



Releasing differentially private event logs using generative models[☆]

Frederik Wangelik^{ID}*, Majid Rafiei^{ID}, Mahsa Pourbafrani^{ID}, Wil M.P. van der Aalst^{ID}

Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany

ARTICLE INFO

Keywords:

Process mining
Event data
Differential privacy
GANs
Diffusion models
Machine Learning

ABSTRACT

In recent years, the industry has been witnessing an extended usage of process mining and automated event data analysis. Consequently, there is a rising significance in addressing privacy apprehensions related to the inclusion of sensitive and private information within event data utilized by process mining algorithms. State-of-the-art research mainly focuses on providing quantifiable privacy guarantees, e.g., via differential privacy, for trace variants that are used by the main process mining techniques, e.g., process discovery. However, privacy preservation techniques designed for the release of trace variants are still insufficient to meet all the demands of industry-scale utilization. Moreover, ensuring privacy guarantees in situations characterized by a high occurrence of infrequent trace variants remains a challenging endeavor. In this paper, we introduce two novel approaches for releasing differentially private trace variants based on trained generative models. With TraVaG, we leverage *Generative Adversarial Networks* (GANs) to sample from a privatized implicit variant distribution. Our second method employs *Denoising Diffusion Probabilistic Models* that reconstruct artificial trace variants from noise via trained Markov chains. Both methods offer industry-scale benefits and elevate the degree of privacy assurances, particularly in scenarios featuring a substantial prevalence of infrequent variants. Also, they overcome the shortcomings of conventional privacy preservation techniques, such as bounding the length of variants and introducing fake variants. Experimental results on real-life event data demonstrate that our approaches surpass state-of-the-art techniques in terms of privacy guarantees and utility preservation.

1. Introduction

Process mining encompasses a set of data-driven techniques used for the discovery, analysis, and enhancement of business processes. These techniques rely on event data, readily available in various information systems such as ERP, SCM, and CRM systems. Over the past decade, process mining and event data analysis have been effectively implemented across various industries, playing a pivotal role in business success.

Much like other data-driven approaches within the broader field of data science, concerns surrounding the privacy of individuals whose data is subject to process mining algorithms have emerged due to the increasing volume of event data and its utilization. Prominent examples are business process management applications in the health care or governmental sector that use sensitive, personal data records to provide decision support. Consequently, privacy regulations such as GDPR (General Data Protection Regulation) [1] place restrictions on data storage and processing, thereby driving the development of privacy preservation techniques.

[☆] Funded under the Excellence Strategy of the Federal Government and the Länder. We also thank the Alexander von Humboldt Stiftung for supporting our research.

* Corresponding author.

E-mail address: frederik.wangelik@rwth-aachen.de (F. Wangelik).

<https://doi.org/10.1016/j.datak.2025.102450>

Received 22 October 2023; Received in revised form 26 February 2025; Accepted 4 April 2025

Available online 15 April 2025

0169-023X/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

Table 1

A simple event log from the healthcare context, including trace variants and their frequencies.

Trace variant	Frequency
$\langle \text{register}, \text{visit}, \text{blood-test}, \text{visit}, \text{release} \rangle$	13
$\langle \text{register}, \text{blood-test}, \text{visit}, \text{release} \rangle$	14
$\langle \text{register}, \text{visit}, \text{hospitalization}, \text{surgery}, \text{release} \rangle$	4
$\langle \text{register}, \text{visit}, \text{blood-test}, \text{blood-test}, \text{release} \rangle$	3

Contemporary techniques for preserving privacy primarily rely on Differential Privacy (DP), a privacy framework that introduces controlled noise into data [2]. This choice is driven by DP's notable qualities, such as its capacity to offer rigorous mathematical privacy protection and guard against PSO (predicate-singling-out) attacks [3]. The aim of DP-based approaches is to introduce noise into the released output in order to conceal the involvement of an individual. Leading-edge research in the field of process mining, incorporating privacy preservation based on DP, focuses on the release of distributions of trace variants. These distributions form the basis for core process mining techniques, namely, process discovery and conformance checking [4]. A trace variant refers to a complete sequence of activities performed by an action or agent. Often, this data contains private information. For instance, in the healthcare context, a trace variant refers to a complete sequence of treatment-related activities performed for a patient that is private information itself and can also be exploited to derive other sensitive information, e.g., the disease of the patient. Table 1 shows a sample of a trace variant distribution in the healthcare context. It is important to note that each trace variant within a distribution is linked to an individual, known as a *case* and no case should have more than one associated trace variant [4]. Hence, this data excerpt motivates that no unauthorized intermediary should be able to link such activities back to individual patients or groups of patients.

To implement Differential Privacy (DP) for trace variants, conventional methods, often referred to as *prefix-based* approaches, introduce noise from a Laplacian distribution into the variant distribution derived from an event log, as discussed in references such as [5,6]. These approaches involve the generation of all possible unique variants based on a given set of activities to ensure the original distribution of variants is DP-compliant. However, since the number of potential variants that can be generated from a set of activities is infinite, prefix-based techniques must place constraints on the length of the generated sequences. Furthermore, to narrow down the search space, these methods typically employ a pruning parameter to exclude less frequent prefixes. This DP generation process comes with significant computational complexity and results in several drawbacks, including (1) *introducing fake variants*, (2) *removing frequent true variants*, and (3) *having limited length for generated variants* [7].

Several approaches have been put forth to address the challenges mentioned above, either partially or in their entirety. One such method, known as SaCoFa, as discussed in [8], aims to alleviate the first and second drawbacks by leveraging insights into the underlying process semantics from the original event data. However, the paper does not delve into the privacy quantification of the additional queries made to acquire knowledge about the underlying semantics. Moreover, the third drawback persists as SaCoFa itself is a prefix-based approach. In [9] and a related work called Libra [10], which builds upon [9], trace variants are transformed into a representation using a Deterministic Acyclic Finite State Automata (DAFSA) to circumvent the mentioned issues. Nonetheless, Libra introduces a clipping parameter to filter out infrequent variants. This clipping parameter grows based on the number of unique trace variants and the strength of privacy guarantees. Consequently, depending on the number of unique trace variants and privacy parameters, Libra may even eliminate all variants, resulting in empty outputs. A recent solution, TraVaS, described in [11], proposes an approach based on differentially private partition selection strategies to tackle the aforementioned challenges. Similar to Libra, TraVaS also requires the removal of infrequent trace variants. However, in TraVaS, the threshold for discarding infrequent variants solely depends on the input privacy parameters and does not increase with the number of unique variants or the size of event data. Nevertheless, for small event data with a high prevalence of unique trace variants, TraVaS might face limitations in providing robust privacy guarantees.

In [7], we introduced *TraVaG* as a fundamentally new approach that incorporates trained, generative models to create differentially private trace variants from an initial input variant distribution. The core concept behind TraVaG involves privately learning crucial characteristics of event data from an underlying event log through the utilization of deep Autoencoder- and Generative Adversarial Networks (GANs) [12]. The GAN, once trained, empowers the generation of new synthetic anonymized variants that closely align with the statistical properties of the original data. To date of publication, it was the first research that had explored the potential of differentially private deep generative artificial neural networks in the domain of process discovery from event log data. Besides several conceptual advantages over traditional selection-based methods such as a data-independent, memory-efficient application and runtime, a pretraining option, and frequency threshold independence, we exemplified stronger data utility preservation capabilities on real-life event data, especially in DP parameter regions of small δ and complex variant distributions.

This paper is an extension of TraVaG [7] that expands the experimental verification and analysis of the TraVaG framework and introduces a new generative model in a differentially private setup for transforming confidential event log data into anonymized trace variant samples. The new anonymization framework is based on *Denosing Diffusion Probabilistic Models* (DDPMs) [13] that already proved to show the best state-of-the-art generative performance on image data [14,15]. Instead of directly using deep networks to approximate variant statistics, DDPMs are iterative probabilistic models that leverage two types of Markov Chains to first gradually add noise to their training data and then learn to reverse the perturbation by denoising mechanisms. In this paper, we transfer the concept of DDPMs into a differentially private environment with event data structures and compare the approach as

well as its performance with our prior GAN-based TraVaG algorithm. This research not only represents a novel effective privatization scheme for the world of process mining, but it is also the first work that investigates the impact of differentially private DDPMs on mixed-type tabular inputs independent of the process mining realm. In addition, we extend the performance analysis of both algorithms to different data structures and more extended (ϵ, δ) DP parameter regimes. Whereas in [7], the focus had been put on rather small, complex trace variant distributions, we now also include a larger and more generic as well as realistic sample event log for the utility comparison between DDPMs and our prior TraVaG approach.

Both approaches have different advantages and use cases. Whereas TraVaG provides fast sampling and large flexibility for complex event logs, differentially private DDPMs allow for faster training, more stable conversion, and less complex architectures. Generally, TraVaG and DDPM as generative models are deployed without data access. Thus, as long as the statistical characteristics of the original data do not significantly change, one does not need to apply DP directly to the original event data. For industry-scale big event data, this property can considerably improve the computational complexities [16]. Moreover, both methods are based on DP-SGD (Differentially Private-Stochastic Gradient Descent) [17] optimization techniques that avoid thresholding on training data or released network outputs. Hence, they can generate infinite and arbitrarily large anonymized synthetic trace variants even if the original variant frequencies are comparably small. Moreover, our experiments on real-life event logs demonstrate the superior performance of both approaches compared to state-of-the-art techniques in terms of data utility preservation for the same privacy guarantees.

The remainder of this paper is structured as follows. In Section 2, we provide a summary of related work. Preliminaries and notations are provided in Section 3. In Section 4, we present the details of TraVaG and the DDPM framework. Section 5 discusses the experimental results based on real-life event logs. In Section 6, a brief summary of privacy, complexity, and data-related challenges is provided, and Section 7 concludes the paper.

2. Related work

Privacy-preserving process mining is recently growing in importance. Several techniques have been proposed to address privacy issues in process mining. In this paper, our focus is on the combination of generative models and so-called *noise-based* anonymization techniques that are based on the notion of *differential privacy*. In the following, we thus provide a summary of relevant work focusing on releasing *differentially private event data* and generating *differentially private synthetic data*.

2.1. Releasing differentially private event data

In [5], the authors apply an (ϵ, δ) -Differential Privacy (DP) mechanism to event logs to safeguard the privacy of *directly-follows relations* and trace variants. This approach combines an (ϵ, δ) -DP noise generator with an iterative query engine, enabling the anonymous release of trace variants with a predefined upper limit on their length. In a subsequent work, [8], SaCoFa is introduced as an extension of [5]. Its primary objective is to optimize query structures by incorporating underlying semantics. Another extension of [5] is PRIPEL, described in [6], where additional event attributes are integrated into the privatized event data.

All the above-mentioned techniques follow the prefix-based approach, which has inherent drawbacks, as discussed in Section 1. To tackle these challenges, the authors of [9] introduce a novel method that converts a trace variant distribution into a Deterministic Acyclic Finite State Automata (DAFSA) representation. This approach aims to retain all original trace variants while minimizing the injection of excessive noise during the anonymization process. A more recent approach, Libra, as outlined in [10], builds upon the concepts presented in [9]. Libra focuses on enhancing utility through subsampling and composing privatized subsamples to release differentially private event data. Additionally, TraVaS, as detailed in [11], presents a novel approach based on differentially private partition selection strategies to address the challenges mentioned in Section 1. Although this method avoids generating fake variants or sequences limited in length, the principle of partition selection still introduces a threshold for infrequent variants.

2.2. Generating differentially private synthetic data

While Differential Privacy (DP)-based generative Artificial Neural Networks (ANNs) have seen substantial research in various data science and machine learning domains, their application in the context of process mining is relatively unexplored. Therefore, we offer a brief overview of relevant work pertaining to structured tabular data.

In [18], the primary focus is on the challenge of generating mixed-type labeled data with a choice of k possible labels. The algorithm, known as DP-SYN, initially divides the dataset into k labeled subsets and subsequently conducts private training of an autoencoder on each partition. In [19], a similar approach is adopted, but instead of an anonymous autoencoder, a *variational autoencoder* (DP-VAE) is employed. DP-VAE assumes that the mapping from real data to a Gaussian distribution can be efficiently learned.

Taking a different direction, [20] explores the use of a Wasserstein Generative Adversarial Network (WGAN) to generate differentially private mixed-type synthetic outputs, utilizing a Wasserstein-distance-based loss function. Building on the concepts introduced in [20,21] combines the principles of WGAN and DP-VAE. It first learns a private data encoding and subsequently generates encoded data. This combined approach was adapted by TraVaG to address the challenges posed by the high dimensionality of event data.

Finally, in [22], the authors describe how to train DDPMs on high-dimensional tabular records by introducing multinomial diffusion models for categorical features. Despite promising sampling performance, the authors, however, did not include DP in their training routines. To the best of our knowledge, our paper thus introduces the first differentially private DDPM experiments on complex tabular data. In the context of non-private generative models for process mining, research primarily focuses on exploiting ANNs and GANs to predict the next state of processes such as [23,24], and [25].

3. Preliminaries

In this section, we introduce the main concepts and definitions utilized throughout the paper. We start with introducing basic notations and mathematical concepts. Let A be a set. $B(A)$ is the set of all multisets over A . Given a multiset $B \in B(A)$ over the elements of a set A , $B(a)$ is the frequency of $a \in B$. Given B_1 and B_2 as two multisets, $B_1 \uplus B_2$ is the sum over multisets, e.g., $[a^2, b^3] \uplus [b^2, c^2] = [a^2, b^5, c^2]$. We define a finite sequence over A of length n as $\sigma = \langle a_1, a_2, \dots, a_n \rangle$ where $\sigma(i) = a_i \in A$ for all $i \in \{1, 2, \dots, n\}$. The set of all finite sequences over A is denoted with A^* .

3.1. Event data (Log)

Process mining techniques employ event data, which typically consist of unique events recorded for each activity execution and are characterized by their attributes, e.g., *activity* and *timestamp*. In this context, a *trace* represents a single execution of a process, comprising a sequence of events related to the same case (individual) and organized in a specific order based on timestamps. Each event can only belong to one trace, and it cannot be repeated within the same trace. Our research primarily concentrates on the control-flow aspect of event logs, where only the *activity* attribute of events within a trace is considered. This specific perspective is referred to as a *trace variant*. Therefore, we define a simplified event log as a multiset of trace variants.

Definition 1 (Simple Event Log). A simple event log L is defined as a multiset of trace variants $L \in B(A^*)$. $\mathcal{L}$ denotes the universe of simple event logs.

Note that in a simple event log representing a distribution of trace variants, one case, which refers to an individual, cannot contribute to more than one trace variant. At the same time, one trace variant can belong to several cases.

3.2. Differential Privacy (DP)

The main idea of differential privacy revolves around introducing controlled noise into the original data in a manner that makes it practically impossible for an observer to definitively discern whether the information of a particular individual is contained within the data [2]. The amount of noise is governed by two key privacy parameters: ϵ , which quantifies the privacy loss (smaller values indicate stronger privacy), and delta δ , which represents the probability of exceeding the privacy loss bound. In the context of our study, which focuses on simple event logs or the distribution of trace variants as our sensitive event data, we specify the definition of differential privacy as outlined in Definition 2.

Definition 2 ((ϵ, δ) -DP for Event Logs). Let L_1 and L_2 be two neighboring event logs that differ only in a single entry, i.e., $L_2 = L_1 \uplus [\sigma]$ for any $\sigma \in A^*$. Also, let $\epsilon \in \mathbb{R}_{>0}$ and $\delta \in \mathbb{R}_{>0}$ be two privacy parameters. A randomized mechanism $\mathcal{M}_{\epsilon, \delta} : \mathcal{L} \rightarrow \mathcal{L}$ provides (ϵ, δ) -DP if for all $S \subseteq B(A^*)$: $Pr[\mathcal{M}_{\epsilon, \delta}(L_1) \in S] \leq e^\epsilon \times Pr[\mathcal{M}_{\epsilon, \delta}(L_2) \in S] + \delta$.

In Definition 2, ϵ as the first privacy parameter, specifies the probability ratio, and δ as the second privacy parameter allows for a linear violation. In the strict case of $\delta = 0$, $\mathcal{M}$ offers ϵ -DP. The randomness of respective mechanisms is typically ensured by the noise drawn from a probability distribution that perturbs the original trace variant distribution and results in non-deterministic outputs. When privacy parameters are set to smaller values, it results in a greater injection of noise into the mechanism's outputs. This, in turn, reduces the probability of deducing the existence of specific instances from these outputs.

3.3. Generative Adversarial Networks (GANs)

Generative Artificial Networks (GANs) represent a class of artificial neural networks designed to generate data samples, often with a focus on capturing the underlying statistical patterns or structures present in the training data [12]. These networks are particularly instrumental in various applications, including image generation, natural language processing, and data synthesis.

The fundamental principle behind GANs is to learn a probabilistic model of the data distribution from a given dataset. To accomplish this goal, the models employ a distinctive adversarial training mechanism, which involves two primary network components: a generator $gen : \mathbb{Z}^m \rightarrow \mathbb{R}^n$ and a discriminator $dis : \mathbb{R}^n \rightarrow \{0, 1\}$. The generator is responsible for producing synthetic data samples, while the discriminator evaluates these samples to determine whether they are real (from the training data) or fake (generated by the generator). Through a competitive process, the generator continuously improves its ability to produce increasingly convincing data samples, and the discriminator enhances its capacity to distinguish between real and fake data. This adversarial training process often results in the generator becoming proficient at generating data that is difficult to distinguish from authentic data. In this context, it is seeded with random multivariate Gaussian noise $z \in \mathbb{Z}^m$ of user-defined dimension m that is converted to synthetic output by the network. TraVaG applies a GAN architecture to synthesize event data from noise that is similar to the original input.

3.4. Autoencoders

Autoencoders are a class of ANNs designed for unsupervised learning and data compression [26]. They serve a dual purpose; (1) *encoding* input data into a lower-dimensional representation and (2) *decoding* this representation to reconstruct the original data. These networks are instrumental in various fields, including image processing, dimensionality reduction, data denoising, and anomaly detection.

The underlying principle of autoencoders involves two primary components: an encoder $enc : \mathbb{R}^n \rightarrow \mathbb{R}^d$ and a decoder $dec : \mathbb{R}^d \rightarrow \mathbb{R}^n$. The encoder processes some high-dimensional input data $x \in \mathbb{R}^n$ and maps it to a compressed representation, often referred to as a bottleneck or latent space $\mathbb{R}^d$ (typically $d \ll n$). The decoder then takes this compressed representation and attempts to reconstruct the original data. During training, the networks enc and dec aim to minimize the difference between the input data and the reconstructed output, which encourages the autoencoder to capture meaningful features and patterns from the data. We employ the autoencoder principle at TraVaG to learn a reduced encoding of input event data.

3.5. Denoising Diffusion Probabilistic Models (DDPMs)

Inspired by nonequilibrium thermodynamics, DDPMs are a class of generative likelihood-based latent variable models that allow matching hidden data distributions by learning to reverse a gradual, iterative noisifying mechanism [13,27]. Both noisifying (diffusion) and denoising operations are represented by a combination of two distinct Markov chains. In the course of this work, we follow the approach introduced in [22,28] and use multivariate Gaussian noise to represent the processes. Given a data sample $x_0 \in \mathbb{R}^n$ that follows an unknown distribution $x_0 \sim q(x_0)$, we define latent variables of equal dimensionality $x_1 \dots x_T \in \mathbb{R}^n$ through a so-called Markovian *forward process* that adds Gaussian noise at step $t \in \{1 \dots T\}$ with variance $\beta_t \in (0, 1)$ as follows:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t \mathbf{I}). \quad (1)$$

Here, $q(x_t|x_{t-1})$ denotes the conditional probability density distribution of x_t given x_{t-1} , $\mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t \mathbf{I})$ represents the Gaussian distribution of x_t with expectation $\mu = \sqrt{1 - \beta_t}x_{t-1}$ and variance $\sigma^2 = \beta_t \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of dimension n . Considering the joint distribution over all latents $x_1 \dots x_T$ conditional on the data sample x_0 then leads to the product

$$q(x_1, \dots, x_T|x_0) = \prod_{t=1}^T q(x_t|x_{t-1}). \quad (2)$$

When choosing a sufficiently long forward process, i.e., large T and significant variance schedule $\beta_1 \dots \beta_T$, the diffusion chain converges to an isotropic Gaussian distribution of the last variable x_T . As noted in [28], Eq. (3) further allows accessing the distribution of an arbitrary forward step t directly conditioned on x_0 :

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)\mathbf{I}) \quad (3)$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. With standard normally distributed noise ϵ the latent random variable x_t is therefore expressed as $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$. To exploit this diffusion principle for generating artificial data samples that follow the same distribution $q(x_0)$, DDPMs complement the noisifying process with a denoising *reverse* Markov chain that is trained to iteratively remove added noise from transformed latents of the forward process. Starting at Eq. (1) and the same perturbation schedule, if all posterior distributions $q(x_{t-1}|x_t)$ for $t \in 1 \dots T$ were known, we could directly initialize standard-normally distributed latent representations of x_T due to $q(x_T) \rightarrow \mathcal{N}(x_T; 0, \mathbf{I})$ for $T \rightarrow \infty$ and simply reverse the forward process to estimate $q(x_0)$ at $t = 1$. However, since $q(x_{t-1}|x_t)$ depends on the entire unknown data distribution, the true posteriors are also unknown and need to be approximated with the help of training samples. Following the investigations from [27,28], the best results for the reverse process were achieved by learning a parameterized estimator $p_\theta(x_{t-1}|x_t)$:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (4)$$

where mean $\mu_\theta(x_t, t)$ and variance $\Sigma_\theta(x_t, t)$ are represented by deep ANNs with parameters θ .

4. Approach

As outlined in Section 2, DP-based generative ANNs have undergone extensive investigation beyond the realm of process mining. Prominent work has revealed common strategies such as the utilization of variational autoencoder architectures, DDPMs, or the integration of GAN architectures. When transferring these concepts to event data, a pivotal consideration is the management of their intricate high-dimensional structure, which can pose notable challenges during the training process, especially when we introduce noise-based privacy measures into the optimization routines. Consequently, we have chosen to pursue two cutting-edge methodologies that have demonstrated exceptional efficacy in managing extensive feature spaces.

First, we demonstrate the idea of TraVaG [7], which combines the compression functionality of autoencoders with the flexibility of GANs. Instead of directly generating new event logs, TraVaG first learns a compressed encoding and then trains a GAN to reproduce data within the encoded latent space. Final datasets are obtained by decoding back the dimension-reduced intermediate format. This principle mitigates the complication of GANs when extracting statistical properties from feature-rich data that is limited in size. Particularly, sparse features can be compressed without significant loss of information while generator networks improve

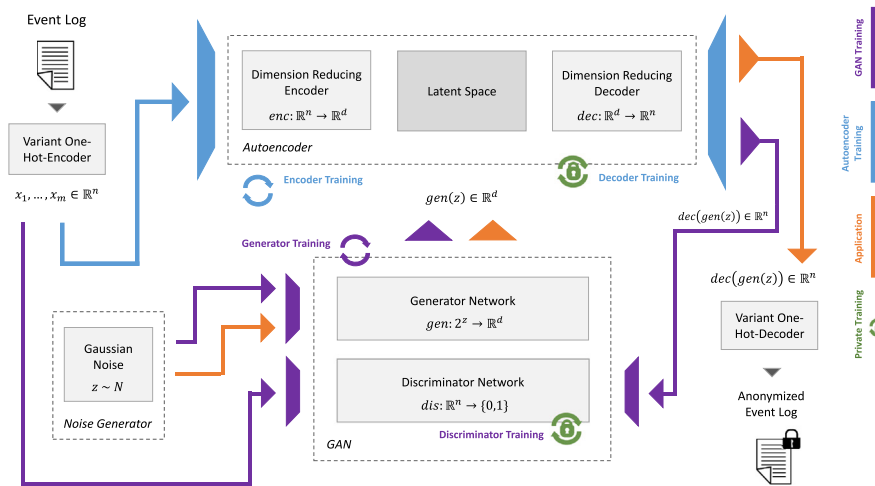


Fig. 1. Simplified workflow of the TraVaG training and application process [7].

their learning performance due to the lower dimension. Moreover, no Gaussian Mixture distribution is enforced on the latent space, as it is the case for typical generative stand-alone autoencoder methods [19].

Second, we adopt the principle of DDPMs as their design resulted in leading generative performance on broad tabular data as well as large and complex images while being fast and stable to train [14,22]. In particular, the self-correction capability due to gradual Markovian perturbation processes allows DDPMs to accurately reconstruct underlying high-dimensional data distributions from training samples. For our work, we train a deep network to directly predict the added noise of the forward process given sample trace variants and corresponding noise scheduling step numbers. The incorporation of differential privacy into the training process then provides enough regularization to guarantee simple architectures with fast convergence.

4.1. Differentially private GANs and autoencoders

The components and workflow of the TraVaG framework are shown in Fig. 1. The process commences with the preprocessing of a basic event log, which contains variant distributions represented as variant-frequency pairs. Here, two common approaches can be taken, each leading to distinct consequences for the final generated outcomes. The first approach involves examining the activities within variants and extracting all subsequences of immediate neighbors, known as Directly-Follows Relations (DFRs). These DFRs are subsequently transformed into either a binary or numerical space, and are then provided to a GAN as a single feature or as two features, inclusive of their respective frequencies. It is essential to note that a drawback of this method is that the generator essentially acts as a sequence constructor, which enables the creation of fake, non-existing variants in the postprocessing phase where all generated activity pairs are interconnected again.

To prevent the generation of such spurious trace variants, we opt for the second alternative, which entails exclusively considering complete variants as input. Thus, a basic event log denoted as L featuring n variants and m cases is encoded in binary form as follows. In a matrix of dimensions $m \times n$, each variant type corresponds to a binary feature column, while each case corresponds to a row instance. A value of 1 appears in the respective variant column for a given case, with 0s occupying all other positions, thus forming a sparse matrix. Importantly, this transformation can be seamlessly reversed to revert to the original data space after the generation process. As a result, in contrast to prefix-based methodologies, TraVaG consistently avoids the generation of fake trace variants. Moreover, the one-hot encoding method does not introduce any alterations to the data statistics, thereby incurring no associated privacy costs. As commonly standardized, we refer to this preprocessing procedure as one-hot encoding and one-hot decoding (see Variant One-Hot-Encoder, Variant One-Hot-Decoder in Fig. 1).

Our training process consists of two primary phases: autoencoder training (blue parts) and GAN training (purple parts). In the following, we provide a broad overview of each training element. Given the central emphasis of our work on privacy considerations, we particularly focus on an in-depth exposition of the privately trained components. For an exhaustive algorithmic breakdown encompassing the network structures, parameter optimization, activation functions, loss metrics, and optimization techniques, we direct interested readers to consult our supplementary documentation, which is accessible on GitHub.¹

Following the preprocessing, all sparse binary variant vectors $x_1 \dots x_m \in \mathbb{R}^n$ are first directed to the autoencoder training phase, i.e. to both encoder and decoder ANNs. These components serve the purpose of converting the high-dimensional data $x_i \in \mathbb{R}^n$ into a more compact representation, the latent space ($\mathbb{R}^d$, $d \ll n$), and vice versa. It is important to note that the dimension d is a hyperparameter of the autoencoder, and its selection is contingent upon the configuration of the GAN. Also, the encoder and

¹ https://github.com/wangelik/TraVaGen/blob/main/supplementary/TraVaG_Supplementary.pdf.

decoder are trained differently. As the encoder does not influence the GAN training process or the generation of new event data, there is no need for it to undergo private optimization, as discussed in [18,29], and [19]. Conversely, the decoder plays a significant role in the anonymization process and is made available to the public. Hence, the corresponding training is carried out with privacy preservation through the utilization of Differentially Private Stochastic Gradient Descent (DP-SGD). Further insights w.r.t. DP-SGD can be found in Section 4.3.

In the next step, the same one-hot encoded data $x_1 \dots x_m \in \mathbb{R}^n$ are used to train a GAN consisting of two feed-forward ANNs; a generator $gen : 2^{\mathbb{Z}} \rightarrow \mathbb{R}^d$ and a discriminator $dis : \mathbb{R}^n \rightarrow \{0,1\}$. It is important to highlight that the primary objective of the generator, denoted as gen , is to generate synthetic data within the output space $\mathbb{R}^d$, which closely resembles the compressed variants. To achieve this, it is initialized with a random multivariate Gaussian noise vector z of user-defined dimension. On the other hand, the discriminator, labeled as dis , is tasked with distinguishing whether its input comes from the decompressed output of the generator $dec(gen(\cdot))$ (classified as fake and assigned to 0), or if it originates from the original data source $x_i, i = 1 \dots m$ (categorized as real and assigned to 1).

Both generator and discriminator components are defined by their network weights and are subjected to an iterative training process in which they engage in a competitive dynamic. The generator's goal is to produce latent space outputs that closely resemble real encoded data, making it challenging for the discriminator to distinguish between fake and real data. On the other hand, the discriminator endeavors to reveal synthetic data records. Over time, this competitive interplay enables the generator to acquire an understanding of the data and capture the statistical characteristics of the input variant distribution through the perspective of the autoencoder. It is important to emphasize that due to the integrated autoencoder, the generator exclusively focuses on the latent space $\mathbb{R}^d$, which is notably easier to model in comparison to the intricate data space $\mathbb{R}^n$. Furthermore, this approach effectively prevents the generator from accessing the actual confidential data space. As a result, it does not require training with DP measures, in contrast to the discriminator, which is again privately optimized using DP-SGD algorithms [21].

Finally, after completing the training of both the autoencoder and GAN, TraVaG is ready to be employed for the generation of novel synthetic anonymized event data (orange parts). The fundamental sampling mechanism mirrors the training phase of the generator. It commences with the generation of a random Gaussian noise sample z . This noise is then processed by the generator, producing the output $gen(z)$. From the latent space, the decoder maps this output to $dec(gen(z))$ within the binary data space. Ultimately, the synthetic one-hot encoded result is transformed back into the realm of variant representations. At this stage, one of the compelling advantages of TraVaG becomes evident in the data format it operates on. Given that the feature space already embodies the various variants present in the original data, TraVaG treats these variants as fixed and merely focuses on learning their distribution during the training process. Consequently, when the framework is put into practice, it can consistently reconstruct an anonymized version of this distribution through multiple iterations, all without the need to create new fake variants.

Generally, the greater the number of synthetic data instances generated, the more refined the resulting TraVaG output becomes. In other words, the newly created anonymized variants progressively approach the distribution of the original variants. Note that this process does not lead to convergence with the actual variant frequencies but rather converges to the internal, learned anonymous representation within TraVaG. Thus, it is advisable to run TraVaG at least as many times as there are cases in the original event log. In situations where smaller privatized datasets are required, the generated output can be down-sampled during postprocessing rounds.

4.2. Differentially private denoising diffusion probabilistic models

The workflow and components of our differentially private DDPM process are shown in Fig. 2. Similar to the preprocessing in Sub Section 4.1 all trace variants are initially one-hot-encoded to force the model to only pick up statistical properties of true variants. Within the DDPM forward process, we uniformly sample m step numbers from the range $1 \dots T$, digest the encoded original variants $x_{01} \dots x_{0m} \in \mathbb{R}^n$ and generate corresponding Gaussian latents $x_{ti} \in \mathbb{R}^n$ for $t \in \{1 \dots T\}, i \in \{1 \dots m\}$ according to Eq. (3) (gray components). The outputs of this process are thus triples consisting of step number, latent sample, and added Gaussian noise.

To estimate the posterior according to Eq. (4) and reverse the diffusion process during sampling, we follow the convention in [27] that showed remarkable DDPM performance when fixing $\Sigma_\theta(x_t, t)$ to $\tilde{\beta}_t = \beta_t(1 - \tilde{\alpha}_{t-1})/(1 - \tilde{\alpha}_t)$ and only learning the distribution mean $\mu_\theta(x_t, t)$. Using Eq. (3) and Bayes theorem, the parameterized mean estimator can be rewritten as

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \tilde{\alpha}_t}} \epsilon_\theta(x_t, t) \right) \quad (5)$$

where $\epsilon_\theta(x_t, t) \in \mathbb{R}^n$ denotes an estimate of the added noise onto x_0 at step t . Given the training triples of latent variants, true added noise and step number, we train a deep ANN to predict $\epsilon_\theta(x_t, t)$ that has the same dimension as the latent and original encoded variants (purple components). In this process, the step number is first embedded via so-called sinusoidal time embedding blocks [30], concatenated to the latent data, and then forwarded to an expanding ANN layer. Finally, the hidden networks are reduced back to the desired noise dimension. Since the training cycles for predicting $\epsilon_\theta(x_t, t)$ demand repeated access to original variant data, we optimize our ANNs with DP-SGD instead of classical gradient updates similar to the TraVaG framework in Section 4.1. As a result, all subsequent actions of the DDPM reverse process are differentially private and allow synthetic sampling according to an anonymized internal estimation of the true data distribution.

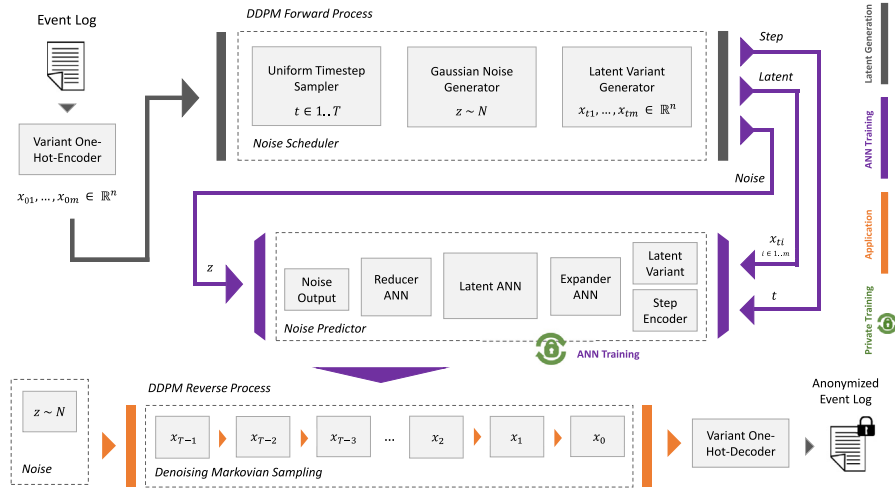


Fig. 2. Simplified workflow of the DDPM training and application process.

To eventually start generating new variants from the noise we begin with initializing x_T based on a standard normal distribution $\mathcal{N}(x_T, 0, \mathbf{I})$. For all steps t from $T-1$ to 1, the Markovian reverse procedure then iteratively samples denoised latents based on Eq. (6) until an approximation of x_0 is reached [27].

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \alpha_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z \quad (6)$$

Here, $\epsilon_\theta(x_t, t)$ represents the predicted output from our trained DDPM ANN, $\sigma_t = \bar{\beta}_t$ and $z \in \mathbb{R}^n$ denotes a standard normally distributed sample. At the end of the sampling routine, all synthetic data are analogously decoded back into their original format. Again, it is recommended to generate at least as many new variants as there used to be in the original dataset to best reconstruct the internally learned estimation of the true data distribution.

4.3. Differentially Private - Stochastic Gradient Descent (DP-SGD)

To introduce DP to classical Stochastic Gradient Descent (SGD), Abadi et al. outlined the following two key steps in their work [17]. Given a dataset $X = \{x_i \in \mathbb{R}^n \mid 1 \leq i \leq m\}$, f as a loss function, and θ as the model parameter. First, the gradient $g_i = \nabla_\theta f_\theta(x_i)$ of each data sample x_i is clipped at some real value $C \in \mathbb{R}_{>0}$ to ensure its L^2 -norm of the gradient does not exceed the clipping value. For our work, we make use of the following clipping function²:

$$\text{clip}(g_i, C) := g_i \cdot \min(1, C/\|g_i\|_2). \quad (7)$$

Then, as depicted by Eq. (8), multivariate Gaussian noise parameterized by a noise multiplier $\Phi \in \mathbb{R}$ is added to the clipped gradient vectors before averaging over the batch $B \subseteq X$. Note that we denote the identity matrix as $\mathbf{I}$ and the Gaussian distribution of unspecified dimension as $\mathcal{N}$.

$$g_B \leftarrow \frac{1}{|B|} \left(\sum_{i \in B} \text{clip}(\nabla_\theta f_\theta(x_i), C) + \mathcal{N}(0, C^2 \Phi^2 \mathbf{I}) \right) \quad (8)$$

The noisyfied, clipped and averaged gradient g_B is now differentially private and can be used for conventional descent steps: $\theta \leftarrow \theta - \eta \cdot g_B$, where η is the so-called learning rate. Note that clipping the single gradients as in Eq. (8) can also be replaced by instead clipping gradients of groups of more data points, so-called *microbatches* [16]. Here, the initial batch B is, therefore, further partitioned into new batches $B_1, \dots, B_k \subseteq B$ each of size r (skipping the dividend). We then obtain the new microbatch-related gradient, as shown in Eq. (9).

$$g_B \leftarrow \frac{1}{k} \left(\sum_{i=1}^k \text{clip}(\nabla_\theta f_\theta(X_{B_i}), C) + \mathcal{N}(0, C^2 \Phi^2 \mathbf{I}) \right). \quad (9)$$

Naturally, standard differentially private SGD (DP-SGD) corresponds to setting $r = 1$. Increasing the value of r while keeping the size of the batch $|B|$ constant primarily results in decreased training runtime and a reduction in the attained training accuracy. Furthermore, it has been demonstrated to not have a substantial impact on privacy, especially for large datasets [16].

In contrast to the conventional DP parameters ϵ and δ , DP-SGD employs the noise multiplier Φ as a control parameter. When transitioning between these two settings, recent research has unveiled a more stringent privacy bound when the batch sampling

² Note that also other clipping strategies exist, as highlighted in [16].

process for B adheres to a particular Poisson schedule [17]. This procedure individually selects each data point from the dataset X with a constant probability denoted as q , often referred to as the sampling rate.

4.4. Privacy accounting

To assess and track the precise level of privacy offered by DP-SGD algorithms, we utilize a concept known as *Renyi Differential Privacy* (RDP) [31]. RDP represents a distinct notion of differential privacy, primarily employed in the context of private optimization. The underlying mathematical principle is the so-called *Renyi divergence*. Given two probability distributions P and Q , the Renyi divergence of order α is defined as follows.

$$D_\alpha(P \parallel Q) := \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim Q} \left(\frac{P(x)}{Q(x)} \right)^\alpha \quad (10)$$

Definition 3 ((α, ϵ) -RDP for Event Logs). Let L_1 and L_2 be two neighboring event logs that differ only in a single entry, e.g., $L_2 = L_1 \uplus [\sigma]$ for any $\sigma \in \mathcal{A}^*$. Given $\alpha > 1$ and $\epsilon \in \mathbb{R}_{>0}$, a randomized mechanism $\mathcal{M}_{\alpha, \epsilon} : \mathcal{L} \rightarrow \mathcal{L}$ provides (α, ϵ) -RDP if $D_\alpha(\mathcal{M}(L) \parallel \mathcal{M}(L')) \leq \epsilon$.

Considering two RDP mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$, we further denote a composition principle as follows [31].

Proposition 1 (Composition of RDP). If $\mathcal{M}_1$ and $\mathcal{M}_2$ are two (α, ϵ_1) -RDP and (α, ϵ_2) -RDP mechanisms for $\alpha > 1$, respectively. Then, the composition of $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfies $(\alpha, \epsilon_1 + \epsilon_2)$ -RDP.

Due to the conceptual similarity between (α, ϵ) -RDP and (ϵ, δ) -DP, the corresponding privacy parameters can be converted [31].

Proposition 2 (RDP Parameter Conversion). If a mechanism $\mathcal{M}$ satisfies (α, ϵ) -RDP with $\alpha > 1$, then for all $\delta > 0$, $\mathcal{M}$ also satisfies $(\epsilon + (\log 1/\delta)/(\alpha - 1), \delta)$ -DP.

An advantage of using the concept of Renyi divergence during an iterative execution of Gaussian mechanisms, such as for DP-SGD, is that it provides a tighter bound on the privacy loss than standard (ϵ, δ) -DP composition. To calculate the final (ϵ, δ) -DP parameters from multiple runs of RDP-based DP-SGD, a sequence of three steps is required. Contingent on the chosen sampling strategy, first, a so-called *subsampling Renyi divergence* needs to be derived. Subsequently, privacy levels are aggregated within the framework of RDP before being transformed back into conventional DP.

1. **Subsampled Renyi Divergence.** Given a sampling rate q and noise multiplier Φ , the RDP privacy parameters for one iteration of DP-SGD can be derived as a non-explicit integral function of $\alpha \geq 1$ [31]. This function is standardized in many privacy-related optimization packages and will be referred to as $\text{RDP}_1(q, \Phi)$ [17].
2. **RDP Composition.** Since DP-SGD is most likely to run iteratively, we need to compose Step 1 over all executions according to Proposition 1. Hence, the resulting RDP parameters of T iterations are obtained by computing $\text{RDP}_T(q, \Phi, T) := \text{RDP}_1(q, \Phi) \cdot T$.
3. **Conversion to (ϵ, δ) -DP.** After retrieving an expression for the overall RDP privacy parameters with RDP_T , we need to convert the respective (α, ϵ) tuple to a (ϵ, δ) guarantee according to Proposition 2. Note that since the ϵ parameter of RDP is also a function of α , Step 3 involves optimizing for α to achieve a minimal ϵ and δ .

In the context of our privatized DDPM and TraVaG training algorithms (see Section 4) we employ this accounting procedure to obtain the respective (ϵ, δ) -DP guarantees for both DDPM ANNs and autoencoder as well as GAN-based discriminator. In the case of TraVaG, the resulting values are then combined into a final privacy cost by the composition theorem of DP [2] (see Theorem 1).

Theorem 1 ((ϵ, δ) -DP Mechanism Composition for Event Logs). Let L be a simple log, and $\mathcal{M}_i, 1 \leq i \leq n$ be (ϵ_i, δ_i) -DP mechanisms. The sequential application of these mechanisms on arbitrary sublogs of L leads to an overall worst-case privacy level parameterized by $(\sum_{1 \leq i \leq n} \epsilon_i, \sum_{1 \leq i \leq n} \delta_i)$. If each $\mathcal{M}_i$ operates on strictly disjoint sublogs of L , the worst-case privacy level is $(\max_{1 \leq i \leq n} \epsilon_i, \max_{1 \leq i \leq n} \delta_i)$, so-called *parallel composition*.

As Theorem 1 states, different (ϵ, δ) -DP mechanisms can be easily combined into more complex algorithms at the cost of a directly measurable cumulative privacy loss. Nevertheless, the result still promises (ϵ, δ) -DP independent of the exact form of composition or query structure.

5. Experiments

Our experimental evaluation encompasses a broad spectrum of the key privacy parameters, $\epsilon \in \{0.001, 0.01, 0.1, 1, 2\}$ and $\delta \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 0.01\}$. These parameter ranges have been chosen in alignment with typical values utilized in industrial applications and in accordance with contemporary research in the field of DP [5,8,10,32]. It is worth emphasizing that we have deliberately included extreme settings, such as $\epsilon = 2$ and $\delta = 0.01$, not because they are practically relevant, but to showcase how the anonymization methods perform when initiated from a weak or non-private baseline.³

³ Generally, δ is recommended to be not larger than $1/|D|$, where $|D|$ is the size of dataset D [2].

Table 2
General statistics of the event logs used in our experiments.

Event log	#Events	#Cases	#Activities	#Variants	Trace uniqueness
Sepsis	15 214	1050	16	846	80%
BPIC-2013	65 533	7554	4	1511	20%
BPIC-2012-App	60 849	13 087	10	17	0.12%

Given the inherent probabilistic nature of (ϵ, δ) -DP, we execute the TraVaG and DDPM generators 100 times across all input event logs and privacy parameter combinations. Subsequently, we report the average results, as the remaining training-induced standard deviation is small enough to validate all systematic trends. A more detailed study of the generator variance is uploaded to GitHub.⁴ For comparison, we assess our findings against TraVaS, an established state-of-the-art technique [11], and the original prefix-based framework, denoted as the *benchmark* [5]. Note that in [11], TraVaS was already compared against SaCoFa [8] and the benchmark approach from [5], and exhibited superior performance. In this work, we have included the benchmark method to facilitate straightforward comparisons. Additionally, it is important to mention that Libra [10] does not accept ϵ as an input parameter but instead computes it based on α as an RDP parameter and the applied sampling strategy. This aspect complicates direct comparisons based on exact ϵ and δ parameters. Nevertheless, one notable observation, in contrast to our generative models, is that Libra returns an empty log for event datasets with numerous infrequent variants (such as Sepsis) when $\delta \leq 10^{-3}$.

The configuration of the ANNs in our generative models is based on a semi-automated tuning approach tailored to the specific input logs. While many design choices and hyperparameters are adjusted based on results from manual testing and research experience, certain parameters, including the *batch size* (B), the *number of iterations* (I), and the *noise multiplier* (Φ), are automatically optimized using a grid-search methodology for fixed privacy levels [33]. A comprehensive list of all resulting configurations for each event log can be found on GitHub.⁵

5.1. Datasets

We investigate the algorithm performance using real-life event data. For this purpose, three event logs with varying sizes and levels of trace uniqueness have been selected. As previously discussed in Section 1 and highlighted in other research papers such as [5,8,34], and [10], privatizing infrequent variants can be particularly challenging. Hence, trace uniqueness serves as a crucial metric for our analysis. The first dataset, known as the Sepsis log, documents hospital processes for Sepsis patients and is notable for containing numerous rare traces [35]. In contrast, the BPIC-2013 dataset encompasses a significantly larger number of cases but also exhibits a trace uniqueness that is four times smaller compared to the Sepsis log. BPIC-2013 pertains to an incident and problem management system known as VINST [36]. Lastly, the BPIC-2012-App log from [37] reports process data associated with various loan applications from a Dutch financial institution. This dataset offers lower-dimensional entries with relatively small trace uniqueness. Note that in the experimental verification, our focus is on data with diverging variant distributions and not primarily large sizes. As a result, big, yet, at the same time, similar-in-shape event logs such as the *Road Traffic Management* dataset [38] are not considered. With 150370 traces over 231 variants, *Road Traffic Management* approximately represents a scaled version of the BPIC-2012-App log, and its analysis would provide insights into the resource-dependent training time rather than on the algorithm's capabilities of picking up complex frequency distributions. Moreover, it is important to underline that all of these logs are authentic examples of confidential human-centric data where the case identifiers are linked to individuals. For more detailed log statistics, we refer to Table 2.

5.2. Evaluation measures

To assess the effectiveness of a (ϵ, δ) -Differential Privacy (DP) mechanism in preserving the utility of data or results, it is crucial to utilize suitable evaluation metrics. The perspective of *data utility* involves assessing the resemblance between two logs, irrespective of their potential future applications. To compute data utility, we rely on specific measures, including *relative log similarity* [11,39] and *absolute log difference* [7,11].

The *relative log similarity* metric assesses the *earth mover's distance* between two distributions of trace variants. It employs the normalized Levenshtein string edit distance as the similarity function to measure the resemblance between trace variants. Consequently, this metric quantifies how closely the variant distribution in an anonymized log aligns with the original variant distribution, with values ranging from 0 to 1.

Absolute log difference accounts for the situations where distribution-based measures provide misleading expressiveness [11]. An instance of this is when event logs exhibit similar variant distributions but significantly differ in size. To calculate the *absolute log difference* value, we adopt the methodology introduced in [11]. First, it involves transforming the input logs into a *bipartite graph* where variants are treated as vertices. Subsequently, a *cost network flow* problem is solved, with demands and supplies being determined based on the absolute variant frequencies. In this context, the associated edge costs are determined by the absolute

⁴ https://github.com/wangelik/TraVaGen/blob/main/supplementary/Uncertainty_Supplementary.pdf.

⁵ <https://github.com/wangelik/TraVaGen/tree/main/supplementary>.

Levenshtein distance between variants. As a result, the outcome of this optimization signifies the minimum number of Levenshtein operations necessary to convert variants in an anonymized log into variants found in the original log. More detailed documentation on the exact algorithms is provided on GitHub.⁶

In addition, we analyze the performance of our methods with respect to *result utility preservation* specifically in the context of *process discovery*, which is a specialized application relying on trace variant distributions. To conduct this assessment, we employ the *inductive miner infrequent* algorithm [40], setting a default noise threshold of 20% to derive process models from the anonymized event logs for all the investigated (ϵ, δ) settings. Then, these resulting models are compared with the original event log to compute token-based replay scores for *fitness* and *precision*, as outlined in [4].

5.3. Data utility analysis

In this subsection, the results of the two aforementioned data utility metrics are presented for all three real-life event logs. Fig. 3 shows the average results on BPIC-2013 in an eight-fold heatmap. The gray fields denote an unsuccessful algorithm execution. For $\delta < 10^{-3}$, the thresholding of TraVaS becomes too strict and removes many variants in the anonymized outputs. On the contrary, the benchmark introduces artificial variants and noise to an extent that is unfeasible to average within reasonable time and accuracy. In opposition, our novel DDPM and TraVaG approaches successfully manage to generate anonymized outputs for $\delta < 10^{-3}$. Due to more stable training and less noise introduction, the diffusion principle even allows working in the high δ -regime for $\epsilon = 0.001$.

More importantly, both DDPM and TraVaG results of *relative log similarity* and *absolute log difference* do not illustrate clear decreasing trends on lower δ within the investigated parameter range. We explain this expected observation by the fact that our trained generative models avoid any pruning mechanism on their output and implement less δ -dependent Gaussian noise via RDP into the gradients (see Section 4.4 and [31]). Whereas the absolute log difference results maintain a rather stable output for the different (ϵ, δ) values, the relative log similarity of TraVaG presents a strong positive ϵ -dependency. As a result, the absolute statistics (absolute Levenshtein distances and absolute frequencies) of the anonymized event data seem to be more similar to the original logs as the variant distributions with increasing noise. A rationale for this discrepancy lies in the comparably small dataset with 7554 instances over 1511 variants. By construction, TraVaG accomplishes reproducing equally sized event logs containing many original variants but fails to pick up some characteristics of the underlying distribution once the input data or the training iterations are limited. Hence, we expect this diverging trend to diminish with increasing training data. Our results from using DDPM demonstrate consistent performance across both evaluation metrics. Notably, DDPM outperforms TraVaG in terms of *absolute log difference*, although it shows a slightly lower performance in *relative log similarity* in the higher ϵ -regime. This observation resembles the different use cases and trade-offs provided by the different model architectures. As our DDPM framework comprises more complex ANNs that adopt high-dimensional variant distributions less accurately with limited training data than TraVaG, the model releases lower relative similarity scores for weaker privacy settings. However, when the privacy guarantees are increased, their iterative processes lead to self-correction of introduced noise that compensates for further performance decrease and keeps the results more invariant.

The data utility results for the Sepsis log are presented in Fig. 4. With only 1050 instances at 846 variants, this dataset is even smaller and more trace-unique than BPIC-2013. Nevertheless, the overall dimensionality shows a significant reduction. As a result, we observe similar, but more pronounced behavior of *relative log similarity* and *absolute log difference* metrics compared to Fig. 3. Extreme examples are the metrics at $\epsilon < 0.01, \delta < 10^{-2}$, where the introduced gradient noise turned out as too intense for the TraVaG model to converge under the given training data size. In contrast, the DDPM that only integrates one privately trained ANN compound and employs gradual noise perturbation, again successfully works even at the low ϵ -regime. For the remaining privacy settings, both DDPM and TraVaG outperform their competitors with respect to the absolute log statistics, while the relative log similarity shows leading performance for DDPM and similar results for TraVaG compared to TraVaS and the prefix-based benchmark. The main cause is rooted in the dataset structure where the lower dimensionality is better manageable for the DDPM architecture so that its gradual denoising advantage leads to stable and unmatched log similarities.

Last but not least, Fig. 5 depicts the performance evaluation of DDPM, TraVaG, TraVaS, and the benchmark on the BPIC-2012-App event log comprising 13087 samples over 17 unique variants. In contrast to Sepsis and BPIC-2013, both *relative log similarity* and *absolute log difference* for DDPM and TraVaG now indicate a slightly increasing data quality with increasing ϵ . Combined with the generally superior metric scores in the less strict privacy regime, this pattern can be understood by a better-learned variant distribution due to the larger training input. Interestingly, DDPM underperforms our TraVaG model for $\epsilon < 1$ and outperforms at $\epsilon > 0.1$. Due to the fundamental differences in the training principle, we assume the gradual denoising chain of the DDPM framework to be suboptimal at strong DP in the event of balanced, low-dimensional data distributions compared to the direct sampling ANN of TraVaG. In addition, despite the log-related performance boost, we notice a considerable underperformance with respect to TraVaS and benchmark at $\delta > 10^{-4}$. Since both TraVaS and the benchmark release variant data by noise addition and thresholding, large event logs with many frequent traces (such as BPIC-2012-App) are hardly affected as long as the threshold is lower than the lowest frequency. On the contrary, our generative models still have to privately learn the underlying data distribution during multiple training iterations, which is more prone to errors and slight deviations. Nevertheless, we note that the general ability of DDPM and TraVaG to capture relevant data characteristics also significantly increases with more frequent variants (see Fig. 5), particularly if enough training data is available.

⁶ https://github.com/wangelik/TraVaGen/blob/main/supplementary/Metrics_Supplementary.pdf.

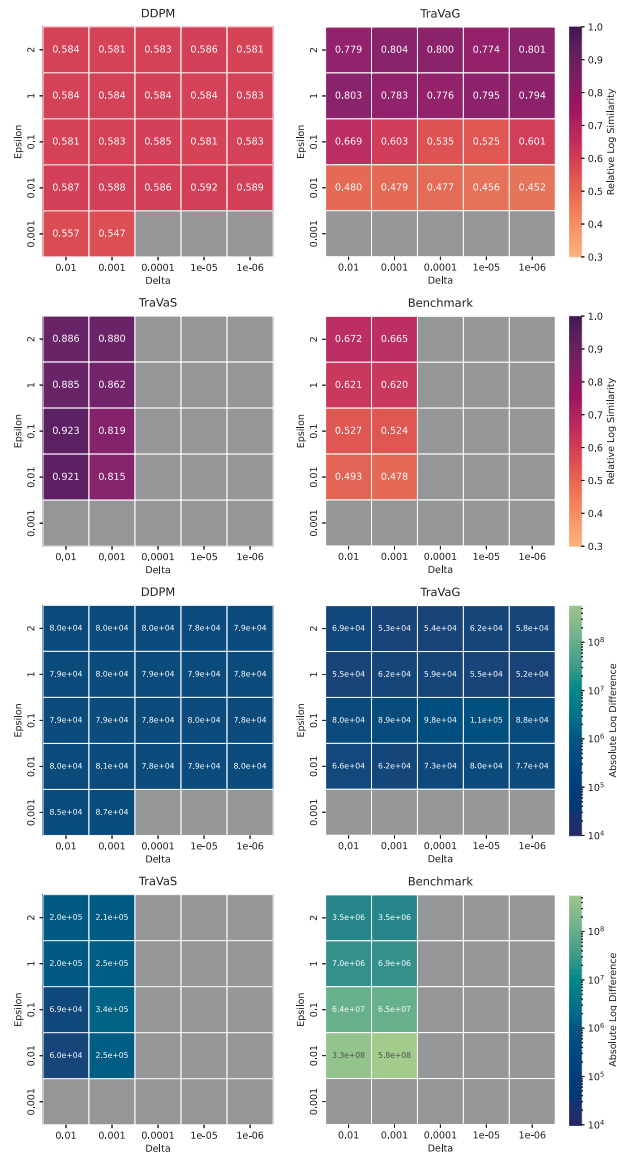


Fig. 3. The relative log similarity and absolute log difference results of anonymized BPIC-2013 logs generated by DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

5.4. Process discovery analysis

Our data utility analysis is complemented with a process discovery evaluation based on *fitness* and *precision* scores. Fig. 6 illustrates the result utility analysis of DDPM, TraVaG, TraVaS, and the benchmark on the BPIC-2013 log. As discussed in Section 5.3, both generative models successfully manage to produce results for small $\delta < 10^{-3}$ where the other methods are not applicable. Except for the three outliers at $\epsilon = 0.1$, both fitness and precision show a stable distribution without considerable dependence on the different privacy parameters and with slight outperformance of TraVaG. We thus conclude that the absolute log difference provides a better proxy for the process-discovery-based performance of DDPM and TraVaG models than relative log similarity. Similarly, the strong scores on both metrics demonstrate a sufficient replay behavior between the model obtained from an anonymized log and the original log. Whereas fitness denotes that the process model still captures most of the real underlying event data, precision depicts only a small fraction of model decisions, not being included in the original event log. Consequently, TraVaG and DDPM accomplish learning the most important facets of the BPIC-2013 variant distribution for the discovery algorithm to produce a fitted model.

The result utility evaluation of the high trace-unique Sepsis log is presented in Fig. 7. With respect to fitness, TraVaG shows similar values as TraVaS but a slight under-performance compared to the benchmark. The main cause for this observation again refers to the infrequent variants and the small log size. While TraVaS maintains a strong δ -related threshold and TraVaG copes with

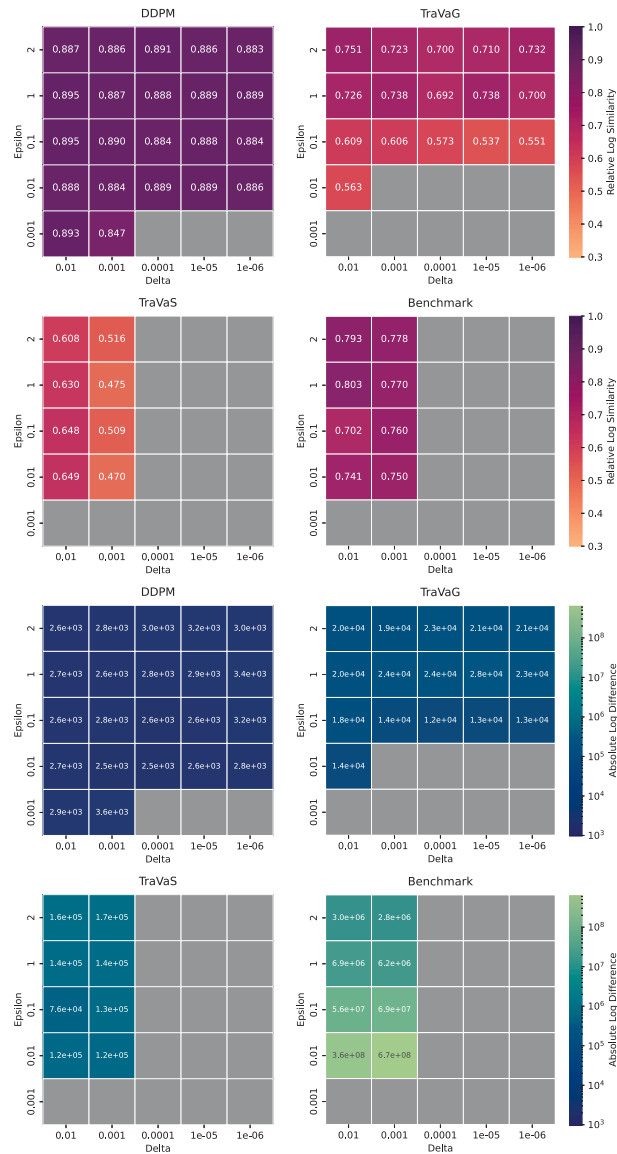


Fig. 4. The relative log similarity and absolute log difference results of anonymized Sepsis logs generated by DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

the limited training data, the benchmark introduces many artificial variants but tends to match the frequent traces. As a result, the discovered process models are able to replay most of the original behavior, in contrast to TraVaG and TraVaS results. According to the aforementioned explanation, precision reflects an inverted trend. Here, the process models resulting from the anonymized event logs using the benchmark technique contain many possible behaviors that are nonexistent in the underlying event log. For TraVaS and TraVaG, we thus achieve more precise anonymized process models. Interestingly, our DDPM framework shows slightly worse fitness and precision performance than TraVaG, despite being superior in the data utility analysis (see Fig. 4). We explain this trend by the fact that the DDPM generator tends to oversample infrequent features and under-represent distribution peaks with limited training input more than TraVaG. Whereas activity-based utility scores optimize and measure over the entire variant distribution and are therefore rather invariant to slight deviations, noise-thresholded process discovery algorithms can lead to more pronounced effects. These disparities are anticipated to diminish as more training data is employed for DDPM.

Fig. 8 demonstrates fitness and precision for all anonymization techniques on the BPIC-2012-App event log. We first highlight the almost constant values of TraVaG and TraVaS which underline that the different noise levels per (ϵ, δ) setting of the privatized outputs do not leak through the noise-thresholded process discovery algorithm. This observation can be traced back to Fig. 5, where the different data utility metrics still show (ϵ, δ) variations as expected. In particular, the slight under-performance of TraVaG compared to TraVaS does not seem to be relevant in the context of anonymized process discovery. On the contrary, the DDPM results reflect

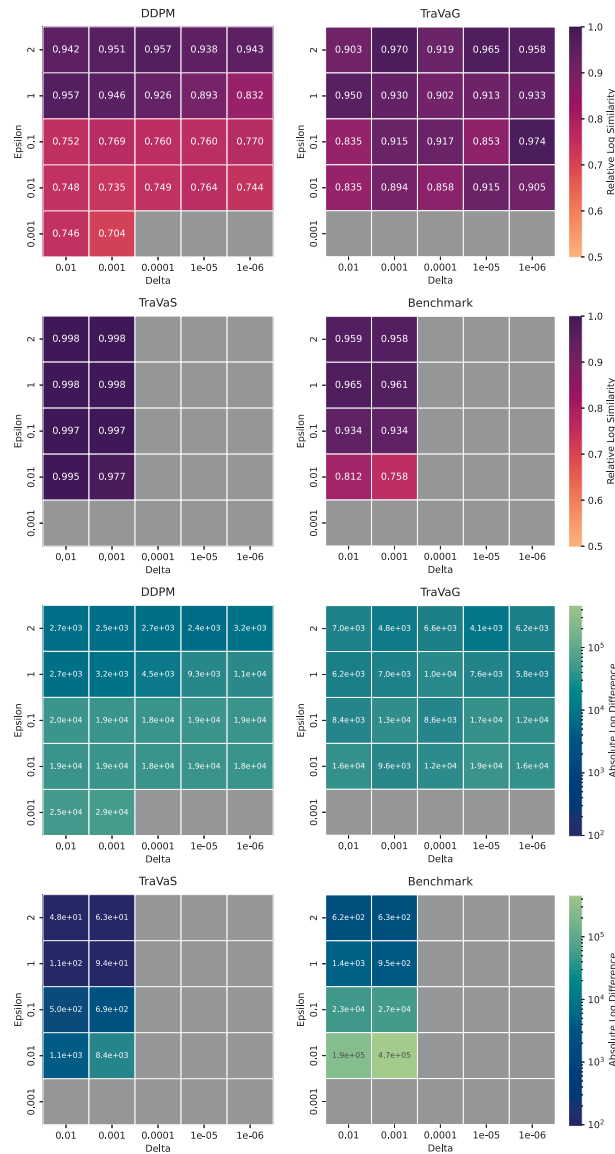


Fig. 5. The relative log similarity and absolute log difference results of anonymized BPIC-2012-App logs generated by DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

the data utility variation and its explanation from Fig. 5, where the gradual sampling technique turned out to be inferior to our GAN setup at the low ϵ -regime. Accordingly, the discovered process models show less accurate replay behavior of the original log (lower fitness) as some more infrequent variants were undersampled during the generation. In direct correspondence, the slightly reduced, privatized models lead to fewer unmatched behavior and thus better overall precision. Finally, we briefly note the three artifacts at $(\epsilon = 0.1, \delta = 10^{-5})$, $(\epsilon = 0.01, \delta = 0.01)$ and $(\epsilon = 0.01, \delta = 10^{-5})$ for TraVaG. Although the resulting scores are still in a similar range as the remaining values, the increased precision at a decreased fitness leads to the assumption of some missing parts within the corresponding privatized models. Such events may occasionally occur if TraVaG fails to learn a specific frequent variant from the underlying data. In fact, the explanation is based on the same assumption that supports the performance variation in the DDPM results.

6. Discussion

The subsequent discussion expands upon the findings derived from the analysis conducted in Sections 4 and 5, focusing on delineating various structural, operational, and technical attributes of the TraVaG and DDPM algorithms in greater depth. First, we

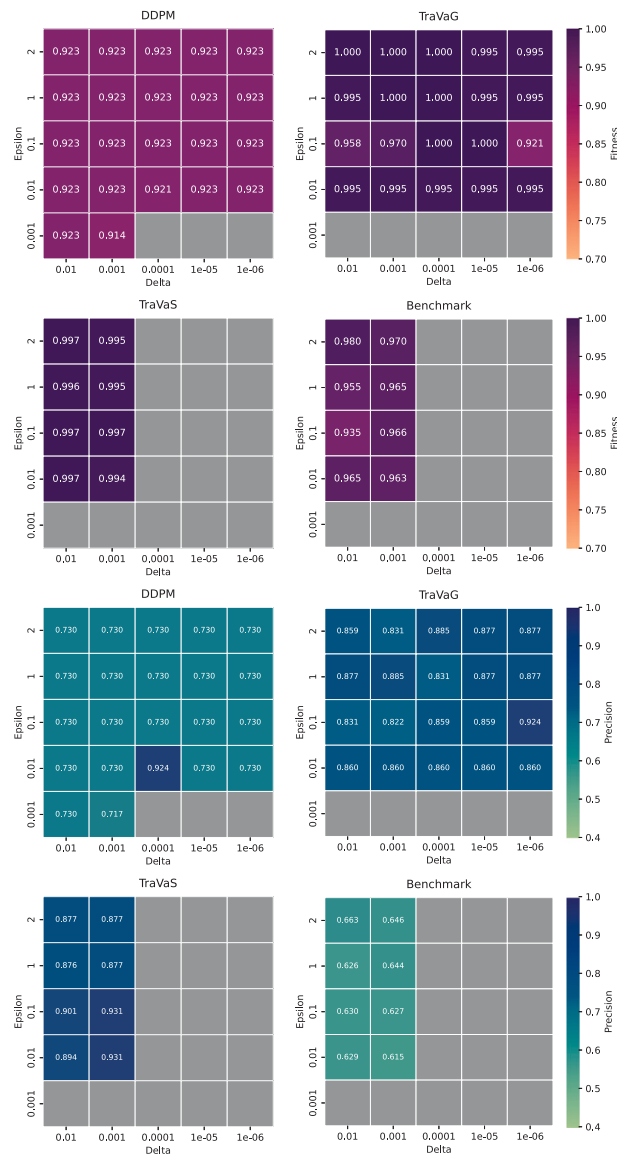


Fig. 6. The *fitness* and *precision* results of anonymized BPIC-2013 event logs generated using DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

discuss the privacy limitations induced by our DP framework. Then, we describe a comparative assessment of complexity from both training and application perspectives. Lastly, elucidation is provided regarding the specification of input data requirements.

We opted for DP as an anonymization technique due to its mathematical privacy notion plus its security, quantifiability, and widespread adoption in industrial contexts. However, alongside these advantages, there exist some drawbacks and limitations. These include the intricate and less intuitive formalism of DP, limited applicability within algorithms, and challenges in interpreting its parameters (ϵ , δ) [2]. Despite dedicated efforts in research focusing specifically on elucidating and tuning DP, achieving transparency in DP-based data protection for uninformed users remains a persistent challenge. Furthermore, it is important to highlight that our data format, particularly in terms of variant frequencies, results in DP protecting individuals who contribute to specific variants through their recorded cases instead of the variants themselves. Given the probabilistic nature of DP and the threshold-independent sampling utilized by our generative models, there may arise privacy-critical scenarios that warrant attention and may hold practical significance. As an illustration, we consider an informed attack model where an adversary possesses access to a trained TraVaG or DDPM generator and is aware that the appearance of a particular variant implies association with cases of a specific individual. Since our generative models retain only true variants without truncating low frequencies, during training, there exists a probability of capturing this variant if it is present in the input event log. In turn, the adversary can exploit upon its occurrence in the sampling process to infer the specific individual. Nevertheless, without detailed domain knowledge, individual cases are still protected by

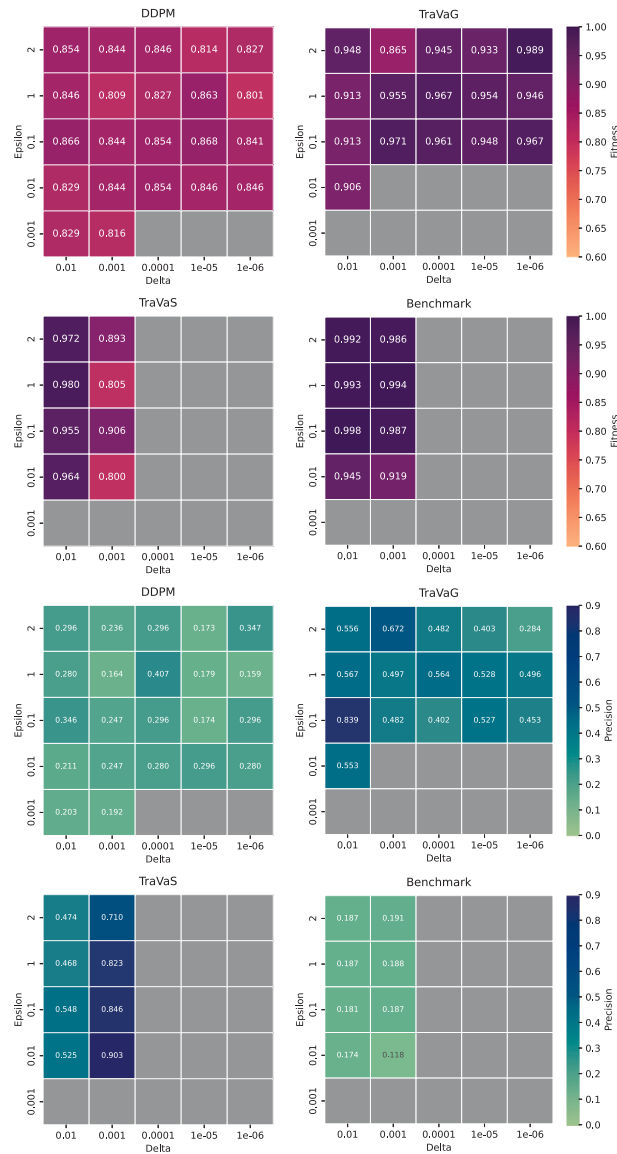


Fig. 7. The *fitness* and *precision* results of anonymized Sepsis event logs generated using DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

both, the aggregation within variant distributions and the noise insertion of DP, despite the design principle of models exclusively releasing true variants.

When analyzing and comparing model complexity for TraVaS, TraVaG, and DDPM algorithms, the different algorithmic classes of the underlying frameworks have to be considered and differentiated. For this paper, all computations were conducted utilizing one NVIDIA Tesla P100 GPU and an Intel Xeon 2.20 GHz CPU. As explained in [11], TraVaS operates as a selection-based technique devoid of explicit training requirements. Instead, it dynamically introduces specific noise during runtime into the variant distribution, a process that scales with the number of variants and can be parallelized. Consequently, the application runtime remains under 1 s, albeit necessitating the event log being loaded into memory. On the contrary, both TraVaG and the DDPM framework heavily rely on distinct ANN structures, with their training and application runtimes contingent upon various factors such as the model architecture, used library implementations, training schedules, and fine-tuned hyperparameters. Whereas for TraVaG, both the Autoencoder and the GAN need to be trained, the DDPM only comprises an untrained noise generator and a single denoising ANN. Due to batch processing, it is not imperative to load the entire event log into memory or GPU. On our hardware setup, once trained, TraVaG requires approximately 0.3 s to generate an artificial event log equivalent in size to BPIC-2012-App (comprising 13 087 cases), while the DDPM generator takes about 7 s. The notable disparity in runtime stems from the iterative sampling approach employed

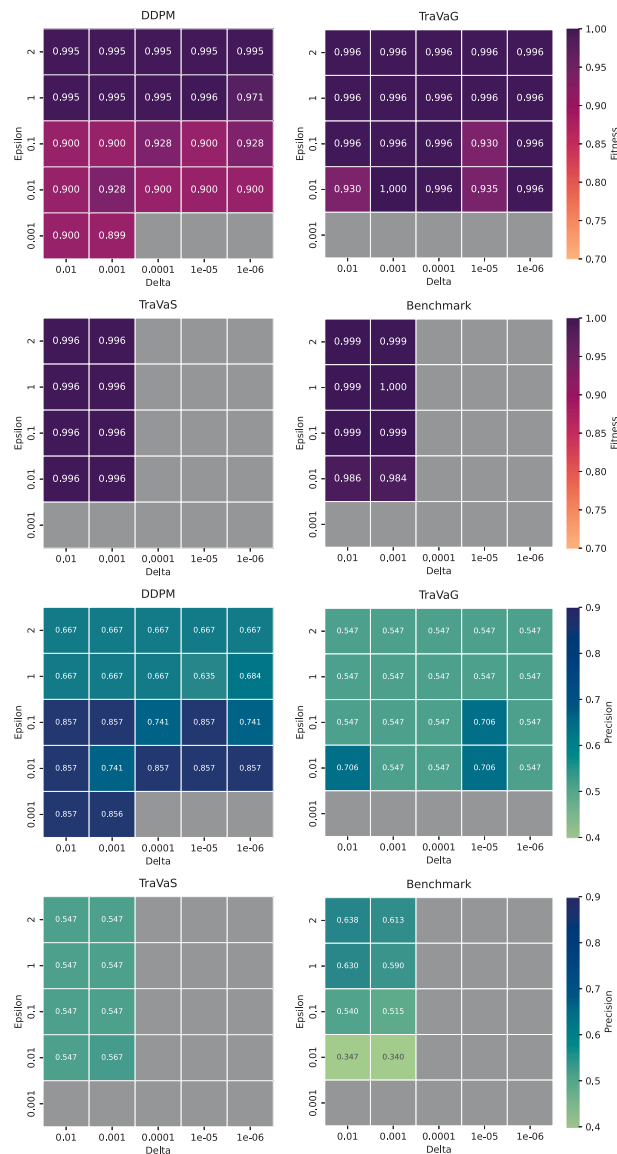


Fig. 8. The *fitness* and *precision* results of anonymized BPIC-2012-App event logs generated using DDPM, TraVaG, TraVaS, and the benchmark. Each value represents the mean of 100 generations for DDPM, TraVaG, and 10 algorithm runs for TraVaS and the benchmark.

by DDPMs, involving a sequence of denoising steps (300 in our implementation), wherein the trained network is applied at each step individually. A more detailed investigation of training and application runtime can be found on Github.⁷

As implied by the ANN-based generators of the TraVaG and DDPM frameworks described in Section 4, the internally learned variant distribution progressively enhances in accuracy with increasing size of the input event logs. Notably, performance improvements are observed with simpler distribution shapes and larger minimal case counts per variant. This naturally prompts the inquiry into determining the minimum training data size requisite for model applicability. Our experiments reveal that this determination is contingent upon several factors, including the magnitude of DP, i.e., the induced noise during training, the architectural configuration of the model, and the trace uniqueness serving as a proxy for the distribution shape. Given a specific event log, model complexity, and (ϵ, δ) parameters, the limit can be ascertained through an iterative process of scaling down variant frequencies until model convergence becomes unattainable. If no convergence appears even for the original event log, likely explanations include either an ill-suited, often excessively intricate model architecture or a training data size below the threshold implied by current configurations. In our experimentation with the Sepsis, BPIC-2013, and BPIC-2012-App datasets (see Section 5.1),

⁷ https://github.com/wangelik/TraVaGen/blob/main/supplementary/Complexity_Supplementary.pdf.

the data utility analysis revealed that the threshold was reached with ϵ ranging between 0.01 and 0.001 for the investigated δ regime and an event log representing 1050 cases across 846 variants (Sepsis).

7. Conclusion

With this work, we introduced two novel differentially private generative frameworks designed to facilitate the secure release of event data while ensuring quantified and guaranteed privacy. Our methodologies have successfully demonstrated that both training a differentially private combination of autoencoders and GANs as well as employing anonymized DDPMs to synthesize anonymized event data from an underlying original variant distribution outperform current state-of-the-art variant anonymization techniques for strong privacy levels in the low (ϵ, δ) range or complex event data structures. Our work on DP-based DDPM infrastructures is the first attempt to leverage privatized DDPMs on structured and high-dimensional tabular data. Furthermore, both generative models offer unique advantages, including an outstanding resource-efficient execution, the absence of distorting noise thresholds, a general acceptance of continuous data streams, and zero fake variant generation.

Overall, these characteristics enable the underlying generative algorithms to work efficiently with complex event data and lower ranges for δ , which is a unique feature to the best of our knowledge. However, it is important to acknowledge that our frameworks entail a more intricate training process and privacy budget management compared to conventional methods, such as TraVaS [11]. Because of the DP-SGD mechanisms that rely on RDP, it is not possible to directly extract or insert the conventional DP parameters (ϵ, δ) into the model training process, as explained in [31]. Instead, we must employ the one-way procedure detailed in Section 4.4. Accordingly, this involves first obtaining the RDP parameters (ϵ, α) based on the noise multiplier Φ , sampling rate q , and the number of iterations T , and subsequently converting these (ϵ, α) parameters into (ϵ, δ) . As a result, guaranteeing specific privacy levels necessitates an iterative analysis of various ANN settings until a suitable configuration is identified. In future research, this dependency on privacy-related hyperparameters could be investigated more comprehensively and potentially integrated into a fully automated tuning strategy. Depending on the available computational resources, such an approach could then transform TraVaG and the DDPM engine into streamlined, parameter-free methods akin to TraVaS.

CRedit authorship contribution statement

Frederik Wangelik: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Majid Rafiei:** Writing – review & editing, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Formal analysis. **Mahsa Pourbafrani:** Writing – review & editing, Validation, Supervision, Resources, Methodology, Investigation, Formal analysis, Conceptualization. **Wil M.P. van der Aalst:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

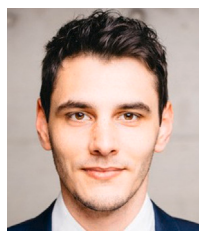
Data availability

All codes and data are published at GitHub and referenced in the document.

References

- [1] GDPR, 2023, URL <http://data.europa.eu/eli/reg/2016/679/oj>. (Accessed 01 October 2023).
- [2] C. Dwork, Differential privacy: A survey of results, in: *Theory and Applications of Models of Computation*, 5th International Conference, TAMC 2008, Proceedings, Vol. 4978, Springer, 2008, pp. 1–19.
- [3] A. Cohen, K. Nissim, Towards formalizing the GDPR's notion of singling out, *Proc. Natl. Acad. Sci.* 117 (15) (2020) 8344–8352.
- [4] W.M.P. van der Aalst, *Process Mining - Data Science in Action*, Second Edition, Springer, 2016.
- [5] F. Mannhardt, A. Koschmider, N. Baracaldo, M. Weidlich, J. Michael, Privacy-preserving process mining - differential privacy for event logs, *Bus. Inf. Syst. Eng.* 61 (5) (2019) 595–614.
- [6] S.A. Fahrenkrog-Petersen, H. van der Aa, M. Weidlich, PRIPEL: privacy-preserving event log publishing including contextual information, in: *Business Process Management - 18th International Conference, BPM 2020, Proceedings*, Vol. 12168, Springer, 2020, pp. 111–128.
- [7] M. Rafiei, F. Wangelik, M. Pourbafrani, 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 - 17th International Conference, RCIS 2023, Proceedings*, in: *Lecture Notes in Business Information Processing*, vol. 476, Springer, 2023, pp. 415–431.
- [8] S.A. Fahrenkrog-Petersen, M. Kabierski, F. Rösel, H. van der Aa, M. Weidlich, SaCoFa: Semantics-aware control-flow anonymization for process mining, in: *3rd International Conference on Process Mining, ICPM 2021, IEEE, 2021*, pp. 72–79.
- [9] G. Elkoumy, A. Pankova, M. Dumas, Mine me but don't single me out: Differentially private event logs for process mining, in: *3rd International Conference on Process Mining, ICPM 2021, IEEE, 2021*, pp. 80–87.
- [10] G. Elkoumy, M. Dumas, Libra: High-utility anonymization of event logs for process mining via subsampling, 2022, CoRR abs/2206.13050.
- [11] M. Rafiei, F. Wangelik, W.M.P. van der Aalst, TraVaS: differentially private trace variant selection for process mining, in: *Process Mining Workshops - ICPM 2022 International Workshops*, Springer, 2022.

- [12] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A.C. Courville, Y. Bengio, Generative adversarial networks, *Commun. ACM* 63 (11) (2020) 139–144.
- [13] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, in: F.R. Bach, D.M. Blei (Eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015*, in: *JMLR Workshop and Conference Proceedings*, vol. 37, JMLR.org, 2015, pp. 2256–2265.
- [14] F. Croitoru, V. Hondru, R.T. Ionescu, M. Shah, Diffusion models in vision: A survey, *IEEE Trans. Pattern Anal. Mach. Intell.* 45 (9) (2023) 10850–10869.
- [15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models, in: *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18–24, 2022*, IEEE, 2022, pp. 10674–10685, <http://dx.doi.org/10.1109/CVPR52688.2022.01042>.
- [16] H.B. McMahan, G. Andrew, A general approach to adding differential privacy to iterative training procedures, 2018, *CoRR* abs/1812.06210.
- [17] M. Abadi, A. Chu, I.J. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, ACM, 2016, pp. 308–318.
- [18] N.C. Abay, Y. Zhou, M. Kantarcioglu, B. Thuraisingham, L. Sweeney, Privacy preserving synthetic data release using deep learning, in: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Vol. 11051*, Springer, 2018.
- [19] Q. Chen, C. Xiang, M. Xue, B. Li, N. Borisov, D. Kaafar, H. Zhu, Differentially private data generative models, 2018, *CoRR* abs/1812.02274.
- [20] L. Frigerio, A.S. de Oliveira, L. Gomez, P. Duverger, Differentially private generative adversarial networks for time dummy-series, continuous, and discrete open data, in: *ICT Systems Security and Privacy Protection - 34th IFIP TC 11 International Conference, SEC 2019, Vol. 562*, Springer, 2019.
- [21] U.T. Tantipongpipat, C. Waites, D. Boob, A.A. Siva, R. Cummings, Differentially private synthetic mixed-type data generation for unsupervised learning, *Intell. Decis. Technol.* 15 (4) (2021) 779–807.
- [22] A. Kotelnikov, D. Baranchuk, I. Rubachev, A. Babenko, TabDDPM: Modelling tabular data with diffusion models, in: A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, J. Scarlett (Eds.), *International Conference on Machine Learning, ICML 2023, 23–29 July 2023*, in: *Proceedings of Machine Learning Research*, vol. 202, PMLR, 2023, pp. 17564–17579.
- [23] K. Li, S. Yang, T.M. Sullivan, R.S. Burd, I. Marsic, Generating privacy-preserving process data with deep generative models, 2022, *CoRR* abs/2203.07949.
- [24] F. Taymouri, M.L. Rosa, S.M. Erfani, Z.D. Bozorgi, I. Verenich, Predictive business process monitoring via generative adversarial nets: The case of next event prediction, in: *Business Process Management - 18th International Conference, BPM 2020, Vol. 12168*, Springer, 2020.
- [25] Y. Lu, Q. Chen, S.K. Poon, A deep learning approach for repairing missing activity labels in event logs for process mining, *Inf.* 13 (5) (2022) 234.
- [26] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in: *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, 2014.
- [27] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12, 2020*.
- [28] A.Q. Nichol, P. Dhariwal, Improved denoising diffusion probabilistic models, in: M. Meila, T. Zhang (Eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18–24 July 2021*, in: *Proceedings of Machine Learning Research*, vol. 139, PMLR, 2021, pp. 8162–8171.
- [29] G. Ács, L. Melis, C. Castelluccia, E.D. Cristofaro, Differentially private mixture of generative neural networks, *IEEE Trans. Knowl. Data Eng.* 31 (6) (2019) 1109–1121.
- [30] A.Q. Nichol, P. Dhariwal, Improved denoising diffusion probabilistic models, in: M. Meila, T. Zhang (Eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, in: *Proceedings of Machine Learning Research*, vol. 139, PMLR, 2021, pp. 8162–8171.
- [31] I. Mironov, Rényi differential privacy, in: *30th IEEE Computer Security Foundations Symposium, CSF 2017*, IEEE Computer Society, 2017, pp. 263–275.
- [32] J. Tang, A. Korolova, X. Bai, X. Wang, X. Wang, Privacy loss in apple's implementation of differential privacy on MacOS 10.12, 2017, *CoRR* abs/1709.02753.
- [33] P. Liashchynskiy, P. Liashchynskiy, Grid search, random search, genetic algorithm: A big comparison for NAS, 2019, *CoRR* abs/1912.06059.
- [34] M. Rafiei, W.M.P. van der Aalst, Group-based privacy preservation techniques for process mining, *Data Knowl. Eng.* 134 (2021) 101908.
- [35] F. Mannhardt, Sepsis cases, 2016, <http://dx.doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460>.
- [36] B.F. van Dongen, B. Weber, D.R. Ferreira, J.D. Weerd, BPI challenge 2013, in: *Proceedings of the 3rd Business Process Intelligence Challenge*, 2013, <http://dx.doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07>.
- [37] B.F. van Dongen, BPI challenge 2012, 2013, <http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f>.
- [38] M.M. de Leoni, F. Mannhardt, Road traffic fine management process, 2015, <http://dx.doi.org/10.4121/UUID:270FD440-1057-4FB9-89A9-B699B47990F5>, URL <https://data.4tu.nl/articles/12683249/1>.
- [39] M. Rafiei, W.M.P. van der Aalst, Towards quantifying privacy in process mining, in: *Process Mining Workshops - ICPM 2020 International Workshops*, vol. 406, Springer, 2020, pp. 385–397.
- [40] S.J.J. Leemans, D. Fahland, W.M.P. van der Aalst, Discovering block-structured process models from incomplete event logs, in: *Application and Theory of Petri Nets and Concurrency - 35th International Conference, PETRI NETS 2014, 2014. Proceedings*, vol. 8489, Springer, 2014, pp. 91–110.



Frederik Wangelik received a B.Sc. in Physics from the RWTH Aachen University with a thesis at CERN, Geneva. Furthermore, he holds three M.Sc. degrees in Physics, Business Administration and Data Science from the RWTH Aachen University with international research projects in the domain of radiology and nuclear physics at Surrey (GB), quantum computing and quantum information processing at quTech, Aachen, entrepreneurship and innovation research at Tsinghua, Beijing (CH), and probabilistic inference and machine learning at Bosch, Stuttgart. For his data science thesis, Frederik specialized in privacy-aware process mining with Majid Rafiei as supervisor at the PADS Chair of Prof. Dr. Wil van der Aalst, Aachen.



Dr. Majid Rafiei is Senior Process and Data Scientist and received his Ph.D. in Process and Data Science from the Chair of Process and Data Science (PADS) at RWTH Aachen University. He also holds as Master of Engineering in Electronic Commerce from Amirkabir University of Technology, Tehran. His research interest includes Process Mining, Machine Learning, Responsible Data Science, and Information Retrieval.



Dr. Mahsa Pourbafrani is Senior Data Scientist and received his Ph.D. in Process and Data Science from the Chair of Process and Data Science (PADS) at RWTH Aachen University. She received her B.Sc. and M.Sc. degrees in computer science from the Amirkabir University of Technology, Tehran, Iran. Her research interest includes machine learning and process mining, which employs data science methods to turn data into actionable insights. The actions are taken using simulation, what-if analysis, and predictions in process mining regarding the performance metrics of processes.



Prof. Dr. Wil van der Aalst is a full professor at RWTH Aachen University leading the Process and Data Science (PADS) group. He is also part-time affiliated with the Technische Universiteit Eindhoven (TU/e). Until December 2017, he was the scientific director of the Data Science Center Eindhoven (DSC/e) and led the Architecture of Information Systems group at TU/e. Since 2003, he holds a parttime position at Queensland University of Technology (QUT).