[33].

Education and Training: Collaborations provide educational opportunities to clinical research employees at hospitals, researchers and students at universities and research institutions, exposing them to different perspectives, methodologies, and cutting-edge research through joint ventures and knowledge exchanges.

Building Networks and Reputation: Collaborations can enhance a hospital's reputation in the medical and scientific community[31]. They extend the hospital's influence and recognition, which can attract top talent and more collaborations in the future.

2.2 Current Practices For Sharing Patient Data

De-identification and Anonymisation of Patient Data: De-identification is the process of obscuring or replacing personal identifiers to prevent the direct association of data with an individual. Common de-identification methods include explicit removal, masking or pseudonymisation of direct identifiers, and aggregating data to remove specificity eg. binning.

Anonymisation aims to ensure that data cannot be linked back to an individual by any means. Anonymisation strips datasets of all personal identifying information but it is not provable when this has been achieved. Conservative measures will strip a lot of information thereby heavily affecting the value of the dataset, and we still cannot be certain that there is not some way to de-anonymise.

For example, the removal of timestamps from a medical dataset as part of an de-identification or anonymisation process is performed because timestamps can be used to re-identify a patient by linking a patient's records over multiple de-identified datasets. The pattern of timestamps can disclose information about a patient's health, as well as their frequencies away from home.

However the stripping of timestamps from a medical dataset erases important information because medical information is highly time-contextual. Part of the richness of medical data is its time-series nature. Medical data that has been stripped of time stamps has reduced richness of data and is limited what can be learnt from it.

Effectiveness of both de-identification and anonymisation techniques is highly dependent on context, which includes the dimensionality, volume, and statistical properties of data. Other important aspects that need to be considered include which types of applications or analyses the data are to be used for, whether the data will be released publicly or with additional access control, and whether the data are tabular, relational, or have longitudinal or transactional characteristics.

Trusted Research Environments: Trusted Research Environments (TREs) are an important part of the data sharing mechanism ecosystem. TREs are the secure infrastructure and governance model that allows researchers to access and analyse data; they are often used in conjunction with other data-sharing mechanisms.

TREs play a major role in controlling data access levels. Initially, data access is controlled through secure authentication and authorisation mechanisms. This means that only approved researchers can access the data, and they can only access specific datasets approved for their role and research projects. Activities in TREs are closely monitored and logged.

In addition, TREs provide both physical and virtual security. Data in TREs are often stored in physically protected facilities. Virtual security measures such as firewalls, intrusion detection systems and regular penetration testing maximise protection against external threats. Finally, to ensure no privacy leakage, data egress from TREs is restricted. Researchers can analyse data within TREs but cannot take it out.

This means that working with data within TREs is far from a comfortable experience [23]. In order to provide security measures, computational resources can be limited and the list of approved software packages for analysis is restricted and not easily updated. There is significant process overhead generated by the need for detailed authentication into remote machines, activity logging, monitoring and compliance checks. There is a steep learning curve in working within a TRE, and new users are heavily dependent on support staff for technical assistance. Finally, the inability to egress data limits the sharing of interim findings and prevents close collaboration on ongoing data analysis.

Honorary contracts and data agreements: In order for non-hospital/clinical staff to work with medical data, they typically either need to become honorary employees of a trust or their current institution need to enter into a data sharing agreement with the trust. Both are lengthy and restrictive.

The process of obtaining an honorary contract typically begins with an initial inquiry and application to the relevant department or clinical group at the hospital. This is followed by credential verification and background checks, including border security investigations. Once these checks are satisfactorily completed, the relevant departments can grant approval.

To get a data agreement signed between two institutions, the first step is to identify the need for data sharing, specifying what data will be shared and how it will be used. Next, security requirements for storing, protecting, and accessing the data must be agreed upon by both parties. All these elements need to comply with relevant regulations. Finally, the agreement must be reviewed by the legal and compliance teams of both institutions to ensure all requirements are met and all parties are protected.

3 From Sharing Real Data to Sharing Synthetic Data

Real data is recorded from real life. Synthetic data is manufactured data, and can be created such that data elements are random, structurally or type accurate, or have distributions that mirror statistical properties of another dataset. In the last case, statistical properties can be directly or indirectly observed, to inform the data manufacturing process. When any properties of one dataset is used to guide the manufacturing process of another dataset, the first dataset is referred to as the 'real' or 'original' data. In the use case presented in this paper, 'real' data is hospital patient data. Our manufactured data is commonly referred to as 'synthetic' data.

While manufactured patient data is not about real individuals, it is a fallacy to imagine that adoption of synthetic data in data sharing practices prevents disclosure of sensitive information. This section shows how synthetic data generators can manufacture outputs which disclose more, or less sensitive information, and how this affects the ways in which outputs can be used.

3.1 Synthetic Data Generators

Synthetic data generators (SDG) manufacture data. There is a tension observed in the process of manufacturing synthetic data which involves three factors: fidelity, utility and privacy. Fidelity measures the extent to which synthetic data resembles the real dataset. Utility is the measure of the usefulness of synthetic data to a given task. Privacy is a measure of the information disclosed about the real dataset during generation of the synthetic dataset. These three factors inform the manufacturing process and limit the ways its outputs can be used. Synthetic data which is very similar to the real dataset (high fidelity) risk leaking information about real patients (low privacy). Conversely, low fidelity datasets typically contain little information relating to the real data, so individuals are unlikely to be identified. However, this low fidelity also limits the dataset's utility. For instance, medical data stripped of personal identifiers such as timestamps loses its richness and reduces the scope of insights that can be derived from it.

However, low-fidelity or coarse-grained datasets can still be useful, as utility is dependent on the context or task. In some cases, low-fidelity datasets are valuable if they provide sufficient information for engineering applications e.g. software testing. When paired with real data, multi-fidelity datasets can reduce computational costs and prevent over-fitting in machine learning tasks [26][27][5]. Low fidelity datasets can remove blockers at the beginning of research for initial exploration, building pipelines, and testing models. These tasks can be conducted in a secure environment restricted to students and researchers, with scripts later ported to the hospital for training on real data if the initial analysis proves promising.

This means that there is a class of low-fidelity datasets that is useful in common research and engineering tasks. The benefits of using these datasets can be realised with little cost to patient privacy.

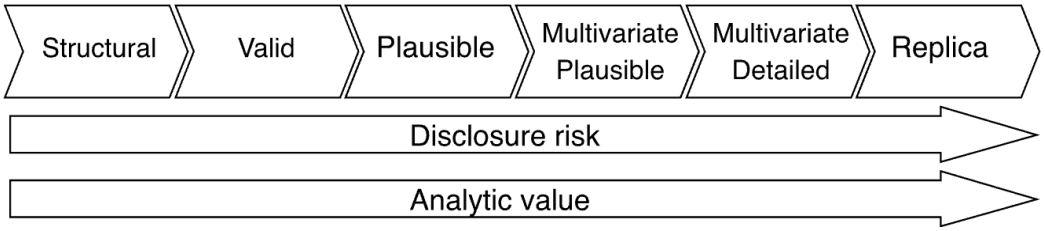

Figure 1: Shows the range of fidelity for synthetic data. High fidelity data can result in higher utility, but also increased risk of identification. Sourced from UK Office of National Statistics[24].

The UK Office of National Statistics [24] have defined a spectrum of fidelity for synthetic data, shown in Figure 1. In the context of healthcare relational datasets:

- **Structurally correct datasets** have the same column names, tables and relationships as real data.

- **Valid datasets** imply that the values in the synthetic dataset are correct and valid, e.g. date of births are valid dates.
- **Plausible datasets** imply that the relationship between values are realistic, e.g. a patient's date of death is not before their date of birth.
- **Multivariate plausible datasets** implies that the values are correlated across different variables, e.g. a male patient is likely to be both heavier and taller than a female patient.
- **Multivariate detailed datasets** are more realistic than a multivariate plausible data set, but less than a replica of the real data. An example are rows of data showing that a patient with a diabetes diagnosis has more records of blood sugar readings than a patient with a broken bone.

3.2 Synthetic Data For UCLH NHS Trust

University College London Hospitals National Health Services Foundation (UCLH NHS) Trust is a pioneering institution within the UK, renowned for its treatment care and specialist services not widely available in other NHS Trusts. It is closely affiliated with University College London; this is a partnership that emphasises research and education, integrating medical research and teaching at the undergraduate and postgraduate levels directly into the clinical environment. As an institute that emphasises medical care, research and education, and as custodians of highly sensitive medical data, UCLH NHS Trust are in a position to leverage research capabilities to supercharge innovation if they can develop a process for thoughtful access to this data. However, consequences of unintentionally releasing identifiable information include loss of individuals' privacy, loss of institutional prestige, as well as substantial legal fines.

3.2.1 Problem Statement

Machine learning (ML) infrastructure are deployed in hospitals to enable AI in healthcare delivery and administration. ML infrastructure supports tasks such as structuring data from electronic health records into a format that can be used as inputs to AI algorithms, deploying image analysis and predictive analysis tools, and presenting the results to healthcare practitioners in a timely and useful format.

To achieve these tasks, engineers who build the infrastructure need to gain an understanding of the data structures and data flow within the hospital. Researchers need to evaluate if target datasets meet their purposes for hypothesis testing, and are adequate in terms of quality and quantity. It is onerous to issue contracts to entire teams of engineers, researchers and students, but there are no other ways to share data with external collaborators.

However, what engineers and researchers need when working on early stages of exploratory analysis to understand data in terms of content, structure and data flow is information about the data, rather than having access to individual rows of data itself. Here is an opportunity to frame the problem as: What information can be released about sensitive data, which is maximally beneficial to engineers and researchers, with minimal cost to patient privacy?

3.2.2 Requirements

Listing 1 enumerates the requirements of building synthetic data generators for UCLH Trust. This section expands on each requirement; the following section demonstrates how the design of SSG fulfils these requirements.

Produce relational datasets for any given schema: Many data holders, including hospitals, store patient electronic health records in relational databases. Data is often structured within complex schema that capture both single observations and time series data. These relational databases also include tables for vocabularies such as definitions of drugs, observations and diagnoses.

Under this requirement, a minimally useful synthetic dataset must at the very least a) be structurally correct. That is, it will contain the same tables, columns, and data types as the real data, and b) meet foreign key constraints. In order to increase analytical value as shown in Figure 1, the synthetic generator will need to generate values which are valid and plausible, e.g. valid gender values and a plausible distribution of height and weight. A multivariate plausible dataset will have values that correlate across multiple tables, e.g. the correlation between gender and height are represented across the 'Demographic' and 'Observation' tables.

An additional complexity here is in generating synthetic time series data, e.g. blood pressure values every ten minutes for a patient in intensive care unit. In order to be multivariate plausible, the data needs to contain the correct frequencies for data collection as well as plausible values that depend on a patient's physiology. This is generated across multiple tables as well.

Generate synthetic data using statistical properties computed from real patients Hospitals are mandated or encouraged by various information acts to release hospital information to the public. The main reasons for this are a) allowing insights into quality of care provided by public or insurance funds and b) to enable patients to make informed decisions regarding where to seek care based on hospital performance and specialisations[33].

The type of information that is released in the public domain includes quality of care indicators, patient safety data, readmission rates and service availability. This includes aggregate data about patient outcomes, infection rates, details on specialised services, bed occupancy, Accidents and Emergency (A&E) wait times as well as statistical properties on patients returning for treatment within a period of discharge. This information is published regularly and does not compromise individual patient privacy.

Synthetic data generators can use aggregate data and statistical properties of real data to generate datasets which are measurably closer to real data. A synthetic dataset generated using public information is unlikely to reveal any additional patient information beyond what is already publicly available.

Ensure that information disclosed about real patients are easily understandable by humans. Aggregates and statistical properties are well-understood mathematical concepts. A comprehensive explanation of such information extracted from real patients datasets for the purpose of generating synthetic data should cover the following three points:

1. **Extracted Information**: Detail what specific information about patients has been extracted.
2. **Computation Process**: Explain how this information is computed.
3. **Usage for Synthetic Data**: Describe how this information is used to shape the synthetic data.

Providing this explanation in a single, human-readable source ensures consistency and prevents obsolescence across multiple data generation iterations. This offers a clear audit trail of the generation process and helps identify the disclosure risks of its outputs.

The concept of synthetic data is complex, people may not understand how data that does not represent real individuals still needs privacy considerations. It is furthermore difficult to understand how the application of differential privacy to aggregates and statistics can provide additional protection.

Differential privacy (DP) [10] is the gold standard that protects individuals within a dataset while still allowing for the useful analysis of the aggregate data. Its internal mechanics of noise addition for the purpose of privacy preservation can leave users without a clear understanding of its outputs and how to interpret them correctly[9].

The application of differential privacy to synthetic data compounds the explanations' complexities. There is a struggle to understand how DP offers probabilistic but not absolute guarantees. Explaining this to custodians of highly sensitive data is difficult because privacy is expected but not always technically feasible.

However, this is an important discussion, there is a necessary understanding to be achieved here because the interplay between privacy and utility governs the results of a differentially private synthetic data generator. The

only people who can take the responsibility for managing the balance between privacy and utility are the data custodians.

4 Generating Synthetic Data Using SQLSYNTHGEN

SQLSYNTHGEN (SSG) is a software package developed to meet the requirements outlined in Section 3.3. When connected to an existing relational database, SSG builds a new empty database with the same schema. It copies over the non-sensitive data, such as look-up tables, and generates structurally correct synthetic data with random values. Optionally, SSG can refine these synthetic values using aggregates and statistical properties. SSG can apply differential privacy to obfuscate the true values of these properties in a measurable way. The new database is then populated with these synthetic values.

4.1 Technical Overview

The default output dataset from SSG is structurally correct and has no disclosure risk. These are datasets that sit on the far left end of the spectrum in Figure 1. No information about the real dataset has been disclosed, beyond the structure in which they are stored. This can already be useful e.g. for building software testing modules and pipe-lining scripts, and can be safely released if vocabularies and schema can be shared. ***This meets REQ-1: Produce relational datasets for any given schema.***

SSG can be further configured to generate synthetic data that (in reference to Figure 1), can be as sophisticated as multivariate plausible data. This is achieved by allowing the user to define SQL statements that extract aggregate statistics and statistical properties from the real data. These extracted values are then used to shape the distributions and marginals of the synthetic data. ***This meets REQ-2: To generate synthetic data using statistical properties computed from real patients.***

As part of its process, SSG generates a human-readable audit trail that details the entire data generation process. This includes what information was extracted from real data, the methods used for extraction, the computed results, and how these values were injected into the synthetic data generation. The audit trail is a human readable file whose contents are incorporated directly into the SDG process. ***Ensure that information disclosed about real patients are easily understandable by humans.***

SSG pipeline design enables the selective production of synthetic datasets with varying levels of fidelity. Users control the shaping of synthetic data by specifying which information is extracted from real data, how it is computed, and how it is utilised. SSG's configuration supports agile development, allowing for incremental fidelity improvements as needed, while maintaining transparency, auditability, and control over privacy risks at every stage. Additionally, users have the option to apply differential privacy to protect the marginals extracted from the source data.

In order to support this design, SSG's process for generating synthetic relational datasets can be broken into three separate steps, as shown in Figure 2. They are as follows:

1. SSG **builds** a new database to store synthetic data. This new database will be populated by synthetic data generated in the next steps. Look-up tables which do not have any privacy concerns are copied over entirely, to maintain foreign key constraints.

2. By default, SSG **generates** random but structurally correct data.

3. As an option, SSG can **refine** random values for higher accuracy by using extracted statistics from real data, with or without DP. For example, mean of height by age and gender can be extracted from real patients and the correlation be used to generate higher fidelity data.

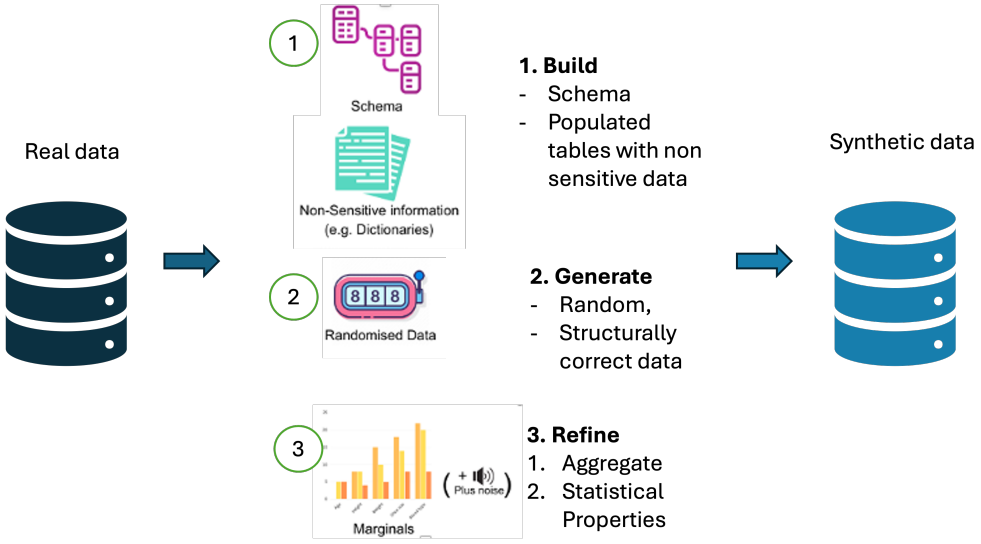

Figure 2: The processes of SQLSynthGen in order

For more information and tutorials about SQLSynthGen, please refer to our repository at `https://github.com/alan-turing-institute/sqlsynthgen`. Our repository [12] contains installation instructions, comprehensive documentation and trouble shooting guides to help get started with the software. The repository also contains a simple tutorial using a Kaggle dataset [7] as well as an advanced example based on the Observational Medical Outcomes Partnership (OMOP)[25], which provides a standardised data model for observational healthcare data.

In the following sections, we demonstrate the use of SSG in creating synthetic data based on a publicly available AirBnB Kaggle dataset [7].

4.2 Building a Replica of a Real Dataset

In this example, let us consider that our dataset is contained in a database called ‘airbnb‘ in a local PostgreSQL instance. We want to port the schema to a new ‘airbnb_synthetic‘ database, and populate the ‘airbnb_synthetic‘ database with synthetic rows that mirror some of the statistical properties of the ‘airbnb‘ dataset.

Build schema tables: We connect to the real dataset by setting connection credentials in environment variables. We run a series of commands `sqlsynthgen make-tables`, `sqlsynthgen create-tables` and `sqlsynthgen make-generators` to auto-generate two Python files.

The first file, ‘orm.py‘, outlines the structure of the PostgreSQL ‘airbnb‘ dataset by mapping each table in ’airbnb’ to a corresponding Python class. Each column in these tables is represented as a class field. This mapping is generated using SQLAlchemy[4], which is a SQL toolkit and Object-Relational Mapping (ORM) library for Python. By using SQLAlchemy in SSG for mapping, users do not need to perform any additional configuration to describe the schema of the real dataset. The ‘orm.py‘ file serves as a foundation for building a new ‘airbnb_synthetic‘ PostgreSQL database, complete with the necessary tables, columns and data types. Listing 2 shows a snippet from ‘orm.py‘ that demonstrates how the ‘users‘ table from the ‘airbnb‘ dataset is mapped as a Python class.

```python
class User(Base):
    __tablename__ = "users"

    id = Column(String, primary_key=True)
    date_account_created = Column(Date)
    ...
```

Listing 2: Section of PostgreSQL table ‘user‘ represented as a Python class

Copy over lookup tables: A lookup table, or a vocabulary, is a table used to store a predefined set of values that are referenced by other tables. They contain a finite and static set of values such as codes, names of categories or descriptions. Look-up tables are a good practice adopted help normalise databases by removing redundancy and enabling efficient data management. They work by using foreign key constraints to ensure values in related tables are consistent and valid. These foreign key constraints need to be satisfied when generating synthetic data in relational datasets. On their own, vocabularies provide only limited utility, since the more interesting aspects of the data are usually found in the non-vocabulary tables.

The fidelity of the synthetic dataset can be improved by ensuring the vocabulary tables have perfect fidelity from the beginning, since they do not raise privacy concerns (although some vocabularies are copyright-protected). In this section, we demonstrate how SSG addresses vocabulary tables by copying them in their entirety, thereby eliminating the need for synthesis.

First we specify vocabulary tables in a `config.yaml`; the listing 3 below denotes ‘countries‘ as a vocabulary table. All values in denoted vocabulary tables are copied to an auto-generated `.yaml` file. Listing 4 shows a snippet of data from the ‘countries‘ table which has been copied to a auto-generated `countries.yaml` file.

```yaml
tables:
    countries:
        vocabulary_table: true
```

Listing 3: A yaml section to demarcate table ’countries’ as a vocabulary table

```yaml
- country_destination: AU
  destination_km2: 7741220
  destination_language: eng
    :
- country_destination: CA
  destination_km2: 9984670
  destination_language: eng
  distance_km: 2828.1333
    :
```

Listing 4: Example of data rows copied from ‘countries‘ vocabulary table

The primary reason for copying vocabularies this way is to maximise transparency for auditing purposes. Data holders can audit each value extracted from the real dataset, before creating any synthetic data. Note that we

have to be careful in making sure that the tables marked as vocabulary tables truly do not hold privacy sensitive data, otherwise catastrophic privacy leaks are possible, where the original data is exposed raw and in full.

The downside of this approach is clear when scaling up to address vocabulary tables which are very large. Therefore our generator pipeline is modular to ensure that vocabularies need only be copied once when creating more rows to add into a synthetic dataset.

Generate Random Values that are Structurally Correct: The second auto-generated file, 'ssg.py', contains Python code that generates random values matching the data types defined by the Python classes. This human-readable Python code serves as part of the audit trail, demonstrating how values for populating each table column are generated. For complex schemas with multiple tables and columns, the generator code for each column is easily identifiable and can be customised independently of rest of the generator.

Listing 5 demonstrates the auto-generated Python code for generating 'id' and 'date_account_created' values for the 'User' table. 'id' is assigned generic, password-like values, and 'date_account_created' is assigned a random date value.

```
class usersGenerator:
    num_rows_per_pass = 1

    def __init__(self, src_db_conn, dst_db_conn):
        pass
        self.id = generic.person.password()
        self.date_account_created = generic.datetime.date()
        ...
```

Listing 5: A Python class for generating synthetic id and date_account_created values for Postgres table 'User'

Refine values using aggregate statistics: The default behaviour of SSG is to generate syntactically correct, random values. This section shows how we incorporate aggregate and statistical properties of real data in order to generate synthetic data that retain those properties.

We demonstrate an example to generate normally distributed synthetic values to populate a 'users.age' column, with reference to the mean and standard deviation values of the real data. The user begins by defining SQL statements in the 'age_stats' section of a 'config.yaml' file. This is demonstrated in listing 6. SSG uses the credentials provided to authenticate to the database and execute SQL statements to compute the required values. Computed values are recorded in an auto-generated `src-stats.yaml` file, demonstrated in listing 7. These can be can be referenced by the Python data generators. Listing 8 shows the Python provider function that generates a distribution of values to meet the statistical properties computed and recorded in 'config.yaml' and 'src-stats.yaml'.

```yaml
src-stats:
    - name: age_stats
    query: >
    SELECT AVG(age)::float AS mean, STDDEV(age)::float AS std_dev
    FROM users
    WHERE age <= 100
tables:
    users:
        row_generators:
          - name: airbnb_generators.user_age_provider
            kwargs:
              query_results: SRC_STATS["age_stats"]
            columns_assigned: age
```

Listing 6: A section of the config.yaml file that shows an SQL statement to compute mean and average of column ‘users.age‘. Results are stored as ‘age_stats‘.

```yaml
age_stats:
- mean: 36.54434029695572
  std_dev: 11.708339792587486
```

Listing 7: Example of mean and standard deviation values computed from ‘users.age‘ column

```python
  import random
  def user_age_provider(query_results):
      mean: float = query_results[0]["mean"]
      std_dev: float = query_results[0]["std_dev"]
      return random.gauss(mean, std_dev)
```

Listing 8: A provider function

The primary reason for extracting information using SQL statements and documenting it in ‘config.yaml‘ is to maximise transparency for auditing purposes. Similar to vocabularies, users can audit information that is disclosed about real data by reviewing the human-readable ‘config.yaml‘ and ‘src-stats.yaml‘ files. Multiple properties, such as marginals, percentiles, and skewness, can be used simultaneously to enhance the fidelity of synthetic data. These computations can be resource-intensive with large datasets. To address this, the SSG generator process is modularised: properties are computed and stored once, allowing subsequent generators to reference these values, which will be reliable provided the real dataset has not changed significantly.

Introduce differential privacy into aggregate statistics: Differential privacy is arguably the most popular technique for providing privacy guarantees on SDGs. Let us imagine two datasets:

- A synthetic dataset B generated with information of person X.

- A synthetic dataset A generated without information of person X.

If both datasets were generated using a differentially-private mechanism, performing a query on dataset A should provide the same, or almost the same, result as performing the same query on dataset B [19]. Differentially private mechanisms hide the presence or absence of person X —or one any individual— in the dataset, which implies strong protection of their privacy [21]. To accomplish this, these mechanisms inject random noise to the synthetic data. The amount of noise is a function of the privacy parameter epsilon ε that measures how similar the datasets A and B are required to be. ε needs to be chosen carefully to provide the required privacy guarantee.

One of the most common fundamental techniques for generating synthetic data in a differentially private involved 3 steps: 1) select, or choose, some queries over the original data, 2) measure, or execute, those queries using a differentially private mechanism, and 3) generate synthetic data using these measurements [20].

SQLSYNTHGEN enables the select and measure steps by supporting differentially private SQL queries in 'src-stats.yaml' (Listing 9).

```yaml
src-stats:
  - name: age_stats
    dp-query: >
      SELECT AVG(age) AS mean, STDDEV(age) AS std_dev
      FROM query_result
    epsilon: 0.5
    delta: 0.000001
    snsql-metadata:
      max_ids: 1
      id:
        type: string
        private_id: true
      age:
        type: float
        lower: 0
        upper: 100
```

Listing 9: A differentially-private SQL query.

Internally, SQLSYNTHGEN uses SMARTNOISE SQL [1] to execute differentially private queries. As seen in Listing 9, SMARTNOISE SQL needs additional information besides the SQL query for applying a differentially private mechanism, including the privacy parameter epsilon ε. Regarding the final generate step, the query results are made available to provider functions —demonstrated in Listing 8— so SQLSYNTHGEN users can use these measures for data generation.

5 Discussion

The proliferation of research on synthetic data over the past five years underscores its significance in addressing data scarcity and sensitivity issues in machine learning. With 25,600 papers published from 2023 to mid-2024 alone, these studies span diverse domains, including computer vision, natural language processing, and healthcare [11], primarily focusing on the generation, evaluation, and application of synthetic data, particularly using GANs [3]. Originally research-driven, these methods are now being translated into practical applications, revealing new challenges and considerations [18].

Our development of a Synthetic Data Generator (SDG) for sharing sensitive hospital information has highlighted these key challenges:

There is a lack of generators developed for relational data: The development of synthetic generators commonly explore image, text data, or tabular data. Our experience is that synthetic data generators overlook the relational data format, possibly because of the foreign key constraints satisfaction criteria. This is a problem because hospital datasets are often stored in relational formats.

There is a lack of explainability in privacy preserving mechanisms: Explainability in synthetic data generators is a crucial issue for custodians of sensitive data, especially in hospitals. The lack of explainability undermines discussions between hospital data stakeholders, including both staff and patients. One discussion impacted by the lack of explainability is that of maintaining a balance between privacy guarantees and the utility of the synthetic data. While ensuring that synthetic data generators do not leak sensitive information is essential, explaining the privacy preservation mechanisms involved can be complex. Furthermore, the processes used by generators based on GANs and deep neural networks are opaque, making it difficult to assure stakeholders of the synthetic data's reliability and safety. Finally, both generators and metrics (e.g., fidelity, diversity) used to evaluate the quality of synthetic data are not easily interpretable.

We specifically addressed this explainability challenge in a series of workshops with patient and public involvement, and using SSG as an exemplar. There were two key messages from our stakeholders. Firstly, they were reassured to understand the distinction in the source of the data. Anonymised data is processed from the original data whereas synthetic data is generated *de novo*. Secondly, they valued using a language that talked about sharing information (with synthetic data) in contrast to sharing data (with anonymisation). There was recognition that information is *already* shared and tools like SSG are trustworthy because they are transparent about what information is used to generate the synthetic data.

Despite its design to address these challenges, our SQLSYNTHGEN tool has several limitations:

Lack of Autonomous Model Discovery: Unlike GANs-based [3] or Bayesian-based [8] generators, SSG cannot autonomously discover underlying models or relationships. Users must predetermine the models, limiting the tool's adaptability and the transferability of algorithms trained on its outputs to real-world data.

Need to Ensure Security: The design of SSG includes copying vocabulary tables in their entirety and executing SQL statements on real data based on user configurations, makes it a powerful tool. However, these features introduce risks of user errors. Accidental copying tables with sensitive data could lead to severe data breeches. Executing SQL statements without proper access controls could damage real patient information.

Lack of Evaluation: SSG allows users to selectively disclose information used to shape synthetic data outputs but it lacks an integrated evaluation mechanism. Since each piece of information is independently disclosed, there is an opportunity here to iteratively fine-tune the balance between fidelity and privacy by combining SSG with an evaluation tool such as TAPAS [15].

6 Conclusion

The number of research papers on synthetic data has surged significantly, indicating its growing importance in addressing data scarcity and sensitivity issues in machine learning. There is a notable gap in the development of synthetic data generators specifically for relational data structures. Most exciting developments on generators

focus on time-series, graph, audio, imaging or tabular data structures, often neglecting the complexities associated with relational databases, such as foreign key constraints. This limitation is significant because many practical applications, particularly in healthcare, rely heavily on relational data formats.

Aside from the oversight in provision for relational data, the lack of explainability in privacy-preserving mechanisms is a critical challenge. For synthetic data to be trusted and widely adopted, especially in sensitive domains like healthcare, stakeholders need to understand how privacy is preserved. The opacity of deep learning models and GANs currently used in generating synthetic data makes it difficult to provide this assurance, which can hinder stakeholder discussions and acceptance.

The direction for future work on the application of synthetic data generation in sensitive data context is clear:

1. **Development of Relational Data Generators:** There is a clear need for synthetic data generators that can handle relational data formats effectively, addressing issues like foreign key constraints.
2. **Improving Explainability:** Enhancing the explainability of synthetic data generation processes will be crucial for gaining stakeholder trust and facilitating broader adoption by custodians of sensitive data.
3. **Integrated Evaluation Frameworks:** Combining synthetic data generators with comprehensive evaluation or attack frameworks can help explainability as well as ensuring an optimal balance between fidelity and privacy.

By addressing these challenges and focusing on these future directions, the practical application of synthetic data can be significantly enhanced, making it a more viable solution for real-world problems, particularly in sensitive domains such as healthcare.

7 Acknowledgements

1. This project was funded by Ecosystem Leadership Award under the EPSRC OobfJ22\100020.
2. This project was supported by the National Institute for Health and Care Research (NIHR)University College London Hospitals (UCLH) Biomedical Research Centre (BRC).
3. The authors thank Olajumoke Olatunji for her help in improving the documentation of SSG and helping us present it.

References

[1] Joshua Allen, Sarah Bird, and Kathleen Walker. Opendp platform for differential privacy, May 2020.

[2] Alhanoof Althnian, Duaa AlSaeed, Heyam Al-Baity, Amani Samha, Alanoud Bin Dris, Najla Alzakari, Afnan Abou Elwafa, and Heba Kurdi. Impact of dataset size on classification performance: An empirical evaluation in the medical domain. Applied Sciences, 11(2), 2021.

[3] Márcio Antunes and Ernesto Oliveira. Survey on synthetic data generation, evaluation methods and gans. Mathematics, 10(15):2733, 2022.

[4] M. Bayer. Sqlalchemy. `https://www.sqlalchemy.org/`.

[5] Emily E. Berkson, Jared D. VanCor, Steven Esposito, Gary Chern, and Mark D. Pritt. Synthetic data generation to mitigate the low/no-shot problem in machine learning. In 2019 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), pages 1–8. IEEE, 2019.

[6] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. IEEE Trans. Pattern Anal. Mach. Intell., 44(11):7327–7347, 2022.

[7] Airbnb New User Bookings. Airbnb new user bookings. Kaggle, https://www.kaggle.com/competitions/airbnb-recruiting-new-user-bookings, 2015. [Accessed: November, 25, 2015].

[8] Kuntai Cai, Xiaokui Xiao, and Graham Cormode. Privlava: Synthesizing relational data with foreign keys under differential privacy. Proceedings of the ACM on Management of Data, 1:1–25, 06 2023.

[9] FK Dankar and K El Emam. Practicing differential privacy in health care: A review. Transactions on Data Privacy, 6(1):35–67, 2013.

[10] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci., 9(3-4):211–407, 2014.

[11] Joao Fonseca and Fernando Bação. Tabular and latent space synthetic data generation: a literature review. Journal of Big Data, 10, 07 2023.

[12] C. Gavidia-Calderon, M. Hauru, I. Stenson, and M. Yong. sqlsynthgen. https://github.com/alan-turing-institute/sqlsynthgen, 2024.

[13] Matt Hancock. Data saves lives: reshaping health and social care with data, 2022. Accessed: 2024-06-04.

[14] S. Harris, T. Bonnici, T. Keen, W. Lilaonitkul, M. J. White, and N. Swanepoel. Clinical deployment environments: Five pillars of translational machine learning for health. Frontiers in Digital Health, 4:939292, Aug 2022.

[15] Florimond Houssiau, James Jordon, Samuel N. Cohen, Owen Daniel, Andrew Elliott, James Geddes, Callum Mole, Camila Rangel-Smith, and Lukasz Szpruch. Tapas: a toolbox for adversarial privacy auditing of synthetic data, 2022.

[16] LA Jones, JR Nelder, JM Fryer, PH Alsop, MR Geary, M Prince, and RN Cardinal. Public opinion on sharing data from health services for clinical and research purposes without explicit consent: an anonymous online survey in the uk. BMJ Open, 12(4):e057579, Apr 2022.

[17] James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N. Cohen, and Adrian Weller. Synthetic data - what, why and how? CoRR, abs/2205.03257, 2022.

[18] James Jordon, Lukasz Szpruch, Francois Houssiau, and Matteo Bottarelli. Synthetic data - what, why and how?, 2022.

[19] A Kopp. Microsoft smartnoise: Differential privacy machine learning case studies. Technical report, Microsoft, 2021. Accessed: 2024-06-08.

[20] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. AIM: an adaptive and iterative mechanism for differentially private synthetic data. Proc. VLDB Endow., 15(11):2599–2612, 2022.

[21] J. P. Near and C. Abuah. Programming Differential Privacy, volume 1. 2021.

[22] Beata Nowok, Gillian M. Raab, and Chris Dibben. synthpop: Bespoke creation of synthetic data in r. Journal of Statistical Software, 74(11):1–26, 2016.

[23] Cian O'Donovan, Sonya Coleman, Dermot Kerr, Christian Cole, Simon Li, David Sarmiento Perez, and Hari Sood. Trusted research environment users. November 2023.

[24] Office for National Statistics. Ons methodology working paper series number 16: Synthetic data pilot. Technical report, Office for National Statistics, 2021.

[25] Observational Medical Outcomes Partnership (OMOP). Omop common data model. https://www.ohdsi.org/data-standardization/the-common-data-model/, 2024.

[26] A. Patra, R. Batra, A. Chandrasekaran, and C. Kim. A multi-fidelity information-fusion approach to machine learn and predict polymer bandgap. Computational Materials Science, 172:109280, 2020.

[27] C. Santoni, D. Zhang, Z. Zhang, and D. Samaras. Toward ultra-efficient high fidelity predictions of wind turbine wakes: Augmenting the accuracy of engineering models via les-trained machine learning. arXiv preprint arXiv:2404.07938, 2024.

[28] G. Schomerus et al. The stigma of alcohol-related liver disease and its impact on healthcare. Journal of Hepatology, 77(2):516–524, Aug 2022.

[29] Synth Team. Synth: The open source declarative data generator, 2021.

[30] A. Tucker, Z. Wang, Y. Rotalinti, and P. Myles. Generating high-fidelity synthetic patient data for assessing machine learning healthcare software. npj Digital Medicine, 3, 2020.

[31] West JD Vasan K. The hidden influence of communities in collaborative funding of clinical science. R Soc Open Sci., 8(8), 2021.

[32] Jason Walonoski, Mark Kramer, James Nichols, Marc Galdzicki, Amelia Quina, Christopher Moesel, Darrell Hall, Tim Duffield, Matthew Gratch, Pascal Coorevits, David Sundwall, Emily Grant, Colin Jones, and Liza Tong. Synthea: Synthetic patient population simulator, 2020. Accessed: 2024-06-08.

[33] Rachel M. Werner and David A. Asch. The unintended consequences of publicly reporting quality information. Journal of the American Medical Association, 293(10):1239–1244, 2005.