# Guaranteeing Privacy through Differential Privacy for Healthcare Process Data

## Master's Thesis


Author : **Hannes Ueck**


Supervisor : Prof. Sander Leemans

Examiners : Prof. Sander Leemans
Prof. Wil van der Aalst


Registration date : 2024-01-09

Submission date : 2024-07-09


This work is submitted to the institute


**LuFG Business Process Management and Engineering, RWTH Aachen University**

# Abstract

Event logs capture the execution of processes, recording activities and additional information. A case represents a single instance of the process and includes a sequence of activity records and case attributes with additional information about the cases. Event logs may contain sensitive personal information that could harm an individual's privacy if published without pre-processing. Many privacy-preserving approaches to event log publishing ensure differential privacy. Differential privacy limits the disclosure of new information about any individual when publishing an event log beyond the publicly available background knowledge. Traditional methods focus on preserving the control flow but omit case attributes, limiting comprehensive process analysis based on these attributes. This thesis addresses this limitation by proposing a novel privacy-preserving event log publishing framework. Our approach ensures privacy for the control flow and case attributes, utilising synthetic tabular data generation approaches that guarantee differential privacy. The framework's simplicity allows for the use of various tabular data generation approaches based on the characteristics of the event log and available computational resources. Experimental results with real-world event data demonstrate the framework's feasibility and highlight the trade-off between data utility and the guaranteed levels of privacy.

# Contents

# Chapter 1

# Introduction

Modern information systems are designed to record activities executed in processes across different domains, such as production or healthcare [1]. This data is consolidated into event logs, which are collections of events, where each event represents an executed activity. A sequence of events that represent an independent execution of the process is referred to as a case. Additionally, additional information on the case level is stored in case attributes. Process mining aims to generate insights from event logs [1]. This often includes discovering a process's control flow, optimising it, detecting deviations from the specified behaviour, or predicting future events. Combined with case attributes, more detailed analyses are possible, e.g. predictive monitoring of processes [2] or identifying differing process behaviour for cohorts of cases [3].

In some instances each case of an event log represents an individual person, such as a patient in a healthcare process. Thus event logs contain personal data about the individuals involved in the process. Legislation enforces the protection of personal data, the General Data Protection Regulation (GDPR) in the European Union [4] or the Privacy Act in Australia [5]. Personal data is considered sensitive and thus needs additional protection when it reveals, e.g. health-related information, ethnic origin or political opinions of individuals [4]. In the context of event logs, this information can be contained in the activities, the time they are executed or the data about the individuals included in the additional attributes. For a hospital process, this might be a patient's treatment, the time when the

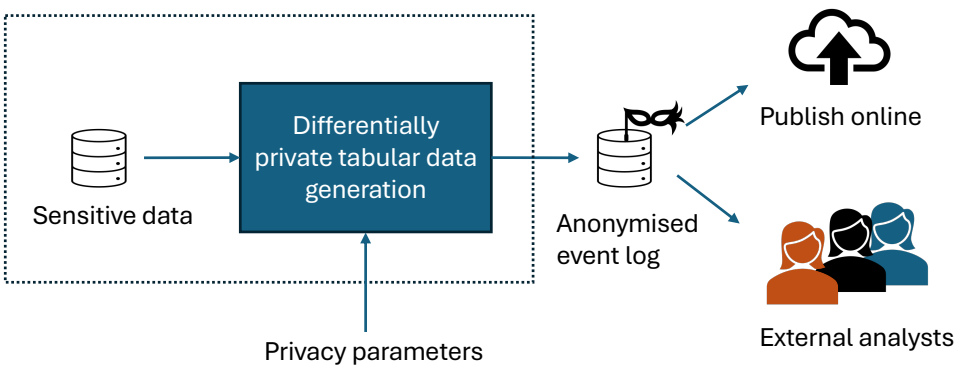


Figure 1.1: Conceptual overview of differentially private tabular data generation.

treatment is executed or the patient's treatment outcome. An adversary, i.e. someone who aims to extract sensitive data from a given event log, might be able to link an individual to a case and re-identify the individual [6]. To protect the privacy of the individuals, it is necessary to apply transformations to the data before publishing it [7, 8].

Privacy-preserving data publishing aims to publish data so that the privacy of the individuals is protected [9]. While preserving the privacy of the individuals, the data should still be useful for subsequent analysis. It has been shown that simple removal of names and unique identifiers from the data might lead to re-identification [10, 11]. Techniques where groups of similar records, i.e. equivalence classes, are created by generalising or suppressing data can be used to protect the privacy of the individuals [12]. However, the underlying assumption is that the adversary knows the victim's data is in the dataset, which can leak information about an individual [9]. A dataset generated using differential privacy ensures that the impact of any single individual on the outcome is strictly limited. Consequently, when such a dataset is published, it provides adversaries only with negligible additional information about any individual beyond what is known as general background knowledge [9]. Figure 1.1 shows the conceptual process of differentially private tabular data generation. Given a dataset and privacy parameters, which specify the required level of protection, a differentially private tabular data generation algorithm generates a synthetic dataset that can be, for example, published online or shared with analysts. At the same time, the synthetic data preserves sufficient utility of the underlying datasets for the analytical application of process mining techniques.

The importance of privacy in the field of process mining has been highlighted by multiple works [13, 1, 14, 15]. Many techniques, such as [16, 17, 18, 19], create equivalence classes of cases. Other approaches, such as [8, 20, 21], focus on providing DP for event logs to limit the impact of each individual on the event log. However, these approaches focus on the control flow and omit other attributes in the event log. The inclusion of additional attributes has only been addressed to a limited extent in [22]. This thesis aims at addressing the strong assumptions limiting the approach's applicability to real-life event logs.

In this work, we develop a method to guarantee privacy not only for the control flow but also for additional attributes present in the event log. We propose a framework that uses synthetic tabular data generation methods to anonymise event logs while ensuring differential privacy. The proposed framework is designed to be flexible and allows for the use of different tabular data generation algorithms based on the characteristics of the event log and available computational resources. We evaluate our framework on real-life event logs and assess the trade-off between data utility and the provided privacy guarantee.

## 1.1 Problem Statement

It is crucial to take measures to protect the privacy of individuals in event logs before sharing them with third parties or the public. Differential privacy is a strong privacy guarantee that limits each individual's impact on the shared data. Existing approaches that ensure differential privacy for event logs focus on the control flow and omit other case attributes present in the event log. This limitation restricts the types of analysis possible on the published event logs to control flow-focused techniques. In the domain of tabular data, synthetic data generation methods have been developed to provide differential privacy for tabular datasets by learning underlying noisy distributions in the data. These methods

have been shown to be effective in preserving the analytical utility of the data while ensuring privacy. However, to the best of our knowledge, these methods have not yet been applied to event logs. In this work, we investigate how tabular data generation methods can be applied to event logs to ensure differential privacy for the control flow and case attributes.

To ensure the analytical utility of event logs we focus on the preservation of several event log properties that largely determine the outcome of many process mining techniques. Hence, we require the generated event logs to be statistically similar to the original data in the following regards:

- The distribution of trace variants, which are the unique control flows observed for cases in the event log
- The mean, variance, and distribution of the case attributes
- The relations between the trace variants and each case attribute
- The relations between the case attributes

At the same time, we aim to provide a strong privacy guarantee, commonly referred to as $(\varepsilon, \delta)$-differential privacy in existing literature for the resulting event log.

## 1.2 Research Questions

Based on the problem statement, this thesis aims to develop a privacy providing framework for event logs by utilizing tabular data generation techniques. We derive the following research questions:

- How can tabular data generation methods be applied to event logs to ensure differential privacy for the control flow and case attributes?
- How does the choice of tabular data generation methods affect the utility of the anonymised event log?
- What is the trade-off between data utility and privacy associated with guaranteeing high levels of privacy?

## 1.3 Contributions

Answering the research questions, we provide three main contributions to the field of privacy-preserving event log publishing. First, we shed light on the connection between tabular data generation methods and event logs to ensure differential privacy for the control flow and case attributes. We discuss the privacy implications of applying these methods to event logs and propose steps to mitigate the arising privacy risks. Second, we develop a framework that applies tabular data generation methods to event logs while ensuring differential privacy. The framework allows for the use of different tabular data generation algorithms and can be extended with future improved algorithms as they are developed. Third, we evaluate how the choice of tabular data generation algorithm affects the utility of the anonymised event log and assess the trade-off between data utility and different privacy levels.

## 1.4 Structure of the Thesis

The remainder of this thesis is structured as follows. In Chapter 2, we introduce preliminary knowledge and notations for the techniques used within our framework. Additionally, measures from existing literature for the evaluation of the similarity between the original and anonymised event logs are presented. Chapter 3 reviews related work in the field of privacy-preserving process mining. Subsequently, in Chapter 4, we discuss the application of tabular data generation to event logs and present our proposed framework. Chapter 5 evaluates the framework on real-life event logs and assesses the trade-off between data utility and privacy. Next, in Chapter 6, we provide a detailed discussion of the evaluation results. Finally, Chapter 7 concludes the thesis and provides an outlook on future work.

# Chapter 2

# Background

This chapter provides the necessary preliminary information for the following chapters. We introduce the relevant concepts from the fields of process mining, machine learning, differential privacy and differentially private tabular data generation.

## 2.1 Process Mining

Process mining is a research area that uses event logs to derive insights about processes in organisations [1]. It aims to discover process models, check the conformance of logs to a model and enhance the processes by identifying bottlenecks or deviations from the expected behaviour. Therefore, it sits in between traditional data science and process science. Most process mining techniques utilise event logs as input, which we formalise in the following section.

### 2.1.1 Event Logs

An event log $L$ consists of traces, each a sequence of events. The events represent the execution of activities in a process. The traces represent the executions of cases in the process and each trace can have case attributes that provide additional information about the case. In this work, we consider only the control flow information and the case attributes.

**Definition 2.1** (Event Log)**.** An event log $L$ with case attributes is defined as a set of tuples, where each tuple consists of a trace and a tuple of case attributes: $L = \{(\sigma_1, \mathbf{ca_1}), (\sigma_2, \mathbf{ca_2}), \ldots, (\sigma_n, \mathbf{ca_n})\}$. Each trace $\sigma$ is a sequence of activities $\sigma_i = \langle a_1, a_2, \ldots, a_k \rangle$, where $a_j$ is the $j$-th event in the trace. The tuple of case attributes $\mathbf{ca_i} = (att_i^1, att_i^2, \ldots, att_i^p)$, where $p$ is the number of case attributes, provides additional information about the case.

Cases can be grouped based on their activity trace. We refer to each unique trace in a log as a trace variant.

**Definition 2.2** (Set of Trace Variants)**.** A trace variant is the set of all unique trace variants in an event log $TV(L) = \{\sigma_i | \sigma_i \in L\}$.

| Case id | Activity | Timestamp | Case outcome |
|---|---|---|---|
| 01 | Register | 2023-01-01 08:00 | - |
| 01 | First Examination | 2023-01-01 08:30 | - |
| 01 | Release | 2023-01-01 09:00 | No treatment |
| 02 | Register | 2023-01-01 08:15 | - |
| 02 | First Examination | 2023-01-01 08:45 | - |
| 02 | Release | 2023-01-01 08:30 | Ambulant treatment |
| 03 | Register | 2023-01-01 08:45 | - |
| 03 | First Examination | 2023-01-01 09:15 | - |
| 03 | CT Scan | 2023-01-01 09:45 | - |
| 03 | Release | 2023-01-01 10:15 | Clinical treatment |
| ... | ... | ... | ... |

Table 2.1: Example event log.

## 2.2 Artificial Neural Networks

In the field of Machine Learning, Artificial Neural Networks (ANN) are trained to approximate non-linear functions [23]. The primary challenge is to optimize the network's parameters to effectively model an unknown function using the given training data. Conceptually, an ANN is a parameterised function $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$, mapping high-dimensional real-valued vectors from an input dimension of size $n$ to an output dimension of size $m$. Within the function $f_\theta$, $\theta$ denotes the collection of the ANN's trainable parameters, termed weights.

At its core, an ANN consists of interconnected units called neurons, modelled after their biological counterparts. Each neuron receives a number of input signals, aggregates them in a weighted sum, and applies an activation function to produce the output signal [23]. There are various options for activation functions, which are typically non-linear [24].

Machine learning is a field of artificial intelligence that focuses on developing algorithms that can learn from and make predictions or decisions based on data. The GAN-based tabular data generation algorithms used in this work use supervised machine learning techniques to generate synthetic data. Among other techniques, we utilise a machine learning architecture, referred to as generative adversarial networks, which we formalise in the following sections.

### 2.2.1 Supervised Gradient-based Learning

In supervised learning, the term "supervised" refers to using known target values to guide the learning process. Specifically, it is assumed that there is access to some dataset $\mathbf{X}$ consisting of pairs $(x, t) \in \mathbb{R}^n \times \mathbb{R}^m$, where each pair represents a specific input $x$ and its corresponding known output $t$ for the ANN. Training refers to the automated process of successively adjusting the weights of the ANN to approximate a function that maps input to output. An objective function is used to quantify how well this task is achieved for the given training data For supervised learning, the objective function is the error between the known target value $t$ and the output $F_\theta(x)$. To calculate the error, a variety of loss functions $L(f_\theta(x), t)$ can be used. Their usage differs based on the overall problem and

**Algorithm 1: Stochastic Gradient Descent**

**Input:** Dataset $\mathbf{X}$
Neural network output function $f_\theta$
Loss function $L(f_\theta(x), t)$
Number of epochs $n$
Learning rate $\alpha$
**Output:** Optimized weights $\theta$

Initialise weights: $\theta$;
**for** *epoch in range*$(n)$ **do**
  **for** $(x_i, t_i) \in \mathbf{X}$ **do**
    `// Compute the model's output for input` $x_i$`:`
    $y_i \leftarrow f_\theta(x_i)$ ;
    `// Compute the gradient of the loss:`
    $g(x_i) \leftarrow \frac{\partial L(y_i, t_i)}{\partial \theta}$;
    `// Update weights:`
    $\theta \leftarrow \theta - \alpha g(x_i)$;

should be selected based on the use case [23].

Given the error of a particular output-target pair $(f_\theta(x), t)$, it is necessary to evaluate how changes in each connection weight would influence the network's overall performance. This is achieved by estimating the partial derivative of the utilised objective function with respect to each weight, given the network's current weights. The vector of partial derivatives is called the gradient of the objective function. The gradient descent algorithm iteratively updates each weight of the ANN in the negative direction of its partial derivative with respect to the loss. As a result, each iteration makes a locally optimal decision to improve the approximation of the target function.

Multiple variants of gradient descent have been proposed, differing in how they update the parameters. For example, stochastic gradient descent (SGD) updates the parameters using only a single training example at each iteration. Algorithm 1 shows the pseudocode implementation of stochastic gradient descent (SGD) [25]. SGD is a variant of the gradient descent algorithm that processes one training example at a time. The algorithm is executed for a number of epochs $n$, where one epoch is one pass through the entire dataset $\mathbf{X}$. In the first step, the neural network weights are initialised randomly. Then, in each epoch for each training example, the model's output is computed. With the computed output, the gradient of the loss function is calculated and then the gradient multiplied by the learning rate is subtracted from the current weights. The learning rate is a meta-parameter that controls how much influence each update has on the model's weights, balancing the trade-off between convergence speed and stability.

Other gradient descent variants differ in the way training examples are processed. Batch gradient descent, for example, processes all training examples at once, which leads to faster implementation due to vectorisation and is called batch gradient descent. In practice, mini-batch gradient descent is used, which combines the advantage of better convergence of the weights in SGD and faster implementation due to vectorisation in batch gradient descent. Mini-batch gradient descent divides the training examples into smaller batches [26].

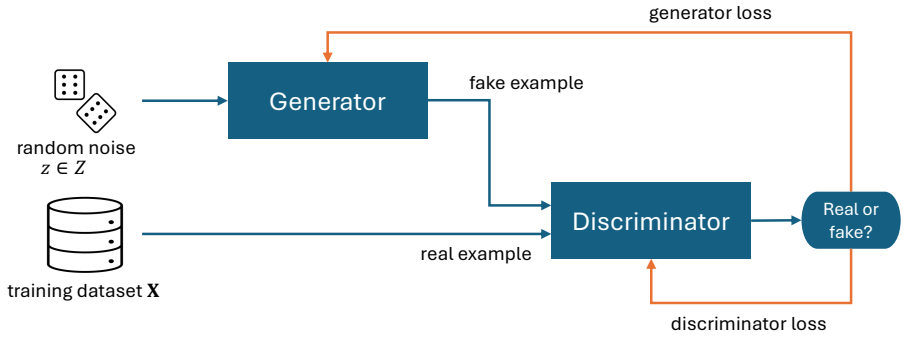


Figure 2.1: Architecture of a Generative Adversarial Network (GAN), illustrating the interaction between the Generator and Discriminator during the training process.

### 2.2.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are a class of neural networks intended for generating synthetic data [27]. Given a training set $\mathbf{X}$, the GAN tries to generate new data points that are as similar as possible to the original data. A GAN consists of two neural networks: a generator $Gen : \mathbb{Z}^m \to \mathbb{R}^n$ and a discriminator $Dis : \mathbb{R}^n \to \{0, 1\}$, coupled in a two-player zero-sum game. The generator is trained to generate data points similar to the training data from noise $z \in \mathbb{Z}^m$. Simultaneously, the discriminator is trained to distinguish between real and the generated data points from the generator. It thus outputs a binary value indicating whether the input is from $\mathbf{X}$ or generated by the generator. Figure 2.1 visualises the described architecture and shows the interactions between the components during training.

In the original paper, the generator and discriminator share the same objective function [27]. However, the generator tries to minimise it and the discriminator maximises it. While the generator increases its ability to generate more realistic data points, the discriminator improves distinguishing between real and generated data points. As such, both networks evolve inside a shared feedback loop, leading to enhanced results for both.

The objective function of the GAN can be adapted to the specific generation task to improve stability during training and the resulting output quality. Different similarity measures can be used in the objective function when measuring how close the generated data points are to the original input. One such measure with beneficial effects is the Wasserstein distance, which improves the training stability of the GAN [28]. The Wasserstein distance is a measure of similarity between two probability distributions and is also known as earth mover's distance as introduced in Section 2.6.1.

## 2.3 Graphical Models

An alternative approach to using GANs to estimate the properties of a dataset are graphical models. Graphical models are probabilistic models that represent dependencies between random variables using graph structures [23]. They are used to model complex distributions for applications such as speech recognition, bioinformatics or tabular data generation [29, 30, 31].

We briefly introduce relevant concepts and notations related to graphical models. Random

variables are quantities whose values are subject to variations due to chance. Probabilities measure the likelihood of these random variables taking on certain values Formally, the probability of a random variable $X$ taking a value $x$ is denoted by $P(X = x)$. The joint probability distribution $P(\mathbf{X})$ over a set of random variables $\mathbf{X} = X_1, X_2, \ldots, X_K$ specifies the probability of every possible combination of values for $\mathbf{X}$.

Conditional probability, denoted $P(X \mid Y)$, measures the probability of a random variable $X$ given that another random variable $Y$ has a specific value. If $P(X \mid Y) \neq P(X)$, then $X$ and $Y$ are dependent; otherwise, they are independent.

A graphical model $G = (V, E)$ consists of a set of vertices $V$ representing the variables and edges $E$ representing the conditional dependencies between these variables. Each node $X_k \in V$ corresponds to a random variable, and each edge $(X_k, X_l) \in E$ indicates a probabilistic relationship between the variables.

Given this notation, we know introduce directed graphical models, also called Bayesian networks. These networks use a directed graph with the edges representing causal relationships [23]. The joint distribution $P(\mathbf{X})$ according to the graph structure of a graph $G$ with $K$ nodes is given by:

$$P(\mathbf{X}) = \prod_{k=1}^{K} P(X_k \mid \mathrm{pa}_k), \tag{2.1}$$

where $\mathrm{pa}_k = \{X_l \in \mathbf{X} \mid (X_l, X_k) \in E\}$ denotes the set of parent nodes of $X_k$ in the graph.

Another type of graphical model is the undirected graphical model, also called Markov random fields. This graphical model uses undirected edges to represent symmetric relationships between variables [23]. The joint distribution $P(\mathbf{X})$ is factorised using the set of maximal cliques of the graph $G$. A clique is a subset of nodes in the graph that are fully connected. The potential function $\phi_c$ is defined for each clique $c \in C$ and assigns a positive value to each clique based on the configuration of variables within the clique. Further, let $X_c$ denote the set of vertices of the clique $c$. The equation for the joint distribution is then given by:

$$P(\mathbf{X}) = \frac{1}{Z} \prod_{c \in C} \phi_c(X_c), \tag{2.2}$$

where $Z$ is the normalisation constant, ensuring that the distribution $P(\mathbf{X})$ is correctly normalized.

## 2.4 Differential Privacy

Differential privacy (DP) is a probabilistic privacy guarantee given on the output of a data processing mechanism [32]. It states that anyone examining the output draws the same conclusions about an individual's private information regardless of whether that individual's data was part of the input to the mechanism or not. The goal of DP is to protect the privacy of individuals in a dataset while allowing for useful data analysis. This section introduces the definition and concept of DP, as well as theorems that describe how DP is maintained when combining techniques that provide this property. Note that DP is not a property of the data but of the mechanism that processes the data.

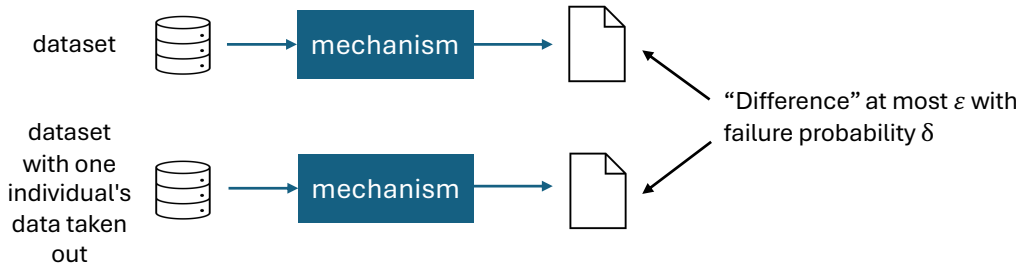


Figure 2.2: Conceptual idea of differential privacy.

The concept of DP is based on the idea of a mechanism $\mathcal{M}$ that processes a dataset $\mathbf{X}$ and produces an output $\mathcal{M}(\mathbf{X})$. Noise is added to the mechanisms output $\mathcal{M}(\mathbf{X})$ to ensure that the output does not reveal too much information about an individual's data. We follow the definition for $(\varepsilon, \delta)$-differential privacy of Dwork et al. [33].

**Definition 2.3** ($\varepsilon, \delta$-Differential Privacy)**.** Let $\mathbf{X_1}$ and $\mathbf{X_2}$ be two datasets differing in at most one data point. Further, let $\varepsilon > 0$ and $\delta \in [0, 1]$ be privacy parameters. Then a randomised mechanism $\mathcal{M}$ provides $(\varepsilon, \delta)$-differential privacy if for all subsets $S$ of the output space of $\mathcal{M}$,

$$\frac{\Pr[\mathcal{M}(\mathbf{X_1}) \in S]}{\Pr[\mathcal{M}(\mathbf{X_2}) \in S]} \leq e^{\varepsilon} + \delta\,, \tag{2.3}$$

where the probability Pr is taken over the randomness introduced by the mechanism $\mathcal{M}$.

While $\varepsilon$ controls the probability ratio, $\delta$ is a parameter that allows for a small probability that the mechanism fails to provide DP. The smaller $\varepsilon$ and $\delta$ are, the stronger the privacy guarantee of the mechanism is. Together, $\varepsilon$ and $\delta$ are referred to as the privacy budget of the mechanism.

$(\varepsilon, \delta)$-differential privacy is also called approximate differential privacy. Strict differential privacy, called $\varepsilon$-differential privacy, is a special case of approximate differential privacy with $\delta = 0$.

Figure 2.2 illustrates the concept of DP. When the output of a mechanism is $\varepsilon$-differentially private, the amount of information about an individual participating in the dataset that can be learned from the output of the mechanism is bounded by a privacy parameter $\varepsilon$. This means that the mechanism's output is almost the same whether the individual's data is included in the dataset or not. In practice, this means that a potential attacker can not use the dataset to gain insights about that individual. In the Definition 2.3, this concept is expressed by bounding the ratio of the probabilities that the mechanism produces a particular output, considering the mechanism's randomness, by a factor of $e^{\varepsilon}$. The additional factor $\delta$ describes the probability that this bound is violated.

### 2.4.1 Theorems of Differential Privacy

Combining multiple differentially private mechanisms requires careful consideration of the privacy budget of the individual mechanisms and how they are combined. The composition theorems of DP provide a framework for analysing the privacy budget of combined differentially private mechanisms [34]. In the following, we explain three of these theorems.

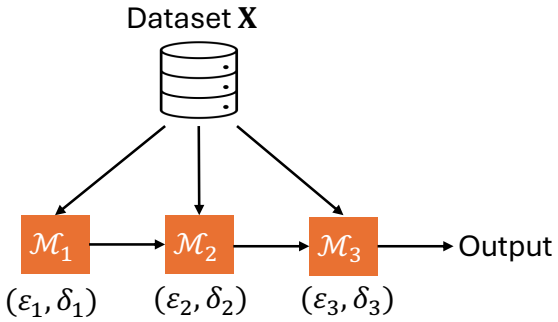


Figure 2.3: Concept of k-fold adaptive composition.

The sequential composition theorem can be applied when multiple differentially private mechanisms are sequentially run on the same dataset. Note that sequentially here refers to the dataset access, each mechanism depends only on the dataset and not on the output of the other mechanisms. The question then arises which privacy guarantees can be given for the union of all observed results. If these conditions are met, the overall privacy budget is the sum of each mechanism's individual privacy budget [34].

**Theorem 2.4** (Sequential Composition)**.** Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k$ be $(\varepsilon_i, \delta_i)$-differentially private mechanism for $i \in [k]$, for which the internal randomness for each mechanism is independent for each repeated access. Then, the combined mechanism $\mathcal{M}$ that runs each $\mathcal{M}_i$ in sequence on the same dataset provides $(\sum_{i=1}^{k} \varepsilon_i, \sum_{i=1}^{k} \delta_i)$-differential privacy.

The k-fold adaptive composition extends the sequential composition theorem by allowing the mechanisms to access the outputs of the previous mechanisms [34]. This allows for applying heterogeneous differentially private mechanisms on the same dataset, where each mechanism can access the previous outputs and the sensitive data. Figure 2.3 illustrates this flow of information when using k-fold adaptive composition.

**Theorem 2.5** (k-fold Adaptive Composition)**.** Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k$ be $(\varepsilon_i, \delta_i)$-differentially private mechanism for $i \in [k]$, $\varepsilon_i > 0$, $\delta_i \in [0, 1]$ and $\tilde{\delta} \in [0, 1]$. Then the combined mechanism using k-fold adaptive composition of $\mathcal{M}_i$ provides $(\tilde{\varepsilon}_{\tilde{\delta}}, 1 - (1 - \tilde{\delta}) \prod_{i=1}^{k} (1 - \delta_i))$-differential privacy with $\tilde{\varepsilon}_{\tilde{\delta}} =$

$$\min \Bigg\{ \sum_{i=1}^{k} \varepsilon_i, \\ \sum_{i=1}^{k} \frac{(e^{\varepsilon_i} - 1)\varepsilon_i}{e^{\varepsilon_i} + 1} + \sqrt{\sum_{i=1}^{k} 2\varepsilon_i^2 \log\left(e + \frac{\sqrt{\sum_{i=1}^{k} \varepsilon_i^2}}{\tilde{\delta}}\right)}, \\ \sum_{i=1}^{k} \frac{(e^{\varepsilon_i} - 1)\varepsilon_i}{e^{\varepsilon_i} + 1} + \sqrt{\sum_{i=1}^{k} 2\varepsilon_i^2 \log\left(\frac{1}{\tilde{\delta}}\right)} \Bigg\}. \tag{2.4}$$

The post-processing theorem states that any processing performed on the output of a differentially private mechanism does not degrade the privacy guarantee as long as the original data is not queried again [33].

**Theorem 2.6** (Post-Processing Theorem)**.** If $\mathcal{M}$ is an $(\varepsilon, \delta)$-differentially private mechanism, and $f$ is any function, then the mechanism $f(\mathcal{M}(\mathbf{X}))$ is also $(\varepsilon, \delta)$-differentially private.

### 2.4.2 Achieving Differential Privacy

Ensuring DP requires noise to be added to the output of the mechanism, i.e. changes that are not necessarily true to the original input [32]. This noise limits the influence a single datapoint can have on the output and thus guarantees DP. There are different basic mechanisms to achieve DP that differ in the noise they add and their applicability.

To achieve DP for a mechanism $M$, we need to go from a given required parameter $\epsilon, \delta$) to a specific level of noise. This noise is added to the output of a function $f$ that represents a query that operates on a dataset. The scale of the noise is determined by the sensitivity of the function $f$. The sensitivity describes the influence a single datapoint can have on the result of the function. Thus, the sensitivity of a function estimates the maximum difference in the output of the function when the input dataset differs in one record.

**Definition 2.7** ($\ell_1$-Sensitivity)**.** Let $f : \mathbb{R}^l \to \mathbb{R}^k$ be a function that takes a dataset with $l$ columns and returns a $k$-dimensional result vector. Then the sensitivity $\Delta_1 f$ of $f$ is defined as:

$$\Delta_1 f = \max_{\mathbf{X_1}, \mathbf{X_2}} \|f(\mathbf{X_1}) - f(\mathbf{X_2})\|_1 \,, \tag{2.5}$$

where $\mathbf{X_1}$ and $\mathbf{X_2}$ are datasets differing in at most one record.

**Definition 2.8** ($\ell_2$-Sensitivity)**.** Let $f : \mathbb{R}^l \to \mathbb{R}^k$ be a function that takes a dataset with $l$ columns and returns a $k$-dimensional result vector. Then the sensitivity $\Delta_2 f$ of $f$ is defined as:

$$\Delta_2 f = \max_{\mathbf{X_1}, \mathbf{X_2}} \|f(\mathbf{X_1}) - f(\mathbf{X_2})\|_2 \,, \tag{2.6}$$

where $\mathbf{X_1}$ and $\mathbf{X_2}$ are datasets differing in at most one record.

There are three key approaches to determining the necessary noise levels, all of which use the sensitivity and a privacy budget $(\epsilon, \delta)$. For example, the Laplace mechanism uses the $\ell_1$-sensitivity, while the Gaussian mechanism and differentially private stochastic gradient descent use the $\ell_2$-sensitivity. Below, we provide the definitions for these privacy mechanisms.

One of these approaches is the Laplace mechanism, which adds noise to the output of a function $f$ by drawing from a Laplace distribution [35]. The distribution is characterised by a scale parameter proportional to the $\ell_1$-sensitivity of the function and the desired privacy level ($\varepsilon$,0). Note that $\delta = 0$ for the Laplace mechanism.

**Definition 2.9** (Laplace Mechanism)**.** Let $f : \mathbb{R}^l \to \mathbb{R}^k$ be a function that takes a dataset $\mathbf{X}$ with $l$ columns and returns a $k$-dimensional result vector. Given the sensitivity of a function $\Delta_1 f$, the Laplace mechanism $\mathcal{L}$ satisfying ($\varepsilon$,0)-differential privacy is defined as:

$$\mathcal{L}(\mathbf{X}) = f(\mathbf{X}) + \eta \,, \tag{2.7}$$

where $\eta = (\eta_1, \eta_2, \ldots, \eta_k)$ and each $\eta_i$ is independently sampled from the laplace distribution $Lap\left(\frac{\Delta_1 f}{\varepsilon}\right)$.

Similar to the Laplace mechanism, the Gaussian mechanism is used to add noise to the output of a function $f$ to achieve $(\varepsilon, \delta)$-differential privacy [33]. The noise is sampled from a Gaussian (normal) distribution which is parameterised by a mean of 0 and a standard deviation proportional to the sensitivity of the function and the desired privacy levels $\varepsilon$ and $\delta$. Compared to the Laplace distribution, the Gaussian distribution has a lower peak and lighter tails, which leads to noise values that are smoothly concentrated around the mean.

**Definition 2.10** (Gaussian Mechanism)**.** Let $f : \mathbb{R}^l \to \mathbb{R}^k$ be a function that takes a dataset $\mathbf{X}$ with $l$ columns and returns a $k$-dimensional result vector. Given the sensitivity of a function $\Delta_2 f$, the Gaussian mechanism $\mathcal{G}$ is defined as:

$$\mathcal{G}(\mathbf{X}) = f(\mathbf{X}) + \eta\,, \tag{2.8}$$

where $\eta = (\eta_1, \eta_2, \ldots, \eta_k)$ and each $\eta_i$ is independently sampled from $\mathcal{N}\left(0, \sigma^2\right)$, with the standard deviation $\sigma$ defined as:

$$\sigma = \frac{\Delta_2 f \sqrt{2\ln(1.25/\delta)}}{\varepsilon}\,. \tag{2.9}$$

The Exponential mechanism is used to achieve DP in scenarios where the output is not necessarily numerical and noise addition is not suitable [33]. Instead, the Exponential mechanism selects an output from a defined set of possible outcomes based on a defined utility function, favouring outputs with higher utility values while still maintaining privacy.

**Definition 2.11** (Exponential Mechanism)**.** Let $\mathcal{R}$ be a set of possible outputs and $\mathcal{X}$ be a set of possible input datasets. Given a utility function $u(\mathbf{X}, r)$ that measures the quality of output $r \in \mathcal{R}$ for a given input dataset $\mathbf{X} \in \mathcal{X}$, the Exponential mechanism $\mathcal{M}$ selects an output $r$ with probability proportional to:

$$\Pr[\mathcal{M}(\mathbf{X}) = r] \propto \exp\left(\frac{\varepsilon u(\mathbf{X}, r)}{2\Delta u}\right)\,, \tag{2.10}$$

where $\Delta u$ is the sensitivity of the utility function, defined as:

$$\Delta u = \max_{r} \max_{\mathbf{X_1}, \mathbf{X_2}} \left|u(\mathbf{X_1}, r) - u(\mathbf{X_2}, r)\right|\,, \tag{2.11}$$

with $\mathbf{X_1}$ and $\mathbf{X_2} \in \mathcal{X}$ differing by at most one record.

### 2.4.3 Differentially Private Stochastic Gradient Descent

Differentially private stochastic gradient descent (DP-SGD) is a technique to train a machine learning model so that the model and the output it produces are differentially private [36]. In contrast to the privacy mechanisms presented in Section 2.4.2, the noise is not added to the output of a function but introduced during the training of the neural network.

DP-SGD is an extension of the SGD algorithm as introduced in Section 2.2.1. The idea is to limit the influence of the training examples on the weight updates made during SGD. This done by first clipping the gradients and then adding noise to the resulting values. Algorithm 2 outlines the adaptations that are made to the stochastic gradient descent

**Algorithm 2:** Differentially Private Stochastic Gradient Descent

**Input:** Dataset $\mathbf{X}$
Neural network output function $f_\theta$
Loss function $L(y, t)$
Number of epochs $n$
Learning rate $\alpha$
Clipping norm $C$
Noise level $\sigma$
**Output:** Optimized weights $\theta$

Initialise weights: $\theta$;
**for** *epoch in range*$(n)$ **do**
  **for** $(x_i, t_i) \in \mathbf{X}$ **do**
    `// Compute the model's output for input` $x_i$`:`
    $y_i \leftarrow f_\theta(x_i)$ ;
    `// Compute the gradient of the loss:`
    $g(x_i) \leftarrow \frac{\partial L(y_i, t_i)}{\partial \theta}$;
    `// Clip the gradients`
    $g(x_i)_{clipped} \leftarrow \frac{g(x_i)}{\max\left(1, \frac{\|g(x_i)\|_2}{C}\right)}$;
    `// Add noise`
    $g(x_i)_{noise} \leftarrow g(x_i)_{clipped} + \mathcal{N}(0, \sigma^2 C^2 \mathbb{1})$;
    `// Update weights:`
    $\theta \leftarrow \theta - \alpha g(x_i)_{noise}$;

in more detail. The clipping norm $C$ and the noise level $\sigma$ are hyperparameters, which influence the resulting privacy budget that is spent on the training.

One benefit of using DP-SGD is that the privacy guarantee is maintained not only for the produced output of the model, but also of the model itself [36]. Any information that can be learned from the model is bounded by the privacy level $\varepsilon$. This allows for the model to be shared so that others can generate predictions themselves.

## 2.5 Tabular Data Generation

This section introduces tabular data generation which makes use of the machine learning and graphical models introduced in Sections 2.2 and 2.3. Tabular data generation aims to generate synthetic data that closely resembles the statistical properties of the original data [37]. The goal is to provide data that can be used for analysis without revealing sensitive information about the individuals in the dataset. While the original data is not disclosed, the synthetic data should keep the statistical properties of the original data, such as the distribution of the data points or the correlations between the columns. Additionally, the mechanisms described in Section 2.4 can be used to provide formal guarantees for the privacy of the individuals whose data is published.

This work focuses on tabular data generation and its application to event logs. We formally define a tabular dataset as follows:

| Approach | Technique | Privacy mechanism |
|---|---|---|
| DP-GAN[38] | GAN | DP-SGD |
| PrivBayes[30] | Bayesian Network | Laplace Mechanism & Gaussian Mechanism |
| MST[39] | Undirected Graphical Model | Laplace Mechanism |

Table 2.2: Overview of tabular data generation approaches

**Definition 2.12** (Tabular Dataset)**.** A tabular dataset $\mathbf{X}$ is represented by a $n \times m$ matrix. The matrix consists of $n$ rows, which are called data points $\mathbf{x_i}$ with $i \in [i]$ and $m$ columns $\mathbf{c_j}, j \in [m]$, which are called attributes. The domain of an attribute can either be numerical or categorical and is denoted by $dom(\mathbf{c_j}), j \in [m]$.

There exist different methods, but generally the methods implement two basic steps. First, a model is fitted on the original data to learn a representation of the underlying joint probability distributions. Then, the synthetic data is sampled from the learned model. For methods relying on neural networks, privacy can be achieved using DP-SGD as discussed in Section 2.4.3. For methods based on graphical models, privacy can be achieved by introducing noise into the collected measures of the dataset by using the Laplace, Gaussian or Exponential Mechanism, as defined in Section 2.4.2.

**Definition 2.13** (Tabular Data Generation Algorithm)**.** Let $\mathcal{A}$ be a tabular data generation algorithm that takes a tabular dataset $\mathbf{X}$ as input. First, the algorithm returns a model $M_\theta = \mathcal{A}_{fit}(\mathbf{X})$ where $\theta$ is the set of parameters of the model. Then, the synthetic dataset $\mathbf{X^P} \sim M_\theta$ is sampled from the model $M_\theta$. Finally, we denote the entire process of generating the synthetic dataset $\mathbf{X^P}$ from the input dataset $\mathbf{X}$ using the algorithm $\mathcal{A}$ as:

$$\mathbf{X^P} = \mathcal{A}(\mathbf{X}), \tag{2.12}$$

where $\mathcal{A}(\mathbf{X})$ represents the combined process of fitting the model and sampling from it.

Two different foundational approaches to tabular data generation have been developed. While graphical model-based methods focus on structured approaches to modelling data distributions, generative machine learning-based methods use neural network architectures to generate synthetic data. Table 2.2 provides an overview of different tabular data generation approaches that are explained in detail in the next sections.

### 2.5.1 Methods based on Graphical Models

The tabular data generation methods based on graphical models offer a more structured approach to estimating the underlying data distribution and generating synthetic data. A brief introduction to graphical models was given in Section 2.3.

Privbayes is a method for tabular data generation that uses a Bayesian network to model the joint distribution of the data [30]. In Bayesian networks, a parent node is a node that has a directed edge pointing to another node, which models the direct dependencies between the attributes in the dataset. First, the algorithm spends half the privacy budget on constructing a Bayesian network over the attributes contained in the dataset to capture the dependencies in the dataset. The rest of the privacy budget is spent on calculating statistics for each attribute based on its parents in the network to estimate the parameters

of the Bayesian network. During the first step of the algorithm, the exponential mechanism is used to introduce noise into the dependencies. In the second step, the measured marginals are perturbed using the Laplace mechanism. This ensures $\varepsilon$-differential privacy for the resulting model and sampled dataset.

The Maximum Spanning Tree (MST) is a method for tabular data generation based on directed graphical models proposed by McKenna et al. [39]. The algorithm consists of three steps. First, a collection of marginals is selected from the possible marginals of the attributes. Then, the algorithm uses the Gaussian mechanism to obtain noisy estimates of the selected marginals. Lastly, a technique proposed by McKenna et al. [31] to construct a graphical model from those marginals is used. The graphical model is then used to sample the synthetic data.

### 2.5.2 Methods based on Generative Neural Networks

The methods based on generative neural networks use the advantages of neural networks to learn the underlying data distribution [37]. Compared to graphical models, GANs are more flexible in capturing the data distributions, but require more computational resources to train. A brief introduction to GANs can be found in Section 2.2.2.

Xie et al. [38] apply the GAN approach to electronic health records. They translate the information from each record into binary vectors which are then used as training examples. The model is trained to generate similar binary vectors, which can then be translated back to electronic health records. For better training stability, the Wasserstein distance is used in the loss function. DP-SGD, as discussed in Section 2.4.3, is used to ensure $(\varepsilon, \delta)$-DP for the model and thus the generated data. Note that only the discriminator needs to be trained via DP-SGD because the generator is not directly exposed to the training data.

## 2.6 Evaluation measures

After using the previously introduced methods to generate datasets in a privacy-preserving way, it is important that the resulting datasets are still useful for subsequent analysis. In the context of process mining the generated datasets are event logs. This section introduces measures to evaluate the similarity of the generated event logs to the original logs.

### 2.6.1 Earth Mover's Distance

The Earth Mover's Distance (EMD) is a measure to quantify the similarity between two frequency distributions [40]. It calculates the minimum amount of work required to transform one distribution into another. The work is defined as the amount of mass that needs to be moved multiplied by the distance it is moved.

**Definition 2.14** (Earth Mover's Distance (EMD))**.** Formally, given two distributions $P$ and $Q$ represented by histograms, EMD is calculated as:

$$\mathrm{EMD}(P, Q) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}} \tag{2.13}$$

where $f_{ij}$ is the flow of mass from bin $p_i$ to bin $q_j$, and $d_{ij}$ is the distance between bins. The flow $f_{ij}$ minimizes the total transportation cost subject to constraints.

Leemans et al. [41] extend the EMD to calculate similarity between trace variant distributions derived from event logs. The distance between two trace variants is calculated using the normalised string edit (Levenshtein) distance [42]. The resulting EMD quantifies how closely two trace variant distribution match, using a value $d \in [0, 1]$.

### 2.6.2 Conformance Checking: Precision and Fitness

Conformance checking is a technique to compare the behaviour recorded in an event log with the behaviour allowed by a process model. In the following, we explain fitness and precision and follow the definitions by Berti el al. [43] and Munoz et al. [44].

Fitness measures to which extent a process model is able to capture the behaviour of the event log without errors. A model that allows for all behaviour observed in the event log achieves a fitness value of 1. In the case that no trace from the event log can be executed in the process model the fitness value is 0.

In contrast, precision measures the conciseness of the model in regard to an event log on a scale of 0 to 1. A low precision score indicates that the model allows for more behaviour than observed in the event log. If the model only allows for the behaviour in the event log then the precision score is 1.

In our context, we use token-based replay to compare the behaviour of the original event log with the behaviour of an anonymised event log. Similarly to Rafiei et al. [45], we derive a petri net from the anonymised event log and replay the original event log on it. This allows measuring the quality of a process model derived from an anonymised event log for tasks like conformance checking.

### 2.6.3 Association Measures

The Pearson correlation coefficient measures the linear relationship between two datasets [46]. It is a value between -1 and 1, where 1 indicates a perfect positive linear relationship and -1 indicates a perfect negative linear relationship.

**Definition 2.15.** Let $\mathbf{x}$ and $\mathbf{y}$ be two vectors of length $n$. The Pearson correlation coefficient $r$ is defined as:

$$r = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - m_x)(\mathbf{y}_i - m_y)}{\sqrt{\sum_{i=1}^{n}(\mathbf{x}_i - m_x)^2}\sqrt{\sum_{i=1}^{n}(\mathbf{y}_i - m_y)^2}} \tag{2.14}$$

where $m_x$ is the mean of the vector $\mathbf{x}$, $m_y$ is the mean of the vector $\mathbf{y}$ and $\mathbf{x}_i$ is the $i$-th element of the vector $\mathbf{x}$.

The point-biserial correlation coefficient is used to measure the relationship between a binary variable and a continuous variable [47]. It uses a t-test with $n-1$ degrees of freedom and is equivalent to the Pearson correlation coefficient. The formula for calculating the point-biserial correlation coefficient is given by:

**Definition 2.16.** The point-biserial correlation coefficient $r_{pb}$ is defined as:

$$r_{pb} = \frac{\mu_1 - \mu_0}{s_y}\sqrt{\frac{N_0 N_1}{N(N-1)}} \tag{2.15}$$

where:

- $\mu_0$ is the mean of the metric observations coded 0
- $\mu_1$ is the mean of the metric observations coded 1
- $N_0$ is the number of observations coded 0
- $N_1$ is the number of observations coded 1
- $N$ is the total number of observations
- $s_y$ is the standard deviation of all the metric observations

Leemans et al. [48] introduce association metrics to measure the relationship between the control flow of a process and the case attributes in an event log. For numerical attributes, from the event log pairs of traces are sampled. For each of those pairs, trace distance and the difference in numerical value between two sampled traces from the event log are calculated. Given these measurements, standard correlation coefficients like the Pearson correlation coefficient are computed. For categorical attributes, they propose to measure the association using a proxy measure because categorical-numerical association measures are not available. The association is calculated using the correlation between the average trace variant distance of the sublogs defined by the categories of the case attributes and the average trace variant distance of the whole log.

### 2.6.4 Statistical Test: Log-Categorical Test

Leeman et al. [48] introduce a statistical test that evaluates whether sublogs defined by the categories of a categorical attribute are derived from the same process. The test uses samples of traces from the event log and measures if the average trace distance within the sublogs is larger than the average trace distance of the whole log. This is performed $n$ times while recording the number of times $e$, where the average trace distance within the sublogs is larger than the average trace distance of the whole log. If $1 - e/n$ is smaller than the significance level $\alpha$, the null hypothesis that the sublogs are derived from the same process is rejected.

# Chapter 3

# Related Work

This chapter provides an overview of the existing approaches to privacy-preserving event log publishing. We first discuss the current state of discussion on privacy in process mining. Then, we present the existing approaches to privacy-preserving event log publishing.

## 3.1 Privacy in Process Mining

Several works have considered privacy in process mining. Van der Aalst et al. [1] mention privacy issues in process mining in the context of cross-organisational process mining. Further, van der Aalst [14] introduces Responsible Process Mining by applying the FACT challenges (Fairness, Accountability, Confidentiality, Transparency) to process mining. The paper highlights the need for responsible process mining techniques that balance these aspects, favouring technological solutions over restrictive regulations. Mannhardt et al. [13] discuss privacy challenges for event logs from industrial processes that involve attributes of individual persons to which legal restrictions apply. They link the privacy challenges to the GDPR requirements and provide guidelines for designing systems incorporating process mining. More recently, Elkoumy et al. [15] provide an overview of privacy threats of process mining applications and present requirements for privacy-preserving process mining. They evaluate existing approaches based on these requirements and identify research challenges. The nature of all the approaches mentioned above is focused on discussing and quantifying the scope of privacy-related problems in process mining.

Burattin et al. [49] present an early approach to privacy-preserving process mining. They propose to encrypt the event log deterministically to anonymise the data. However, deterministic encryption assigns the same value to the same input and preserves frequencies and distances, which, as the authors point out, might lead to de-anonymisation. Rafiei et al. [50] propose a method to derive process models and social networks from encrypted event logs.

## 3.2 Group-based Privacy for Event Logs

One research direction is generalising or suppressing information in the event log to protect individuals' privacy. The underlying privacy guarantees are formalised through $k$-anonymity [12] and its extensions such as $l$-diversity [51] and $t$-closeness [52]. The under-

lying attacker model for these privacy guarantees is that the adversary already knows that the victim's data is contained in the dataset. The attacker's goal is to link an individual to a record in the dataset based on the uniquness of their data and background knowledge the attacker already knows.

Von Voigt et al. [6] highlight the risk of re-identification in event logs and introduce measures to express the uniqueness of an event log. They explore how a set of case attributes or event attributes can uniquely identify an individual in the event log. Rafiei et al. [17] define anonymisation operations based on group-based privacy for event logs and propose to record privacy metadata directly in the XES event log format. Similarly, Pika et al. [16] evaluate the impact of anonymisation operations on data utility and propose a method to record the privacy metadata in the XES event log format. In [19], Rafiei et al. propose the TLKC-privacy model, which is an extension of the LKC-privacy model that generalises group-based privacy guarantees. Their method uses limitations on the available background knowledge to improve the data utility of the anonymised event log. More recently, Fahrenkrog-Petersen et al. [18] present a prefix-based approach to anonymise the event log, guaranteeing k-anonymity and t-closeness while minimising the utility loss.

## 3.3 Differential privacy for event logs

Several approaches have been proposed to anonymise event logs while guaranteeing DP, as introduced in Section 2.4. In contrast to the group-based privacy guarantees provided by the approaches in the previous section, DP operates on the assumption that the attacker has no information on whether an individual's data is contained in the dataset or not. Therefore, the attacker aims to derive this information from the anonymised event log.

In the following sections, we present the existing approaches which focus on the control flow and omit other information in the event logs. Additional attributes are only included in one other approach presented in Section 3.3.5. In the last section, we provide an overview of the related work and position our work in regard to it.

### 3.3.1 Prefix-based Trace Variant Queries

Prefix-based trace variant queries are derived by querying a noisy prefix tree built from the event log [8, 21]. Due to the noise introduced in the tree, the resulting trace variants are differentially private. The noise might produce new trace variants that are not in the original event log.

Mannhard et al. [8] introduced the application of DP for event logs. They propose privacy-preserving queries for the frequencies of directly-follows relations and trace variants. The frequencies of directly-follows relations are directly obtained from the log and are privatised by adding noise using the Laplace mechanism as in Definition 2.9 For the frequencies of the trace variants, they propose building a prefix tree from the event log and then querying the tree for the frequencies of the trace variants of some specified maximum length. Equally, the frequencies are privatised by adding noise using the Laplace mechanism. They note that this method is only computationally feasible for a short maximum trace length or aggressive pruning of low-frequency prefixes. Additionally, higher maximum trace lengths lead to the introduction of traces that were not in the original event log.

Another approach to obtain trace-variant frequencies is Semantics-Aware Control-Flow Anonymization (*SaCoFa*) [21]. *SaCoFa* builds a prefix tree using a score function that evaluates each activity added to a branch in the tree to determine if adding it violates the predefined rules. The rules for the score function are based on the known process behaviour or domain knowledge derived from the original event log. To ensure DP, the result of the score function is privatised using the exponential mechanism, as described in Definition 2.11. After the tree is constructed, it is pruned to reduce the exponential growth of the tree with the prefix length. Finally, the resulting trace variant counts are derived from all paths in the tree that end in the trace end symbol. It should be noted that using knowledge derived from the original event log to construct the rules for the score function might degrade the privacy guarantee.

### 3.3.2 Graph-based Trace Variant Queries

Elkoumy et al. [53] propose an approach for releasing differentially private trace variant distributions using Deterministic Acyclic Finite State Automata (DAFSA). In this approach, the event log is represented as a DAFSA, where each sequence of events corresponds to a path from the initial state to the final state. A transition in the DAFSA represents the execution of an activity given a set of prefixes and postfixes for executed activities. To maintain privacy, a contingency table detailing the source and target states of each transition and the counts of such transitions in the event log is created. The Laplace mechanism is used to add noise to these transition frequencies in the contingency table. This ensures DP by altering the counts. The method oversamples the transitions to preserve the original trace variants and prevent the introduction or removal of new variants. All transitions of traces that pass through changed transitions are oversampled as well. This means duplicating entire traces as needed to maintain consistent counts. The relative time for every event is also made private using the Laplace mechanism, ensuring that the time information is also differentially private.

In *LIBRA* [54], the previous approach is extended using subsampling, which is a technique to lower the bounds of DP that can be achieved. The subsampling technique applies the privacy mechanism from the original approach to subsets of the event log sampled using Poisson subsampling [55]. The subsampled and privatised event logs are merged to obtain the final differentially private event log. This achieves a lower privacy budget than the original approach, allowing for a higher utility of the released event log. The authors include a filtering of infrequent trace variants to reduce the risks of re-identification. This filtering is based on thresholds for infrequent values presented by Chaudhuri et al. [56]. However, the privacy guarantee given by Chauduri et al. is $\varepsilon$-privacy, which includes the $\varepsilon$-value as a linear bound, instead of an exponential bound as in DP. Thus, using the thresholds without further privacy accounting might lead to a violation of $(\varepsilon, \delta)$-differential privacy.

### 3.3.3 GAN-based Trace Variant Generation

Rafiei et al. [57] propose a method to generate synthetic event logs using a generative adversarial network (GAN). They adopt an architecture used for tabular data generation and use it to release differentially private trace variant distributions. The GAN is trained to reproduce the trace variant distribution of the original event log using DP-SGD. This method suffers from the drawbacks discussed in Section 4.3. That is, the approach does

| Approach | Technique | Control flow | Event Timestamps | Case attributes |
|---|---|---|---|---|
| "Mine me" [53] | DAFSA | ✓ | ✓ | x |
| SaCoFa [21] | Prefix-tree | ✓ | x | x |
| Libra [54] | DAFSA | ✓ | ✓ | x |
| TraVaG [57] | GAN | ✓ | x | x |
| TraVaS [45] | Trace selection | ✓ | x | x |
| PRIPEL [22] | Prefix-tree | ✓ | ✓ | (✓) if independent |
| Our work | Trace selection & TDG | ✓ | x | ✓ |

Table 3.1: Existing approaches to privacy-preserving event log publishing

not remove or add trace variants, so including a single individual with a unique trace variant could lead to re-identification.

For completeness, we also mention the approach by Li et al. [58], who propose a GAN-based approach for generating synthetic trace variant distribution. However, they do not provide a privacy guarantee for the generated event logs.

### 3.3.4 Trace Variant Selection

A different approach to publishing differentially private trace variant distributions is presented by Rafiei et al. [45]. They adopt a method for $(\varepsilon, \delta)$-differentially private partition selection that releases category-aggregation pairs. In the context of event logs, the categories are the trace variants and the aggregations are the counts of the trace variants. The method adds noise directly to the trace variant counts and removes all trace variants where the count is below a certain threshold, where the added noise and the threshold are based on the privacy budget.

### 3.3.5 Trace Variant Queries with Contextual Information Matching

Fahrenkrog-Petersen et al. [22] present a two-step approach that first anonymises the control flow and then adds contextual information to the trace variants. That means the anonymised dataset preserves information about the control flow, timestamps and case attributes. The control flow is anonymised using the privacy-preserving trace variant queries proposed by Mannhardt et al. [8]. The obtained trace variants are matched back to the original cases from the event log based on the similarity of the traces. Next, noise is added to the timestamps of the original event log to ensure DP. Finally, the case attributes are anonymised separately using privacy mechanisms based on the data type.

Applying the privacy mechanisms in parallel to the same data point assumes the independence of the case attributes as well as the case attributes and timestamps. This limitation prevents the applicability of the approach to real-life event logs. Further, the authors do not discuss how the privacy guarantee of the two steps is composed or the possible degradation of the privacy guarantee as a result of matching the anonymised traces back to the original data is not discussed.

### 3.3.6 Overview of Differentially Private Event Log Publishing Approaches

Table 3.1 provides an overview of the existing privacy-preserving event log publishing approaches. As presented, the existing approaches to privacy-preserving event log publishing focus mainly on the control flow, as in [45, 57]. [53, 54, 22] also include timestamps in the anonymised event log. Only [22] investigated the inclusion of case attributes in the anonymised event log. However, their method assumes that the case attributes are independent, which does not hold for most common event logs. Furthermore, the privacy accounting for matching the anonymised traces back to the original event log is not discussed.

We identify two main groups of approaches to providing DP to the control flow. The first group uses prefix-based trace variant queries to release differentially private trace variant distributions [8, 21]. Here, all possible unique trace variants up to a certain length are considered and the frequencies of these trace variants are anonymised. This might introduce new trace variants that are not in the original event log. The second group privatises the trace variant distributions while only producing trace variants that were in the original event log [53, 54, 57, 45]. While *Libra* [54] and *TraVaS* [45] include filtering that removes infrequent trace variants, *"Mine me"* [53] and *TraVaG* [57] reproduce the exact trace variants with changed frequencies. In Section 4.3, we discuss the privacy implications of the latter approach.

The work in this thesis uses the differentially private trace variant selection presented in *TraVaS* [45]. This prevents generating new trace variants during anonymisation, as is the case for the prefix-based methods [8, 21]. While simultaneously avoiding information leakage by removing infrequent trace variants, as would be the case when using *"Mine me"* [53] or *TraVaG* [57]. We do not make use of Libra [54] because the filtering of infrequent trace variants is based on $\varepsilon$ privacy. In a second step, this work uses TDG as presented in Section 2.5 to anonymise the case attributes of the event log. This improves *PRIPEL* [22] by enabling the anonymisation of dependent case attributes.

# Chapter 4

# Applying tabular data generation methods to event logs

This chapter explains the design and reasoning behind the proposed framework to generate $(\varepsilon, \delta)$-differentially private event logs. We first link tabular data generation to differentially private event log generation. Next, we discuss implications for privacy and how to mitigate them. Lastly, we present the complete framework and explain each step in detail.

## 4.1 Overview of the Framework

We first give a brief overview of our proposed framework. The framework is designed to transform an event log into a differentially private synthetic event log while maintaining essential statistical properties. This transformation is achieved through several steps, as illustrated in Figure 4.1.

1. **Transform Event Log to Tabular Data**: The event log is converted into a tabular format, where each row represents a case and includes the trace variant and case attributes.
2. **Apply Differential Privacy to Trace Variants**: The trace variants are filtered to ensure that infrequent trace variants are removed. This is achieved using the

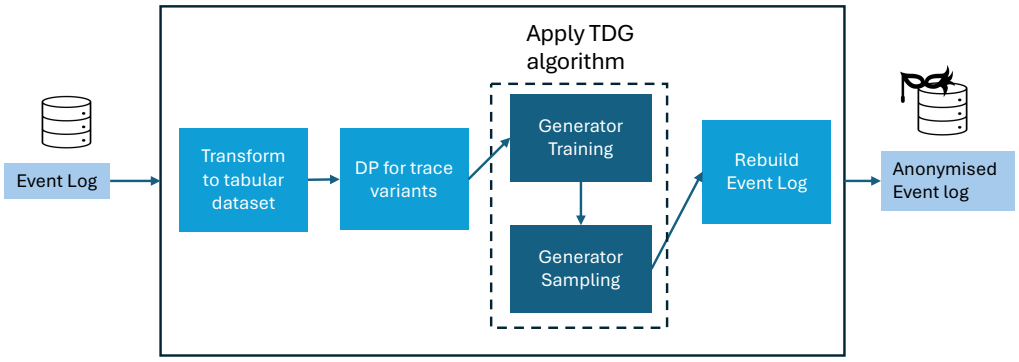


Figure 4.1: Architecture of the framework to generate differentially private event logs.

*TraVaS* [45] algorithm.

3. **Apply Tabular Data Generation Algorithm**: The tabular data with the set of differentially private trace variants and case attributes is processed further by a tabular data generation algorithm to produce a differentially private synthetic dataset.

4. **Rebuild Event Log**: The differentially private synthetic tabular data is transformed back into an event log format.

## 4.2 Linking Tabular Data Generation to Event Logs

Tabular data generation algorithms are used to provide $(\varepsilon, \delta)$-differential privacy for tabular data [59, 30]. The algorithms take a tabular dataset and privacy parameters as input. Event logs, as defined in Definition 2.1, consist of a collection of cases, each containing a sequence of events and case attributes. We propose aggregating the event log by case to create a tabular dataset. Each row in the tabular dataset represents a case and contains the trace variant and the case attributes. Table 4.1 shows an aggregated version of the event log from Table 2.1. The trace variant is represented by a categorical value and the case attributes are represented by numerical or categorical values.

**Definition 4.1** (Aggregated Event Log)**.** Let $L$ be an event log and $\mathbf{X}$ the tabular dataset constructed from $L$. For each tuple $(\sigma_i, \mathbf{ca_i}) \in L$, a datapoint $\mathbf{x_i} \in \mathbf{X}$ is created. A datapoint is the concatenation of the numerical representation of the trace variant $c_i^v = f(\sigma_i)$ with the case attributes in $\mathbf{ca_i}$. Thus, $\mathbf{x_i} = c_i^v \circ c_i^{att^1} \circ c_i^{att^2} \circ \cdots \circ c_i^{att^p}$.

| Case id | Variant | Case outcome |
|---|---|---|
| 01 | $v_1$ | No treatment |
| 02 | $v_2$ | Ambulant treatment |
| 03 | $v_3$ | Clinical treatment |
| 04 | $v_1$ | Clinical treatment |
| 05 | $v_3$ | No treatment |
| 06 | $v_1$ | No treatment |
| ... | ... | ... |

Table 4.1: Aggregated event log by case with variant names and case attribute "Case outcome".

An assumption that differentially private tabular data generation algorithms make is that the number of categories of an attribute in the dataset is much smaller than the number of individuals in the dataset. We argue that this assumption is violated for the column $c_v$ that contains the trace variants and provide detailed reasoning in Section 4.3. Usually and especially for unstructured processes, there are many different possible trace variants in an event log, possibly even infinite. Therefore, the trace variants need to be pre-processed before applying tabular data generation algorithms so that this assumption is not violated. We propose to use *TraVaS* [45] a technique to release $(\varepsilon, \delta)$-differentially private pairs of trace variants and their counts. We limit the trace variants to those produced by *TraVaS* to ensure that the assumption is not violated. Further, this gives the benefit of providing DP for the trace variants as well.

To combine the privacy budget spent on the *TraVaS* algorithm and the tabular data generation algorithm, we can use the k-fold adaptive composition theorem, as introduced in Section 2.5. With both techniques combined we provide $(\varepsilon, \delta)$-differential privacy for:

- The released categories of trace variants
- The distribution of the trace variants
- The distribution of the case attributes
- The dependencies between the trace variants and the case attributes
- The dependencies between the case attributes

## 4.3 Privacy Implications of Trace Variants for Private Tabular Data Generation

In traditional tabular data generation settings, it is assumed that the numbers of categories in the domains of the columns of the dataset $dom(\mathbf{c_j}), j \in [p]$ are much smaller than the numbers of individuals in the dataset. That means single individuals have a negligible impact on the domain of the columns $dom(\mathbf{c_j}), j \in [p]$. Given this assumption, tabular data generation algorithms produce private datasets where the domains of the columns are equal to those of the original dataset, i.e. $dom(\mathbf{c_j}) = dom(\mathbf{c_j^P}), j \in [p]$ [60]. When considering datasets derived from event logs, we argue that this assumption is violated for the column $c_v$ that contains the trace variants. Usually and especially for unstructured processes, there are many different possible trace variants in an event log. In the worst case, every case in the event log has a unique trace variant. Thus, every individual could be linked to a unique trace variant. If now the domain of the trace variants $dom(\mathbf{c_v^P})$ is derived from the input data, the algorithm would leak information about an individual as we will show in the following.

Consider a scenario where an attacker knows the trace variant related to an individual and knows that only this individual could produce this trace variant. Then there are two ways how the attacker could infer information:

(1) If the individual is in the event log, the anonymised event log would contain cases with such a trace variant. In that case, the attacker could infer that the individual is included in the event log.

(2) If the individual is not in the event log, the anonymised event log would not include cases with such a trace variant. In that case, the attacker could infer that the individual was not part of the event log.

This scenario illustrates that information about an individual could be leaked because the algorithm derives the domain of the trace variant column directly from the input, i.e. $dom(\mathbf{c_v^P}) = dom(\mathbf{c_v})$.

A naive approach to prevent this information leakage would be to filter out any unique trace variants. We argue that to fulfil the assumption that individuals have limited impact on the domains of the columns of the dataset $dom(\mathbf{c_j}), j \in [p]$, small groups of individuals linked to the same trace variant need to be removed. Similar to the previous scenario, the attacker can gain information based on the inclusion or exclusion of the trace variant in

the anonymised event log. Note that this information leakage occurs independently of the privacy budget $(\varepsilon, \delta)$. DP cannot be guaranteed because the assumption that the size of the attribute's domain is much smaller than the number of individuals in the dataset is violated.

In *TraVaS* [45], the authors apply a technique which only releases trace variants if the noisy count is above a certain threshold. We use this method to filter out infrequent trace variants before applying the tabular data generation algorithm.

Note that in Libra [54], a threshold to filter out infrequent trace variants is used. However, this threshold is based on a different privacy guarantee, namely $\varepsilon$-privacy [56]. This guarantee differs from DP by including $\varepsilon$ as a linear factor instead of an exponential factor. The authors do not discuss how this different privacy guarantee combines with the differentially private mechanisms of their approach in their work. Therefore, we are not able to compare the two approaches directly.

## 4.4 Framework

At the beginning of this chapter we gave a brief overview of our framework. This section explains our framework that aims to generate synthetic event logs that satisfy $(\varepsilon, \delta)$-differential privacy for the control flow and case attributes in detail.

The user provides input to the framework in the form of an event log $L$, specification of the case attributes types and two privacy budgets. While the first privacy budget $(\varepsilon_{\text{TraVaS}}, \delta_{\text{TraVaS}})$ is used to derive a set of differentially private trace variants, the second privacy budget $(\varepsilon_{\mathcal{A}}, \delta_{\mathcal{A}})$ is spent on the tabular data generation algorithm.

As outlined before, the main steps of the framework are the following:

1. Transform Event Log to Tabular Data
2. Filtering infrequent trace variants
3. Apply Tabular Data Generation Algorithm
4. Rebuild Event Log

While the generated event log does not contain any real data, it should resemble the original event log regarding statistical properties. Therefore, the generated event log should be statistically similar in the following dimensions:

- The distribution of trace variants
- The mean, variance, and distribution of the case attributes
- The relations between the trace variants and each case attribute
- The relations between the case attributes

The framework is designed to be flexible and allows for different tabular data generation algorithms based on the characteristics of the event log and available computational resources. This is indicated by the dashed box in Figure 4.1, which contains the steps performed by the tabular data generation algorithm. Each tabular data generation algorithm that is used should guarantee $(\varepsilon, \delta)$-differential privacy for the generated dataset and

should ingest and produce a tabular dataset as defined in Definition 2.12. The anonymised event log generated by the framework is $(\varepsilon, \delta)$-differentially private in the trace variants and case attributes and their dependencies.

In the following we explain each step of our framework in more detail.

### 4.4.1 Transform Event Log to Tabular Data

This step implements the transformation of the event log $L$ to a tabular dataset $\mathbf{X}$. According to Definition 4.1, the tabular dataset is constructed by aggregating the event log by case. We let the user choose which case attributes to include in the tabular dataset $\mathbf{X}$. Any directly identifying information, such as case IDs or patient IDs, is omitted during the aggregation process.

### 4.4.2 Apply Differential Privacy to Trace Variants

After constructing the tabular dataset $\mathbf{X}$ from the previous step, we want to limit the trace variants $c_v$ to a $(\varepsilon, \delta)$-differentially private selection of trace variants. This ensures that no information leakage occurs, as discussed in Section 4.3. We use the *TraVaS* algorithm proposed by Rafiei et al. [45] to obtain a set $\tilde{c_v}$ of $(\varepsilon, \delta)$-differentially private trace variants. By removing all rows where the trace variant is not in the set $\tilde{c_v}$ from the tabular dataset $\mathbf{X}$, we obtain a new tabular dataset $\tilde{\mathbf{X}}$.

### 4.4.3 Apply Tabular Data Generation Algorithm

This step takes the tabular data set $\tilde{\mathbf{X}}$, applies a tabular data generation algorithm $\mathcal{A}$ and returns the anonymised tabular dataset $\mathbf{X^P}$. As defined in Section 4.2, we denote the application of the tabular data generation algorithm by $\mathbf{X^P} = \mathcal{A}(\tilde{\mathbf{X}})$. After this step, $\mathbf{X^P}$ satisfies $(\varepsilon, \delta)$-differential privacy, given that $\mathcal{A}$ guarantees $(\varepsilon, \delta)$-differential privacy. Note that any tabular data generation method that guarantees $(\varepsilon, \delta)$-differential privacy can be interchangeably used. Some methods additionally require the user to specify types for the columns in the dataset. This is abstracted in the framework and must be specified once in the beginning for each case attribute in the event log.

### 4.4.4 Rebuild Event Log

The generated tabular data can be transformed back into an event log format by creating a case for each row in the generated tabular dataset $\mathbf{X^P}$. For each row, a case is created that is annotated with the information from the case attributes and corresponding events from the variant information. Based on the post-processing theorem of DP, the privacy guarantee of the anonymised dataset is preserved under any data transformation that does not involve additional queries to the original data. This means that any derived event log from $L_p$ remains differentially private. Thus, improving the data utility after generation, e.g. by removing impossible combinations of case attributes, does not violate the privacy guarantee.

## 4.5 Calculating the final Privacy Budget

The framework takes as input two privacy budgets $(\varepsilon_{\text{TraVaS}}, \delta_{\text{TraVaS}})$ and $(\varepsilon_{\mathcal{A}}, \delta_{\mathcal{A}})$. Both budgets are used to ensure $(\varepsilon, \delta)$-differential privacy for the final event log. The output from the first mechanism, *TraVaS*, is used as input for the tabular data generation algorithm. Therefore, instead of using the sequential composition theorem, the resulting composed privacy budget $(\varepsilon, \delta)$ can be calculated using the k-fold adaptive composition theorem as defined in Section 2.5.

# Chapter 5

# Evaluation

This chapter evaluates the proposed framework by applying it to three real-world event logs. The resulting anonymised event logs are examined regarding the similarity of the control flow, case attributes and the relationships within the event log to the original event logs. We compare the results of the tabular data generation algorithms within our framework in terms of data quality and runtime.

## 5.1 Implementation of the Framework

We implement the framework in Python [61], which is a high-level, general purpose programming language. We mainly used the following packages to implement the framework:

- **PM4PY** [62] for processing the event logs.
- **pandas** [63] for further processing of the event logs and data manipulation.
- **Synthcity** [64] for the implementation of tabular data generation algorithms
- **smartnoise-sdk**[1] for the implementation of tabular data generation algorithms
- **TraVaS**[2] for differentially private trace variant selection

The framework consists of the pre-processing of the event log, the application of the tabular data generation algorithms and the transformation back to an event log, as detailed in Chapter 4. The user can choose the tabular data generation algorithm, the privacy budget and the event log to be anonymised. Further, the user specifies the type of the attributes present in the event log. That is if the attribute is discrete, continuous, binary or categorical. This is necessary because some algorithms implement special pre-processing for some types of attributes.

The framework is designed to be flexible in the tabular data generation algorithm, which is ensured by using a common interface for the algorithms. For each algorithm, a wrapper is provided for the initialisation of the algorithm and the fitting of the algorithm to the data. During initialisation, the privacy budgets are set and the types of attributes are

[1] https://github.com/opendp/smartnoise-sdk
[2] https://github.com/wangelik/TraVaS

| Event Log | Number of cases | Percentage of unique trace variants | Case attribute types |
|---|---|---|---|
| Sepsis[65] | 1050 | 80.57% | binary, categorical |
| BPIC13[66] | 7554 | 20.00% | categorical |
| Traffic Fine[67] | 150370 | 0.15% | discrete, continuous, categorical |

Table 5.1: Characteristics of the event logs.

passed to the algorithm. The fitting of the algorithm takes the original tabular dataset and the number of samples to produce, which we set to the number of cases in the original event log. Then the algorithm is fitted on the data and the synthetic data is generated. Lastly, the synthetic data is transformed back to an event log. This procedure implements the steps of our framework as explained in Section 4.4.

## 5.2 Evaluation Setup

We evaluate the framework on three real-life event logs using three different algorithms and five different privacy budgets. Due to the non-deterministic behaviour of the tabular data generation algorithms, we run each algorithm on each event log for each privacy budget five times. Limited computational resources prevent running the experiments more frequently. The extended training times of *DPGAN* result in the exclusion of the Traffic Fine event log for this particular algorithm.

**Choice of Event Logs** An overview of the chosen event logs is provided in Table 5.1. The chosen logs vary in the number of cases, the amount of uniqueness in the trace variants and the types of case attributes.

In the following we give a brief overview over the most important log properties of the chosen event logs. The Sepsis event log [65] comprises 1050 Sepsis cases from a hospital with 16 different activities. The traces in the event log exhibit a high unique trace variants rate of 80.57%. That means that around 80% of the trace variant are contained in the event log only once. The case attributes of the Sepsis event log are mainly binary with the exception of the age attribute, which in this case is categorical due to binning. In comparison, the BPI Challenge 2013 (BPIC13) event log [66], which includes 7554 cases of an incident response management process from a car manufacturer, shows a unique trace variant rate of 20.00%. The case attributes in this event log are all categorical and related to the product or organisation involved in the case. The Traffic Fine event log [67] contains 150370 cases extracted from an information system managing road Traffic Fines with a share of unique variants of only 0.15%. The event log contains numerical attributes which relate to the fine for the traffic violation.

**Choice of Tabular Data Generation Algorithms** We choose three different implementations of tabular data generation algorithms for the evaluation. The chosen algorithms are explained in Section 2.5 The selection is based on the availability of each algorithms implementation and the results in existing literature. Tao et al. [59] compare differentially private synthetic data generation algorithms. They find that *MST* and *PrivBayes*

| $\varepsilon_{TDG}$ | $\varepsilon_{TraVaS}$ | $\varepsilon$ | $\delta$ |
|---|---|---|---|
| 0.1 | 1 | 1.1 | 0.75 |
| 1 | 1 | 2 | 0.75 |
| 1.5 | 1 | 2.5 | 0.75 |
| 3 | 1 | 4 | 0.75 |
| 10 | 1 | 11 | 0.75 |

Table 5.2: Privacy budgets when combining *TraVaS* and TDG algorithms with $\delta_{TDG} = 0.5$, $\delta_{TraVaS} = 0.5$ and $\tilde{\delta} = 0.001$ using the k-fold adaptive composition theorem.

perform consistently well in regard to the quality of the generated data. That means that the distributions of attributes are preserved and correlations between attributes are maintained. Due to those reasons, we choose *PrivBayes* [30], *MST* [31] and *DPGAN* [38]. We include *DPGAN* as it is an approach that uses a generative adversarial network to generate synthetic data, instead of using a graphical model. We refer to Section 2.5 for an overview of the algorithms. For *PrivBayes* and *DPGAN* we use the implementation in the framework synthcity. *MST* implemented in the framework smartnoise-sdk. As was mentioned earlier, the framework is not limited to the use of these three algorithms and other or new algorithms can be used with little additional effort.

**Choice of Privacy Budgets** We evaluate the framework for different privacy budgets. The privacy budget for the tabular data generation algorithms is chosen from the parameter range $\varepsilon_{TDG} \in \{0.5, 1, 1.5, 3, 10\}$. For TraVaS, we fix the privacy budget to $\varepsilon_{TraVaS} = 1$ and $\delta = 0.5$. The final privacy budget for the framework is calculated using the k-fold adaptive composition theorem of DP as defined in Definition 2.5. We choose $\tilde{\delta} = 0.001$ for the caluculation of the final privacy budget. Table 5.2 shows how the separate privacy budgets for TraVaS and our framework combine. This results in privacy budgets for the combined algorithms in the privacy parameter range of $\varepsilon \in \{1.1, 2, 2.5, 4, 11\}$ with $\delta = 0.61$.

## 5.3 Evaluating the Event Log Similarity

Our method aims to preserve the following aspects of the event log:

- The distribution of trace variants
- The mean, variance, and distribution of the case attributes
- The relations between the trace variants and each case attribute
- The relations between the case attributes

In the following sections, we examine for each of these properties how well they are preserved after applying the framework. The used evaluation measures are introduced in Section 2.6. We note that these measures can only be a proxy to estimate the data utility for process mining and analysis tasks. The assumption is that an anonymised event log that is statistically similar to the original event log produces similar analysis results.

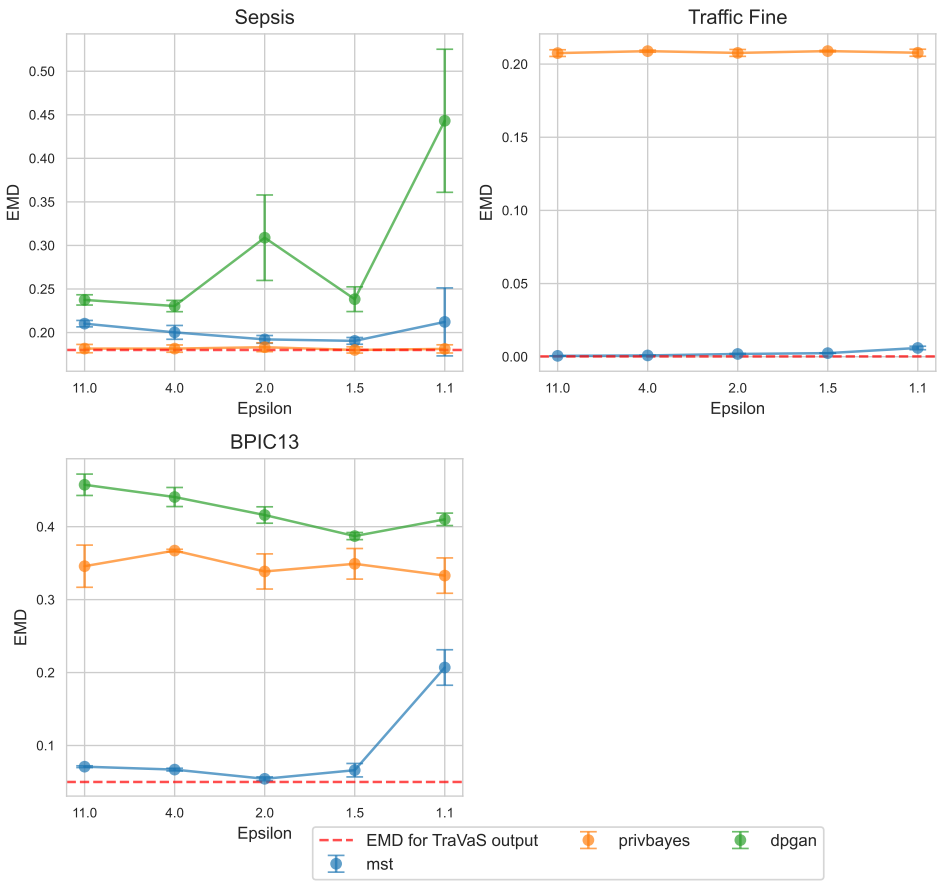


Figure 5.1: Comparison of Earth Mover's Distance (EMD) for different values of $\varepsilon$ across different event logs and algorithms.

### 5.3.1 Distribution of Trace Variants

**Earth Mover's Distance** The first aspect we evaluate the similarity of the trace variant distributions between the original and the anonymised event logs. This serves as a proxy to evaluating the data utility of the anonymised event logs regarding the trace variants and their distribution. As a measure of similarity, we use the Earth Mover's Distance (EMD) for event logs introduced in Section 2.6.1.

Figure 5.1 shows the Earth Mover's Distance (EMD) for different values of $\varepsilon$ across different event logs and algorithms. Low EMD scores translate to high similarity of the trace variant distributions. The dashed red lines present the scores obtained when computing the EMD between the trace variant distribution output of *TraVaS* and the original event log. As described in Section 4.4.2, we limit the trace variants in the input event log for the TDG algorithm to the set of trace variants contained in the trace variant distribution produced by *TraVaS*. Thus, this is sort of a baseline for the results of our framework.

As can be seen in the figure, for all three event logs and for the *MST* algorithm, the EMD tends to increase with lower values of $\varepsilon$. Especially for the BPIC13 event log the EMD

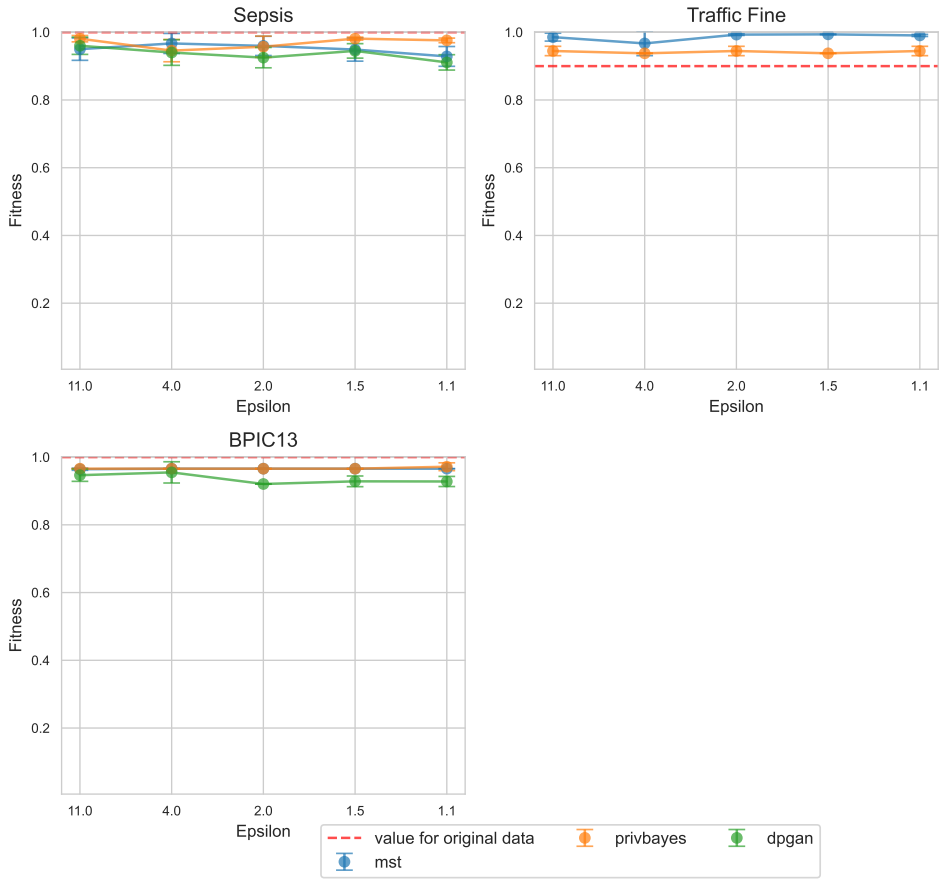


Figure 5.2: Comparison of fitness scores for different values of $\varepsilon$ across different event logs and algorithms.

increases sharply for $\varepsilon = 1.1$. The *PrivBayes* algorithm shows the best results in the Sepsis event log, but is signficantly worse than *MST* for the Traffic Fine event log at scores of 0.2 compared to 0.01 for *MST*. In general, *MST* produces the most consistent results while being on a similar value to the EMD scores of the output of *TraVaS*.

**Conformance checking** Additionally, we assess how well the anonymised event logs can be used for process discovery and conformance checking tasks. We discover a process model from the anonymised event log using the inductive miner infrequent [68] with a noise threshold of 20%. Then we replay the original event log on this discovered process model and compute the fitness and precision scores as introduced in Section 2.6.2. A higher fitness score indicates that more traces from the original event log can be replayed on the private process model. For precision scores, a higher score indicates that the private process model does not allow for more behaviour than the original process model.

In Figure 5.2 we compare the fitness scores for different values of $\varepsilon$ across the different event logs and algorithms. Similarly, Figure 5.3 shows the precision scores. The fitness score for the Sepsis event log when replaying the original event log on the original process model

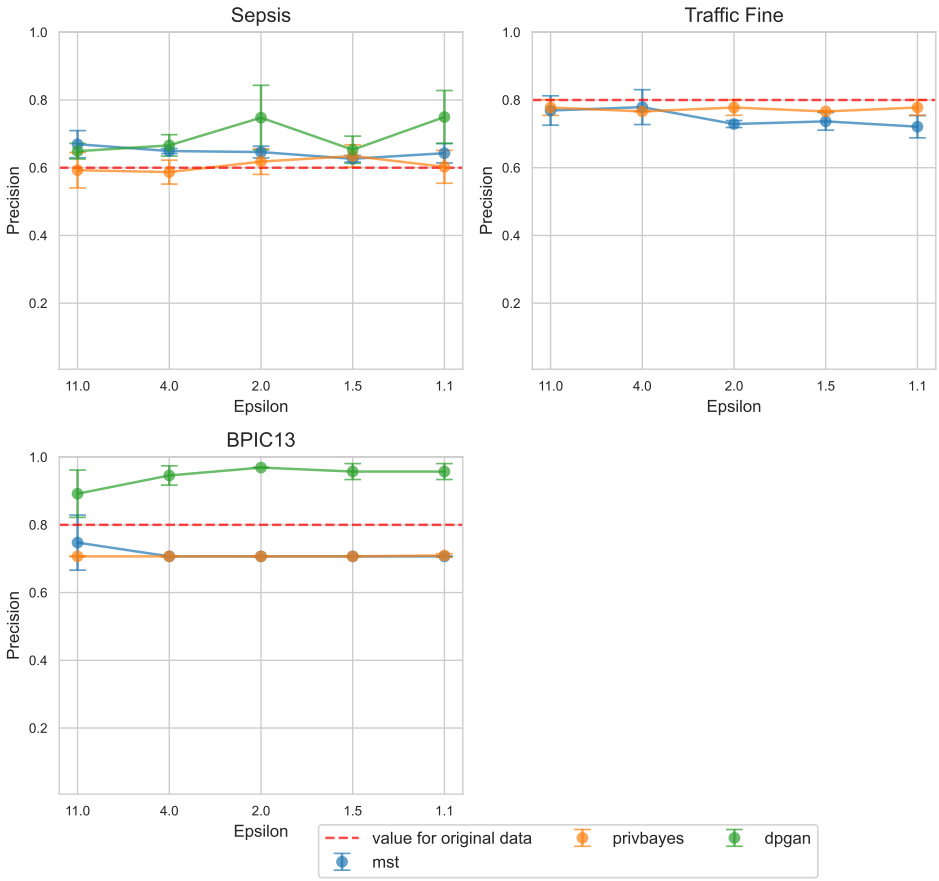


Figure 5.3: Comparison of precision scores for different values of $\varepsilon$ across different event logs and algorithms.

is 1. When replaying the original event logs on the private process models, the fitness scores decrease to a range between 0.91 and 0.98. All three models achieve similar fitness scores and the fitness score stays above 0.91 for all privacy budgets. This indicates that the models discovered from the anonymised event logs still capture most of the behaviour from the original event log. When looking at the precision scores for the Sepsis event logs, the precision scores derived from the private process models are higher than the precision scores of its original counterpart. This is in line with the decreased fitness scores and indicates that the private process models are more restrictive. Sepsis has a high number of infrequent trace variants, which are removed by limiting the generated trace variants to the differentially private selection of trace variants produced by *TraVaS*.

For the BPIC13 event log, we observe an inverse trend. The fitness scores for the private process models are higher compared to their original counterparts ranging. The non-private process model achieves a fitness score of 0.9, while private models derived from event logs generated by *MST* produces higher fitness scores in the range of 0.96 and 0.99. *PrivBayes* stays in the lower range of 0.93 to 0.95. The non-private process model achieves a precision score of 0.8. The precision scores of the private process models are lower than

but still in the range of 0.72 to 0.78. We see that for this event log the private process models are more concise than the original process model. The Traffic Fine event log has a very low trace uniqueness rate, thus almost all behaviour from the original event log is captured by the private process models.

Lastly, for BPIC13 the fitness score for the non-private process model is at 1.0 and the precision score at 0.8. The private process models achieve fitness scores of at most 0.97 and precision scores of at most 0.75. The only exception is the *DPGAN* algorithm, which achieves precision scores of up to 0.95. BPIC13 has a lower uniqueness rate than the Sepsis event log, but higher than the Traffic Fine event log. Lower fitness scores and lower precision scores might be case by the combination of removing infrequent trace variants and changed frequencies of the trace variants.

In conclusion, we observe that *MST* generally produces the most consistent results across the different event logs, maintaining EMD scores close to the baseline provided by *TraVaS*. The *PrivBayes* algorithm performs well for the Sepsis event log but falls short for the Traffic Fine event log. In terms of conformance checking, all algorithms maintain high fitness scores, indicating that the anonymised event logs capture most of the behaviour contained in the original event log. The precision scores of the private models show more variability, while still being close to the scores achieved for the non-private model.

### 5.3.2 Mean, Variance, and Distribution of the Case Attributes

**Mean and Standard Deviation** Table 5.3 shows the mean and standard deviation values of the case attributes for the Sepsis event log for all selected models and privacy budgets. As can be seen in the table, the mean values of the case attributes in the anonymised event logs are preserved better for higher values of $\varepsilon$. For the *MST* algorithm, the mean values of the case attributes are similar to the original event log for $\varepsilon = 10.5$. With decreasing values of $\varepsilon$, the mean values of the case attributes deviate more from the original event log. For $\varepsilon = 0.6$ the mean values start to differ significantly, which can be attributed to the high levels of noise introduced in the generation algorithm. The *PrivBayes* algorithm shows a similar trend, but the mean values of the case attributes have generally higher deviations from the original mean values than for the *MST* algorithm. Especially for the binary attributes `hypotensie` and `oligurie`, the mean values are significantly different from the original event log. *DPGAN* shows for almost all values the highest deviations from the original mean values for all privacy budgets.

For the standard deviation of the case attributes, the differences between the algorithms are smaller. The scale of the standard deviation is also preserved better for higher values of $\varepsilon$. Generally, the degradation of the standard deviation is less severe than for the mean values.

The mean values of the case attributes for the Traffic Fine event log are shown in Table 5.4. Generally, the results look worse than for the Sepsis event log. This time, *PrivBayes* shows the better results compared to *MST*. There is not a clear trend regarding the influence of $\varepsilon$ on the means' and standard deviations' preservation. The worse results for the Traffic Fine event log might be due to the generally higher standard deviations of the case attributes compared to the Sepsis event log.

| **MST** | | | | | | |
|---|---|---|---|---|---|---|
| $\varepsilon$ | $\mu_{age}$ | $\mu_{infection\ suspected}$ | $\mu_{hypotensie}$ | $\mu_{infusion}$ | $\mu_{oligurie}$ | $\mu_{hypoxie}$ |
| orig | 70.08 ± (17.36) | 0.81 ± (0.39) | 0.05 ± (0.22) | 0.76 ± (0.43) | 0.02 ± (0.15) | 0.02 ± (0.14) |
| 10.0 | 66.77 ± (19.34) | 0.67 ± (0.47) | 0.03 ± (0.18) | 0.62 ± (0.49) | 0.02 ± (0.14) | 0.02 ± (0.14) |
| 3.0 | 65.25 ± (20.26) | 0.67 ± (0.47) | 0.03 ± (0.18) | 0.61 ± (0.49) | 0.04 ± (0.19) | 0.02 ± (0.15) |
| 1.0 | 55.64 ± (21.86) | 0.67 ± (0.47) | 0.05 ± (0.21) | 0.62 ± (0.48) | 0.04 ± (0.19) | 0.04 ± (0.17) |
| 0.5 | 55.66 ± (21.63) | 0.70 ± (0.45) | 0.05 ± (0.19) | 0.62 ± (0.48) | 0.05 ± (0.15) | 0.02 ± (0.08) |
| 0.1 | 55.27 ± (21.55) | 0.51 ± (0.50) | 0.50 ± (0.50) | 0.50 ± (0.50) | 0.51 ± (0.50) | 0.40 ± (0.40) |
| **PrivBayes** | | | | | | |
| $\varepsilon$ | $\mu_{age}$ | $\mu_{infection\ suspected}$ | $\mu_{hypotensie}$ | $\mu_{infusion}$ | $\mu_{oligurie}$ | $\mu_{hypoxie}$ |
| orig | 70.08 ± (17.36) | 0.81 ± (0.39) | 0.05 ± (0.22) | 0.76 ± (0.43) | 0.02 ± (0.15) | 0.02 ± (0.14) |
| 10.0 | 63.44 ± (20.03) | 0.64 ± (0.48) | 0.28 ± (0.42) | 0.59 ± (0.49) | 0.25 ± (0.39) | 0.04 ± (0.17) |
| 3.0 | 61.86 ± (20.68) | 0.65 ± (0.47) | 0.12 ± (0.27) | 0.59 ± (0.49) | 0.09 ± (0.24) | 0.03 ± (0.17) |
| 1.0 | 63.54 ± (20.14) | 0.61 ± (0.49) | 0.09 ± (0.24) | 0.58 ± (0.49) | 0.15 ± (0.26) | 0.17 ± (0.33) |
| 0.5 | 62.63 ± (20.33) | 0.66 ± (0.47) | 0.24 ± (0.36) | 0.58 ± (0.49) | 0.16 ± (0.28) | 0.08 ± (0.23) |
| 0.1 | 59.60 ± (21.17) | 0.61 ± (0.48) | 0.26 ± (0.38) | 0.58 ± (0.49) | 0.23 ± (0.34) | 0.08 ± (0.20) |
| **DPGAN** | | | | | | |
| $\varepsilon$ | $\mu_{age}$ | $\mu_{infection\ suspected}$ | $\mu_{hypotensie}$ | $\mu_{infusion}$ | $\mu_{oligurie}$ | $\mu_{hypoxie}$ |
| orig | 70.08 ± (17.36) | 0.81 ± (0.39) | 0.05 ± (0.22) | 0.76 ± (0.43) | 0.02 ± (0.15) | 0.02 ± (0.14) |
| 10.0 | 62.01 ± (24.28) | 0.36 ± (0.48) | 0.95 ± (0.22) | 0.38 ± (0.49) | 0.93 ± (0.24) | 0.96 ± (0.20) |
| 3.0 | 67.24 ± (19.96) | 0.34 ± (0.47) | 0.95 ± (0.21) | 0.43 ± (0.49) | 0.95 ± (0.20) | 0.97 ± (0.18) |
| 1.0 | 68.70 ± (20.72) | 0.07 ± (0.24) | 0.82 ± (0.34) | 0.31 ± (0.46) | 0.94 ± (0.22) | 0.97 ± (0.15) |
| 0.5 | 80.01 ± (13.24) | 0.38 ± (0.49) | 0.95 ± (0.19) | 0.34 ± (0.46) | 0.55 ± (0.49) | 0.93 ± (0.23) |
| 0.1 | 69.07 ± (25.51) | 0.28 ± (0.44) | 0.45 ± (0.49) | 0.70 ± (0.45) | 0.59 ± (0.49) | 0.66 ± (0.47) |

Table 5.3: Comparison of means and standard deviations for different privacy levels on the sepsis dataset using different tabular data generation algorithms.

| **MST** | | | |
|---|---|---|---|
| $\varepsilon$ | $\mu_{amount}$ | $\mu_{totalamount}$ | $\mu_{points}$ |
| orig | 71.42 ± (100.54) | 23.97 ± (40.25) | 0.08 ± (0.58) |
| 10.0 | 115.18 ± (81.66) | 54.21 ± (25.92) | 0.87 ± (0.54) |
| 3.0 | 130.36 ± (82.18) | 54.29 ± (27.09) | 0.87 ± (0.55) |
| 1.0 | 130.54 ± (84.38) | 33.97 ± (29.73) | 0.47 ± (0.51) |
| 0.5 | 84.90 ± (72.41) | 33.98 ± (30.16) | 0.47 ± (0.51) |
| 0.1 | 71.44 ± (55.31) | 30.66 ± (29.07) | 0.25 ± (0.37) |
| **PrivBayes** | | | |
| $\varepsilon$ | $\mu_{amount}$ | $\mu_{totalamount}$ | $\mu_{points}$ |
| orig | 71.42 ± (100.54) | 23.97 ± (40.25) | 0.08 ± (0.58) |
| 10.0 | 71.15 ± (196.67) | 42.50 ± (136.90) | 0.14 ± (0.83) |
| 3.0 | 68.15 ± (175.39) | 43.67 ± (142.89) | 0.13 ± (0.79) |
| 1.0 | 71.11 ± (196.68) | 42.49 ± (136.67) | 0.14 ± (0.83) |
| 0.5 | 68.11 ± (176.25) | 43.81 ± (145.20) | 0.13 ± (0.80) |
| 0.1 | 71.13 ± (198.06) | 46.18 ± (153.00) | 0.16 ± (0.88) |

Table 5.4: Comparison of means and standard deviations for different privacy levels on the traffic fine dataset using different tabular data generation algorithms.

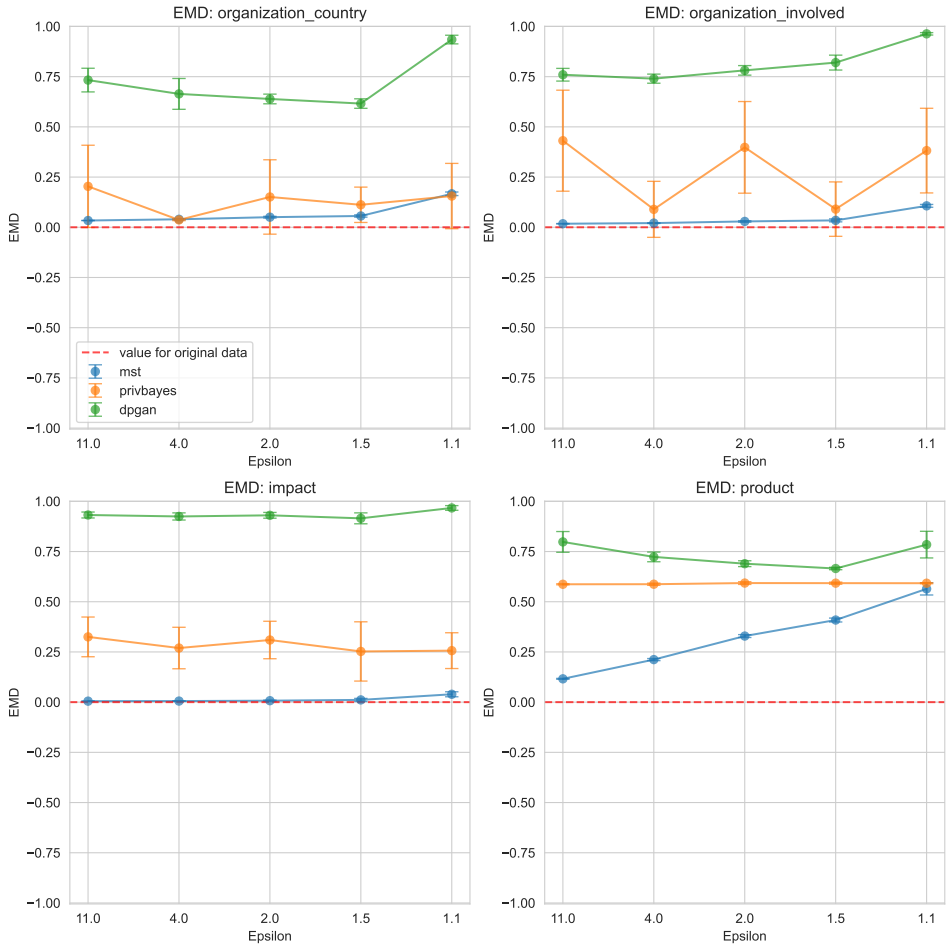


Figure 5.4: Comparison of EMD between the case attribute category distributions for the original and generated event logs for the BPIC13 event log.

**Distribution** To quantitatively assess the similarity of the case attributes distributions between the original and the anonymised event logs, we use the EMD as introduced in Section 2.6.1. In Figure 5.4, we compare the EMD between the categorical case attribute distributions of the original and generated event logs for the BPIC13 event log. For all case attributes, the *MST* algorithm shows the lowest EMD scores, indicating that the distributions are preserved better than for the other algorithms. In contrast, *DPGAN* performs the worst, with the highest EMD for all case attributes. Generally, with higher values of $\varepsilon$, the EMD increases. As seen before when looking at the mean values, the increase in noise introduced by the tabular data generation algorithms leads to a degradation of the estimation quality. Interestingly, the sharpest increase in EMD scores when decreasing the privacy budget can be observed for the case attribute `product`. This attribute contains a high number of unique values (704) with some categories being frequent and a long tail of categories. For stronger privacy guarantees, the introduced noise leads to a more uniform distribution of the categories, which increases the EMD scores.

Instead of looking at the EMD for the case attribute distributions, we can also compare the distributions visually. Figure 5.5 shows the age category distributions for the Sepsis

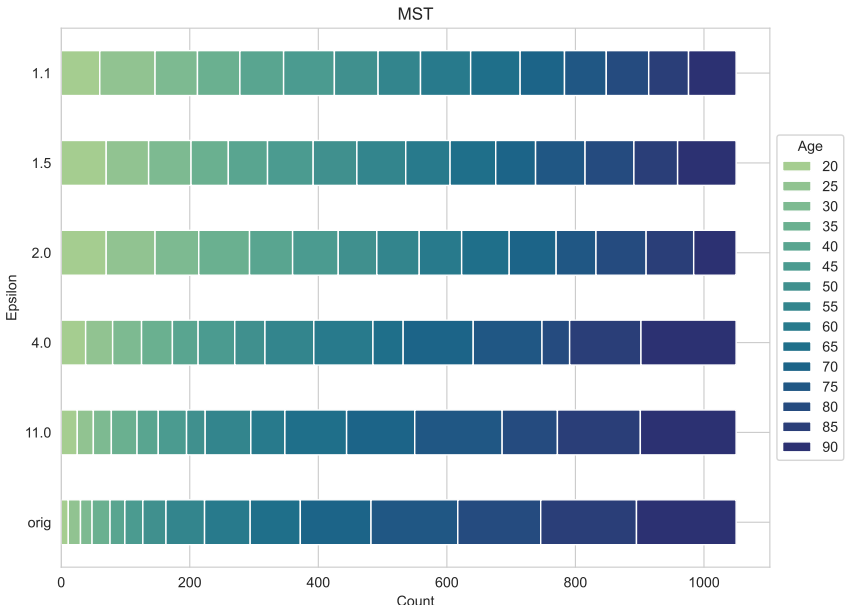


Figure 5.5: Age category distributions for the Sepsis event log using the *MST* algorithm for different values of $\varepsilon$.

event log using the *MST* algorithm for different values of $\varepsilon$. *MST* shows a clear trend of the age group distributions becoming more accurate with increasing privacy budgets. The distribution of age groups of the original event log is skewed towards the age groups of 70-94. For privacy budgets of $\varepsilon = 11$ and $\varepsilon = 4$, the distribution of the age groups is similar to the original event log. Lower $\varepsilon$ lead to a more uniform distribution of the age groups.

In summary, the evaluation of the mean, standard deviation, and distribution of case attributes shows that the algorithms achieve to produce event logs similar to the original event logs. However, the results are dependent on the type and characteristics of the case attributes. The *MST* algorithm generally preserves the mean and standard deviation better, especially for higher values of $\varepsilon$, indicating lower levels of noise introduction. *PrivBayes* shows higher deviations in mean values, particularly for binary attributes, while *DPGAN* demonstrates the most significant deviations in mean values. For the Traffic Fine event log, the results are less favourable compared to the Sepsis event log, with *PrivBayes* performing relatively better. Distribution analysis using EMD scores highlights that *MST* retains the closest resemblance to original distributions, whereas *DPGAN* exhibits the highest EMD scores. Visual inspection of `age` category distributions for event logs produced by *MST* show that the counts of age groups are better estimated for higher privacy budgets.

### 5.3.3 Relations between the Trace Variants and each Case Attribute

**Association Metrics** We examine the relations between the control flow and the case attributes by computing association metrics for the anonymised event logs as introduced in Section 2.6.1. In Figure 5.6, the association metrics between the control flow and case

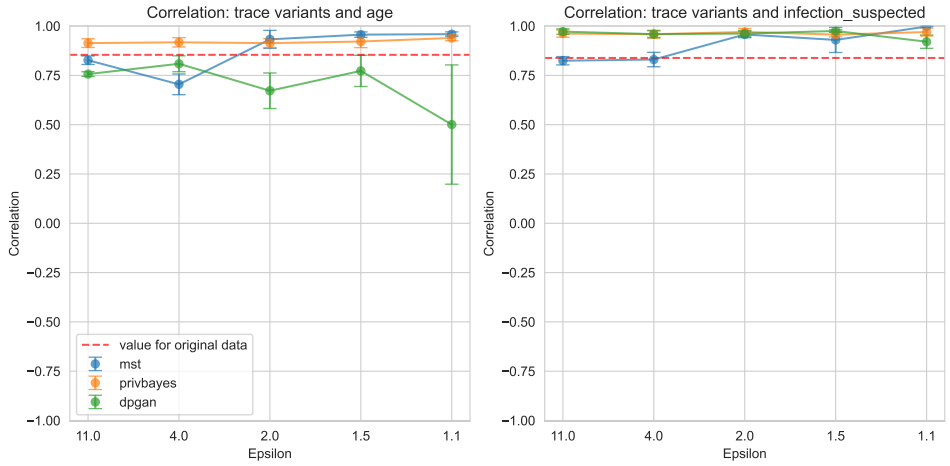


Figure 5.6: Comparison of the case variants' correlation with different case attributes for the Sepsis event log for different values of $\varepsilon$ and algorithms.

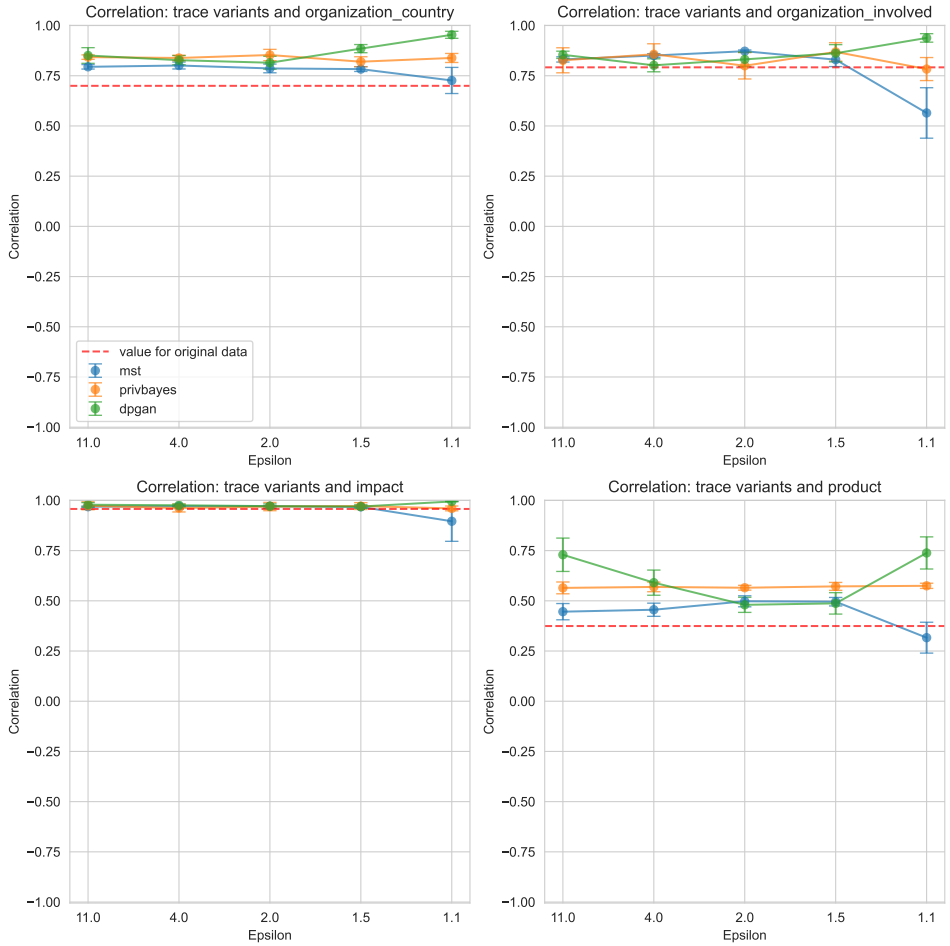


Figure 5.7: Comparison of the case variants' correlation with different case attributes for the BPIC13 event log for different values of $\varepsilon$ and algorithms.

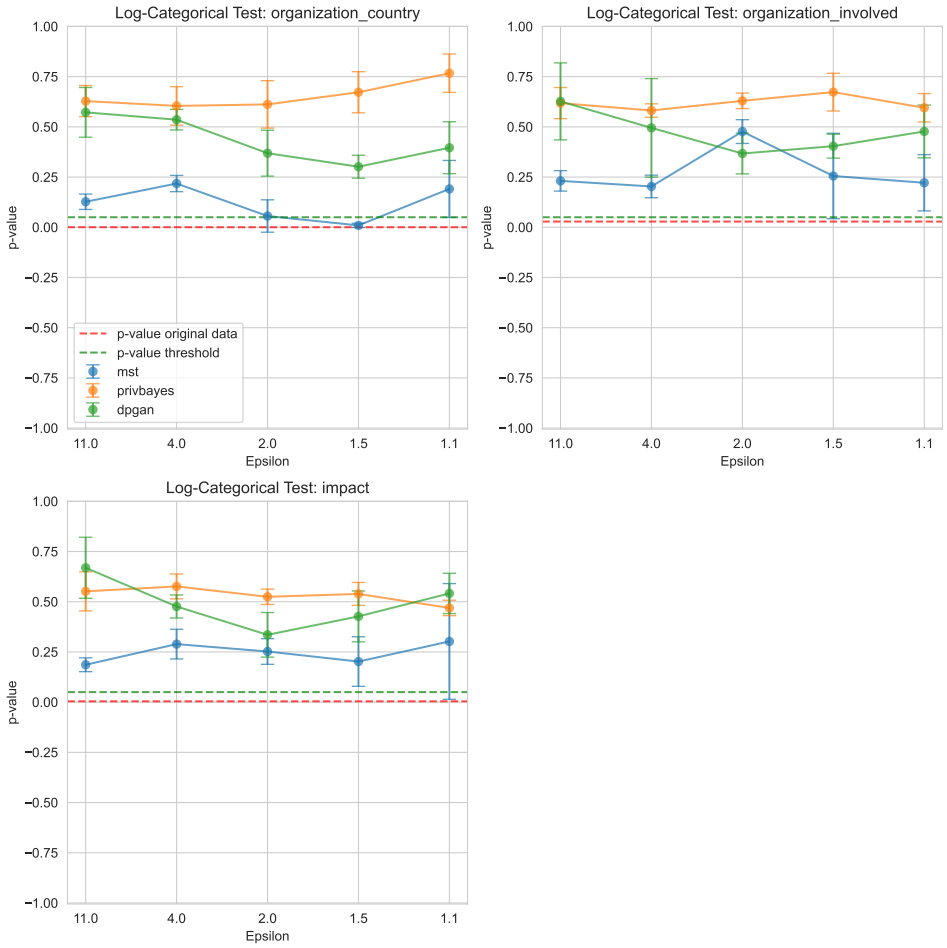


Figure 5.8: Comparison of p-values for the log-categorical test for the BPIC13 event log for different values of $\varepsilon$ and algorithms.

attributes for the Sepsis event log for different values of $\varepsilon$ and algorithms are presented. The original Sepsis event log shows a high correlation between the control flow and both case attributes `age` and `infection_suspected`. The results are mixed for the two case attributes `age` and `infection_suspected`. The *PrivBayes* algorithm shows the best results for the `age` attribute *DPGAN* shows worse results for `age` but is on par with *PrivBayes* for `infection_suspected`.

The associations between the case variants and the case attributes for the BPIC13 event log are shown in Figure 5.7. As can be seen in the figure, the original BPIC13 event log shows a high correlation between the case attributes for the case attributes `organization_country`, `organization_involved` and `impact`. All three algorithms generate anonmyised event logs that preserve these correlations. It can be seen that for the lowest privacy budget of $\varepsilon = 1.1$, the exhibited correlations in the event logs start to deviate. The correlation of the case variants and the case attribute `product` is overestimated for all algorithms.

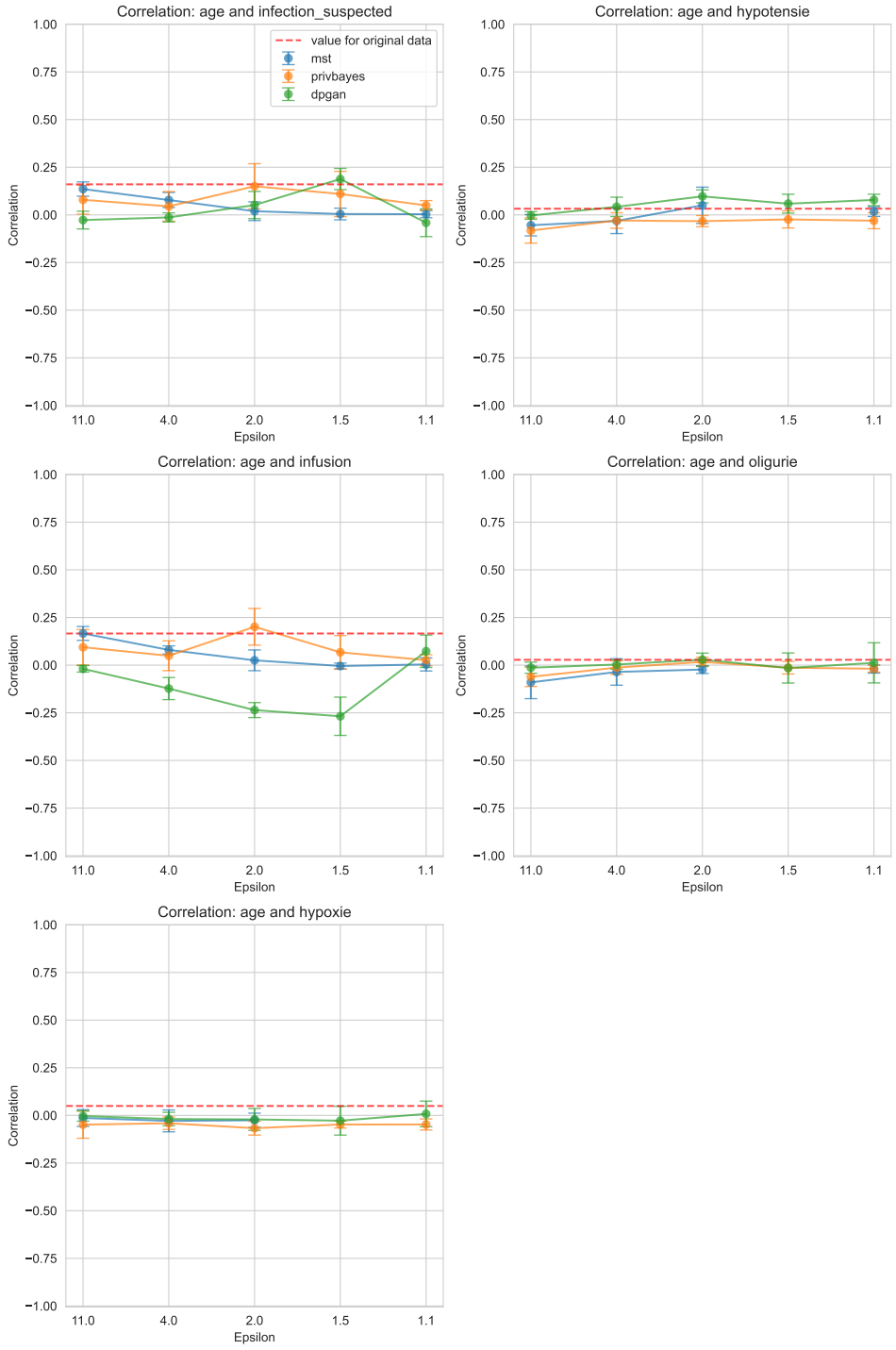


Figure 5.9: Comparison of correlation measures between case attributes for the Sepsis event log for different values of $\varepsilon$ and algorithms.

**Log-categorical Statistical Test** Figure 5.8 shows the p-values for the log-categorical statistical test for the BPIC13 event log for different values of $\varepsilon$ and algorithms. The log-categorical test as introduced in Section 2.6.4 is used to test for different behaviour of the control flow in sublogs defined by the categories of the case attributes. The lower the p-value, the more significant the difference in the control flow behaviour between the sublogs. For the original BPIC13 data set, the p-values for all case attributes are below the threshold of 0.05. Only the *MST* algorithm achieves to reproduce this for `organization_country` for $\varepsilon \in \{1, 1.5\}$. The other two algorithms do not achieve producing an event log where the control flow behaviour differs significantly between the sublogs. When comparing the p-value results for the different algorithms, the *MST* algorithm shows the best results, followed by *DPGAN* and *PrivBayes*.

In conclusion, the measures show that the algorithms produce event logs where the correlations between case variants and case attributes are preserved to some extent. *PrivBayes* outperforms the other algorithms, while sometimes overestimating the correlation of the process behaviour with a case attribute. The results for the other algorithms are less consistent. For lower privacy budgets we can observe higher variability in the output for *MST* and *DPGAN*.

### 5.3.4 Relations between the Case Attributes

We evaluate the preservation of the relations between the case attributes by comparing the correlation measures between the case attributes for the original and anonymised event logs. As introduced in Section 2.6.3, we use the Pearson correlation coefficient for numerical-numerical case attribute pairs and the point-biserial correlation for binary-numerical attribute pairs.

Figure 5.9 shows the correlation measures between the case attributes of the Sepsis event log for different values of $\varepsilon$ and algorithms. To calculate the point-biserial correlation, we interpret the age groups as numerical values. The original Sepsis event log shows a low correlation score for the case attributes `infection_suspected` and `infusion` with the case attribute `age`. The results for the *MST* algorithm show that these correlations are preserved only for $\varepsilon = 11.0$. The other algorithms produce event logs where the correlations are preserved inconsistently across the privacy budgets. *MST* shows the consistent decline in the preservation of the correlations with decreasing privacy budgets. No correlation is found between the other case attribute and the case attribute `age` in the original event log. This is reproduced by all the algorithms and for all privacy budgets.

The correlation measures between the case attribute of the Traffic Fine event log are shown in Figure 5.10. All considered case attributes in the Traffic Fine event are numerical and we use the Pearson correlation coefficient to calculate the correlations. The original Traffic Fine event log shows low correlations between all pairs of case attributes. While the *MST* algorithm underestimates the correlations between the case attributes, the *PrivBayes* algorithm overestimates the correlations. The correlation scores are consistent for the different privacy budgets except for the correlation between `amount` and `points`. For this correlation at $\varepsilon \in \{1.1, 1.5\}$, *MST* switches from underestimating the correlation to a higher correlation score.

From the results in the section, we conclude that correlations between case attributes are not preserved for all investigated algorithms. *PrivBayes* consistently achieves to detect

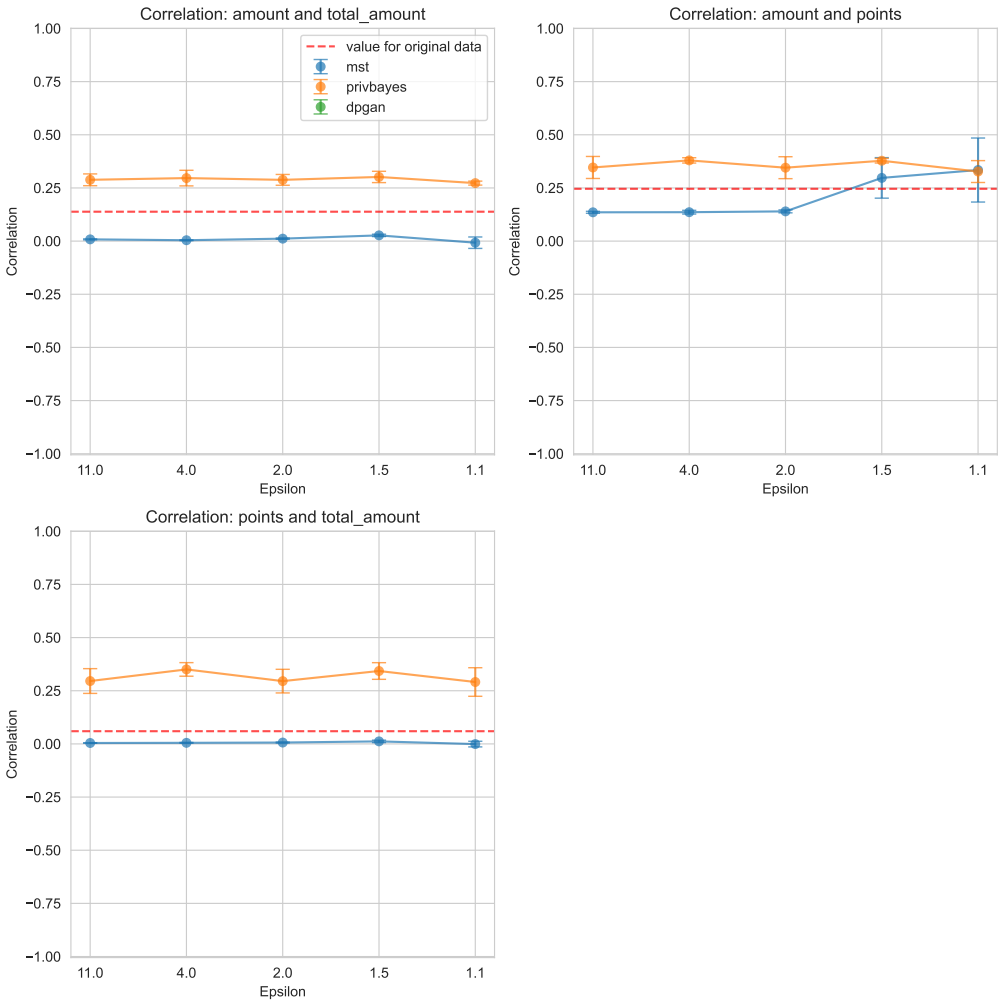


Figure 5.10: Comparison of correlation measures between case attributes for the Traffic Fine event log for different values of $\varepsilon$ and algorithms.

the correlation between attributes but produces event logs where the case attributes are correlated to strongly. The other algorithms do not reliably reproduce the correlations for many case attributes. We cannot detect a clear influence of the privacy budget on the extent to which the correlations are preserved.

| Algorithm | Sepsis | BPIC13 | Traffic Fine |
|---|---|---|---|
| MST | 13.30s | 10.86s | 16.55s |
| PrivBayes | 0.45s | 0.51s | 12.66s |
| DPGAN | 149.21s | 1081.04s | 4h15m |

Table 5.5: Comparison of average runtimes for the TDG algorithms across different event logs. The runtimes are averaged over five runs and obtained running the algorithms on a machine with an Apple M2 Pro chip and 16GB of RAM.

## 5.4 Runtime Comparison

Table 5.5 shows the runtimes of the different TDG algorithms. The results indicate that *MST* and *PrivBayes* have significantly shorter runtimes compared to *DPGAN*. Specifically, *MST* and *PrivBayes*, which are based on estimating the parameters of graphical models, completed their tasks in seconds to minutes. In contrast, *DPGAN*, which relies on a computationally intensive neural network architecture, required substantially more time, with runtimes extending to several hours for the Traffic Fine event log.

# Chapter 6

# Discussion

In this section, we discuss our framework for $(\varepsilon, \delta)$-differentially private event log publishing. The approach advances the state-of-the-art by ensuring DP not only for the control flow but also for case attributes in event logs.

We employ tabular data generation algorithms, such as MST [39], *DPGAN* [38] and PrivBayes [30], to generate synthetic event logs that are statistically similar to the original event logs. Because these algorithms are designed for tabular data, we transform the event log into a tabular dataset. We make use of *TraVaS* [45], a $(\varepsilon, \delta)$-differentially private algorithm to release trace variant distributions. This private trace variant distribution is used to limit the trace variants present in the input of the tabular data generation algorithm. A tabular data generation algorithm is then applied to the tabular dataset to ensure DP for the case attributes and relations between the case attributes. We calculate the final privacy budget for the framework using the k-fold adaptive composition theorem of DP [34]. This enables us to account for the use of *TraVaS*' output in the tabular data generation algorithm next to the original event log.

As described in the problem statement and research questions, we aim to provide a method to ensure DP for event logs while preserving the statistical characteristics of the data. Our evaluation shows that the results differ per event log and algorithm used. We find that *MST* is generally the most consistent across multiple dimensions. *MST* produces the best results regarding preserving the trace distributions by maintaining EMD scores close the baseline. Further, the statistical characteristics of case attributes are best reproduced by *MST*. The dependencies between the trace variants and the case attributes are similarly well estimated by *MST* and *PrivBayes*. Additionally, *PrivBayes* has the edge when looking at how well the correlations between case attributes are preserved. The results for *DPGAN* are inconsistent and almost always worse than the results for the other algorithms. Our findings regarding the performance of the algorithms is consistent with the benchmark performed by Tao et al. [59] using common tabular datasets.

Further, we observe that the data utility decreases with stronger privacy guarantees. For the lowest privacy budget $(\epsilon, \delta) = (1.1, 0.75)$, we see a sharp decrease in data utility for almost all measures. This is most pronounced for *MST* as for weaker privacy guarantees, *MST* manages to reproduce most of the original event log's characteristics well. *PrivBayes* and *DPGAN* show a more gradual decrease in data utility with increasing privacy levels

or even non-monotonic behaviour. However, their best results are still only on-par or worse than the results of *MST*. We see the same privacy-utility trade-off as reported in the literature [69].

On a practicality note, we find that using graphical model-based algorithms like *MST* is less complex. The quality of the algorithms' outputs is not dependent on hyperparameter tuning like is the case for the GAN-based approaches. Additionally, the runtime of the graphical model-based algorithms is significantly lower than the runtime of GAN-based approaches.

In general, we find that the framework is a promising approach to ensure DP for event logs while preserving the statistical characteristics of the data. However, in practice the choice of privacy budget and tabular data generation algorithm is crucial. Before the anonymised event log is shared, a thorough evaluation of the data utility should be performed to ensure that the anonymised event log is still useful for the intended analysis.

## 6.1 Limitations of the Evaluation

One limitation of the evaluation is the choice of the privacy budget range. We evaluate the framework in the privacy parameter range of $\varepsilon \in \{1.1, 2, 2.5, 4, 11\}$ with $\delta = 0.75$. While this range covers a wide spectrum of privacy levels, it does not include lower privacy levels. However, as seen in the evaluation, we can already see strong degradation in data utility at the lower end of the chosen privacy budget range. We expect that the data utility will further decrease at lower privacy levels. Further research is needed to verify this assumption. Additionally, we limited the evaluation to one privacy budget of $(\varepsilon_{TraVaS}, \delta_{TraVaS}) = (1, 0.5)$ for the differentially private trace variant selection of *TraVaS* [45]. This choice is based on the evaluation of *TraVaS* [45] but it limits the validity of our results. Performing a systematic evaluation of privacy budgeting to find the optimal privacy budget for the mechanisms is an interesting avenue for future research.

Another limitation lies within the choice of tabular data generation algorithms. We evaluate the framework using MST, *DPGAN* and PrivBayes. This choice of algorithms is guided by a benchmark survey by Tao et al. [59]. The availability of runnable implementations limited this selection. We cannot guarantee that the chosen algorithms are the best suited for the task. For example, *CTAB-GAN+* [70] or *DP-CTGAN* [71] are promising algorithms that could be integrated and evaluated in future work. Further, we did not perform hyperparameter tuning for the algorithms. Especially the training of *DPGAN* is sensitive to the choice of hyperparameters, which could be an explanation for the poor performance.

We evaluate the framework on three real-life event logs, the Sepsis event log, the BPI Challenge 2013 and the traffic fine event log. While the event logs are diverse in their characteristics, they still provide a limited view of the framework's applicability. Further evaluation on other event logs is necessary to validate the generalisability of the framework.

We run each tabular data generation algorithm five times to account for the non-deterministic nature of the algorithms. In the evaluation we average the results of the five runs and provide the standard deviation for the results. However, with more computational resources, we could run the algorithms more times to get a more accurate estimate

of the results.

For deriving the fitness and precision values of the generated event logs, we use the noise threshold of the inductive miner to filter out noise [68]. This threshold is set to the default value of 0.2. However, this threshold influences the discovered models and thus other noise thresholds could lead to different results and findings.

# Chapter 7

# Conclusion

The increasing use of process mining techniques has led to an increase in demand for event logs. However, sharing event logs is often not possible due to privacy concerns. Current state-of-the-art event log anonymisation techniques that guarantee differential privacy allow for the release of event logs. Unfortunately, so far, the anonymised event logs only contain information about the control flow of the process. Including case attributes in the anonymised event logs allows for more detailed analysis.

This thesis proposes a framework for generating anonymised event logs with case attributes. The framework ensures $(\varepsilon, \delta)$-differential privacy for the generated event log. We use a combination of a differentially private trace variant release algorithm and a tabular data generation algorithm. The trace variant release algorithm restricts the trace variants produced by the tabular data generation algorithm. This ensures that the generated trace variants in the generated event log are differentially private. The tabular data generation algorithm is used to anonymise the case attributes and the dependencies between all event log contents.

We evaluated the framework on three real-life event logs for different tabular data generation algorithms and privacy budgets. We find that MST produces the overall best results regarding the generated event log's similarity to the original event log. Further, we find that for stronger privacy guarantees, i.e. lower privacy budget values, the similarity to the original event log decreases. Hence, there is a trade-off between privacy and utility and the privacy budget should be chosen carefully for practical use. As Section 6 mentions, careful evaluation is necessary to determine if the anonymised event log is still useful for the indented analysis. The framework's flexibility regarding tabular data generation algorithms ensures that future advancements in tabular data generation algorithms can be tapped into.

We see several interesting avenues for future work. First, further evaluation of the framework can provide useful insights into the strengths and weaknesses of the approach. We limited the evaluation to three event logs; however, evaluating other event logs with different characteristics ensures the generalisability of the results and might reveal new insights. Further, constructing synthetic event logs with known characteristics would allow for a more controlled evaluation. While we already saw a sharp decrease in data utility for the lower end of the privacy budget range, further evaluation is needed to assess the trade-off

between privacy and utility.

Second, the choice of tabular data generation algorithms is crucial for the framework's performance. Developing a new tabular data generation algorithm or integrating existing algorithms that have yet to be implemented in the framework can improve the framework's performance. Sticking to already used algorithms, a systematic evaluation of hyperparameters' influence can help improve their performance.

Lastly, extending the framework to allow for the inclusion of other dimensions in the event log can further enhance its utility for more complex analyses. For example, timestamps or updates to event attributes with more granularity could be included.

# Bibliography


[1] W. van der Aalst, A. Adriansyah, A. K. A. de Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. van den Brand, R. Brandtjen, J. Buijs, A. Burattin, J. Carmona, M. Castellanos, J. Claes, J. Cook, N. Costantini, F. Curbera, E. Damiani, M. de Leoni, P. Delias, B. F. van Dongen, M. Dumas, S. Dustdar, D. Fahland, D. R. Ferreira, W. Gaaloul, F. van Geffen, S. Goel, C. Günther, A. Guzzo, P. Harmon, A. ter Hofstede, J. Hoogland, J. E. Ingvaldsen, K. Kato, R. Kuhn, A. Kumar, M. La Rosa, F. Maggi, D. Malerba, R. S. Mans, A. Manuel, M. McCreesh, P. Mello, J. Mendling, M. Montali, H. R. Motahari-Nezhad, M. zur Muehlen, J. Munoz-Gama, L. Pontieri, J. Ribeiro, A. Rozinat, H. Seguel Pérez, R. Seguel Pérez, M. Sepúlveda, J. Sinur, P. Soffer, M. Song, A. Sperduti, G. Stilo, C. Stoel, K. Swenson, M. Talamo, W. Tan, C. Turner, J. Vanthienen, G. Varvaressos, E. Verbeek, M. Verdonk, R. Vigo, J. Wang, B. Weber, M. Weidlich, T. Weijters, L. Wen, M. Westergaard, and M. Wynn, "Process Mining Manifesto," in *Business Process Management Workshops*. Berlin, Heidelberg: Springer, 2012, pp. 169–194.

[2] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive Monitoring of Business Processes: A Survey," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 962–977, Nov. 2018.

[3] S. J. J. Leemans, S. Shabaninejad, K. Goel, H. Khosravi, S. Sadiq, and M. T. Wynn, "Identifying Cohorts: Recommending Drill-Downs Based on Differences in Behaviour for Process Mining," in *Conceptual Modeling*. Cham: Springer International Publishing, 2020, pp. 92–102.

[4] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council.

[5] Commonwealth of Australia, "Privacy act 1988," 1988.

[6] S. N. von Voigt, S. A. Fahrenkrog-Petersen, D. Janssen, A. Koschmider, F. Tschorsch, F. Mannhardt, O. Landsiedel, and M. Weidlich, "Quantifying the re-identification risk of event logs for process mining: Empiricial evaluation paper," in *Advanced Information Systems Engineering*, vol. 12127. Nature Publishing Group, 2020, p. 252.

[7] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. F. Sani, A. Koschmider, F. Mannhardt, S. N. Von Voigt, M. Rafiei, and L. V. Waldthausen, "Privacy and Confidentiality in Process Mining: Threats and Research Challenges," *ACM Trans. Manage. Inf. Syst.*, vol. 13, no. 1, pp. 1–17, Mar. 2022.

[8] F. Mannhardt, A. Koschmider, N. Baracaldo, M. Weidlich, and J. Michael, "Privacy-preserving process mining: Differential privacy for event logs," *Business & Information Systems Engineering*, vol. 61, pp. 595–614, 2019.

[9] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-preserving data publishing: A survey of recent developments," *ACM Comput. Surv.*, vol. 42, no. 4, jun 2010.

[10] L. Sweeney, "Simple demographics often identify people uniquely," *Health (San Francisco)*, vol. 671, no. 2000, pp. 1–34, 2000.

[11] A. Narayanan and V. Shmatikov, "How to break anonymity of the netflix prize dataset," 2007.

[12] L. Sweeney, "k-anonymity: A model for protecting privacy," *International journal of uncertainty, fuzziness and knowledge-based systems*, vol. 10, no. 05, pp. 557–570, 2002.

[13] F. Mannhardt, S. A. Petersen, and M. F. Oliveira, "Privacy challenges for process mining in human-centered industrial environments," in *2018 14th International Conference on Intelligent Environments (IE)*, 2018, pp. 64–71.

[14] W. M. van der Aalst, "Responsible data science: using event data in a "people friendly" manner," in *Enterprise Information Systems: 18th International Conference, ICEIS 2016, Rome, Italy, April 25–28, 2016, Revised Selected Papers 18*. Springer, 2017, pp. 3–28.

[15] G. Elkoumy, S. A. Fahrenkrog-Petersen, M. F. Sani, A. Koschmider, F. Mannhardt, S. N. n. Von Voigt, M. Rafiei, and L. V. Waldthausen, "Privacy and confidentiality in process mining: Threats and research challenges," *ACM Trans. Manage. Inf. Syst.*, vol. 13, no. 1, oct 2021.

[16] A. Pika, M. T. Wynn, S. Budiono, A. H. ter Hofstede, W. M. van der Aalst, and H. A. Reijers, "Privacy-preserving process mining in healthcare," *International Journal of Environmental Research and Public Health*, vol. 17, no. 5, 2020. [Online]. Available: https://www.mdpi.com/1660-4601/17/5/1612

[17] M. Rafiei and W. M. van der Aalst, "Privacy-preserving data publishing in process mining," in *Business Process Management Forum: BPM Forum 2020, Seville, Spain, September 13–18, 2020, Proceedings 18*. Springer, 2020, pp. 122–138.

[18] S. A. Fahrenkrog-Petersen, H. van der Aa, and M. Weidlich, "Optimal event log sanitization for privacy-preserving process mining," *Data & Knowledge Engineering*, vol. 145, p. 102175, 2023.

[19] M. Rafiei, M. Wagner, and W. M. van der Aalst, "Tlkc-privacy model for process mining," in *International Conference on Research Challenges in Information Science*. Springer, 2020, pp. 398–416.

[20] G. Elkoumy, A. Pankova, and M. Dumas, "Differentially private release of event logs for process mining," *Information Systems*, vol. 115, p. 102161, May 2023.

[21] S. A. Fahrenkog-Petersen, M. Kabierski, F. Rösel, H. van der Aa, and M. Weidlich, "Sacofa: Semantics-aware control-flow anonymization for process mining," in *2021 3rd International Conference on Process Mining (ICPM)*. IEEE, 2021, pp. 72–79.

[22] S. A. Fahrenkrog-Petersen, H. van der Aa, and M. Weidlich, "PRIPEL: Privacy-Preserving Event Log Publishing Including Contextual Information," in *Business Process Management*, ser. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 111–128.

[23] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning.* Springer, 2006, vol. 4, no. 4.

[24] C. Nwankpa, W. L. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *ArXiv*, vol. abs/1811.03378, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:53208763

[25] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers.* Springer, 2010, pp. 177–186.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, p. 139–144, oct 2020.

[28] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning.* PMLR, 2017, pp. 214–223.

[29] A. Mittal and A. Kassim, *Bayesian network technologies: applications and graphical models: applications and graphical models.* IGI global, 2007.

[30] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," *ACM Trans. Database Syst.*, vol. 42, no. 4, oct 2017.

[31] R. Mckenna, D. Sheldon, and G. Miklau, "Graphical-model based estimation and inference for differential privacy," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 09–15 Jun 2019, pp. 4435–4444.

[32] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, ser. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2006, p. 1–12.

[33] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[34] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in *International conference on machine learning.* PMLR, 2015, pp. 1376–1385.

[35] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography*, S. Halevi and T. Rabin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 265–284.

[36] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[37] A. Yale, S. Dash, R. Dutta, I. Guyon, A. Pavao, and K. P. Bennett, "Generation and evaluation of privacy preserving synthetic health data," *Neurocomputing*, vol. 416, pp. 244–255, 2020.

[38] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," 2018.

[39] R. McKenna, G. Miklau, and D. Sheldon, "Winning the nist contest: A scalable and general approach to differentially private synthetic data," *arXiv preprint arXiv:2108.04978*, 2021.

[40] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International journal of computer vision*, vol. 40, pp. 99–121, 2000.

[41] S. J. Leemans, A. F. Syring, and W. M. van der Aalst, "Earth movers' stochastic conformance checking," in *Business Process Management Forum: BPM Forum 2019, Vienna, Austria, September 1–6, 2019, Proceedings 17*. Springer, 2019, pp. 127–143.

[42] L. Yujian and L. Bo, "A normalized levenshtein distance metric," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1091–1095, 2007.

[43] A. Berti and W. M. van der Aalst, "A novel token-based replay technique to speed up conformance checking and process enhancement," in *Transactions on Petri Nets and Other Models of Concurrency XV*. Springer, 2021, pp. 1–26.

[44] J. Munoz-Gama and J. Carmona, "A fresh look at precision in process conformance," in *International Conference on Business Process Management*. Springer, 2010, pp. 211–226.

[45] M. Rafiei, F. Wangelik, and W. M. P. van der Aalst, "Travas: Differentially private trace variant selection for process mining," in *Process Mining Workshops*. Cham: Springer Nature Switzerland, 2023, pp. 114–126.

[46] D. Freedman, R. Pisani, and R. Purves, "Statistics (international student edition)," *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[47] D. Kornbrot, "Point biserial correlation," *Wiley StatsRef: Statistics Reference Online*, 2014.

[48] S. J. Leemans, J. M. McGree, A. Polyvyanyy, and A. H. ter Hofstede, "Statistical tests and association measures for business processes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 7, pp. 7497–7511, 2023.

[49] A. Burattin, M. Conti, and D. Turato, "Toward an anonymous process mining," in *2015 3rd International Conference on Future Internet of Things and Cloud*. IEEE, 2015, pp. 58–63.

[50] M. Rafiei, L. Von Waldthausen, and W. M. van der Aalst, "Supporting confidentiality in process mining using abstraction and encryption," in *International Symposium on Data-Driven Process Discovery and Analysis*. Springer, 2018, pp. 101–123.

[51] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *Acm transactions on knowledge discovery from data (tkdd)*, vol. 1, no. 1, pp. 3–es, 2007.

[52] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd international conference on data engineering*. IEEE, 2006, pp. 106–115.

[53] G. Elkoumy, A. Pankova, and M. Dumas, "Mine me but don't single me out: Differentially private event logs for process mining," in *2021 3rd International Conference on Process Mining (ICPM)*, 2021, pp. 80–87.

[54] G. Elkoumy and M. Dumas, "Libra: High-Utility Anonymization of Event Logs for Process Mining via Subsampling," in *2022 4th International Conference on Process Mining (ICPM)*. Bolzano, Italy: IEEE, Oct. 2022, pp. 144–151.

[55] Y. Zhu and Y.-X. Wang, "Poission subsampled rényi differential privacy," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 7634–7642. [Online]. Available: https://proceedings.mlr.press/v97/zhu19c.html

[56] K. Chaudhuri and N. Mishra, "When random sampling preserves privacy," in *Advances in Cryptology - CRYPTO 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 198–213.

[57] M. Rafiei, F. Wangelik, M. Pourbafrani, and W. M. P. van der Aalst, "TraVaG: Differentially Private Trace Variant Generation Using GANs," in *Research Challenges in Information Science: Information Science and the Connected World*, ser. Lecture Notes in Business Information Processing. Cham: Springer Nature Switzerland, 2023, pp. 415–431.

[58] J. Li, B. J. Cairns, J. Li, and T. Zhu, "Generating synthetic mixed-type longitudinal electronic health records for artificial intelligent applications," *npj Digit. Med.*, vol. 6, no. 1, pp. 1–18, May 2023, number: 1 Publisher: Nature Publishing Group.

[59] Y. Tao, R. McKenna, M. Hay, A. Machanavajjhala, and G. Miklau, "Benchmarking differentially private synthetic data generation algorithms," *arXiv preprint arXiv:2112.09238*, 2021.

[60] J. Jordon, J. Yoon, and M. Van Der Schaar, "Pate-gan: Generating synthetic data with differential privacy guarantees," in *International conference on learning representations*, 2018.

[61] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[62] A. Berti, S. van Zelst, and D. Schuster, "Pm4py: A process mining library for python," *Software Impacts*, vol. 17, p. 100556, 2023.

[63] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445. Austin, TX, 2010, pp. 51–56.

[64] Z. Qian, B.-C. Cebere, and M. van der Schaar, "Synthcity: facilitating innovative use cases of synthetic data in different data modalities," 2023.

[65] F. Mannhardt *et al.*, "Sepsis cases-event log," *Eindhoven university of technology*, vol. 10, 2016.

[66] B. Van Dongen, B. Weber, D. Ferreira, and J. De Weerdt, "Business process intelligence challenge 2013," in *CEUR Workshop Proceedings*, vol. 1052. Citeseer, 2013.

[67] M. M. de Leoni and F. Mannhardt, "Road traffic fine management process," 2015.

[68] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, "Discovering block-structured process models from event logs containing infrequent behaviour," in *Business Process Management Workshops*. Cham: Springer International Publishing, 2014, pp. 66–78.

[69] T. Li and N. Li, "On the tradeoff between privacy and utility in data publishing," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 517–526.

[70] Z. Zhao, A. Kunar, R. Birke, H. Van der Scheer, and L. Y. Chen, "Ctab-gan+: Enhancing tabular data synthesis," *Frontiers in big Data*, vol. 6, p. 1296508, 2024.

[71] M. L. Fang, D. S. Dhami, and K. Kersting, "Dp-ctgan: Differentially private medical data generation using ctgans," in *International Conference on Artificial Intelligence in Medicine*. Springer, 2022, pp. 178–188.