Kuhn, Martin; Grüger, Joscha; Bergmann, Ralph

PALSYN: a method for synthetic multi-perspective event log generation with differential private guarantees

Process Science

Provided in Cooperation with:
Springer Nature

Suggested Citation: Kuhn, Martin; Grüger, Joscha; Bergmann, Ralph (2025) : PALSYN: a method for synthetic multi-perspective event log generation with differential private guarantees, Process Science, ISSN 2948-2178, Springer International Publishing, Cham, Vol. 2, Iss. 1, https://doi.org/10.1007/s44311-025-00033-5

This Version is available at:
https://hdl.handle.net/10419/335074

Kuhn *et al. Process Science* (2025) 2:28
https://doi.org/10.1007/s44311-025-00033-5

Process Science

RESEARCH Open Access

PALSYN: a method for synthetic multi-perspective event log generation with differential private guarantees

Martin Kuhn[1,2*], Joscha Grüger[1,2] and Ralph Bergmann[1,2]

*Correspondence:
Martin Kuhn
martin.kuhn@dfki.de
[1]German Research Center for Artificial Intelligence (DFKI), Trier, Germany
[2]Business Information Systems II, Trier University, Trier, Germany

Abstract
The increasing reliance on data-driven technologies such as Artificial Intelligence and Process Mining has transformed various sectors. Yet, access to real-world data is often restricted by privacy concerns. Synthetic data offers a promising solution by enabling secure data sharing while preserving key characteristics for analysis. This paper introduces the Private Autoregressive Log Synthesizer (PALSYN), a novel approach for generating synthetic event logs with differential privacy guarantees. It employs advanced deep learning techniques to capture the complexity of event data while ensuring privacy. In contrast to existing methods, PALSYN can synthesize private multi-perspective event logs. The evaluation demonstrates the approach's ability to generate synthetic event logs that closely resemble the original data across key metrics. However, the results highlight the inherent privacy-utility trade-off, with stricter privacy settings introducing noise that strongly impacts utility. By enabling the generation of synthetic event logs with formal privacy guarantees, PALSYN demonstrates significant potential for securely sharing event data in privacy-sensitive domains.

Keywords Synthetic event logs, Process mining, Differential privacy, Deep learning, Generative models

Introduction

Data-driven technologies such as Artificial Intelligence (AI) and Process Mining have a profound impact on various sectors, driving the development of innovative approaches and solutions. These technologies are fundamentally dependent on data for progress. However, access to real-world data is not always guaranteed, particularly due to privacy concerns like General Data Protection Regulation (GDPR). A promising solution is synthetic data generation, which can ensure data utility while safeguarding privacy. When designed appropriately, synthetic datasets prevent direct links to real individuals' identities, enabling privacy-preserving data sharing while retaining useful analytical characteristics (Jordon et al. 2022).

Springer

Numerous methods for generating synthetic data have been developed across various fields and data formats. These include synthetic image generation using generative adversarial networks (GANs) (Karras et al. 2021) and diffusion models (Rombach et al. 2022), text generation with large language models (LLMs) (Brown et al. 2020), tabular data synthesis (Xu et al. 2019), and event log generation (Burattin et al. 2022). Notably, the generation of synthetic event logs represents an emerging and increasingly significant area of research. Event logs consist of traces, each representing a sequence of events that play a critical role in areas such as process mining (Aalst 2016). Each event typically contains information such as the activity performed, the case it belongs to, timestamps, resources involved, and other domain-specific attributes. Event logs that include rich contextual information beyond control-flow are called multi-perspective event logs (Mannhardt 2018). Furthermore, event logs present unique challenges, including ad hoc flexibility, numerous attributes, temporal dependencies, and highsequence uniqueness (Munoz-Gama et al. 2022). Existing methods for generating synthetic event logs often require extensive manual configuration. This includes defining process models, guards, distributions, and probabilities (Burattin et al. 2022; Grüger et al. 2023; Kuhn et al. 2024; Esgin and Karagoz 2019). Such manual approaches can produce synthetic data that lack realism and fails to serve its intended purposes (Grüger et al. 2023; Kuhn et al. 2024). On the other hand, there are multiple deep learning approaches for generating synthetic event logs, including methods based on GANs (Li et al. 2024; Taymouri et al. 2020), recurrent neural networks (RNNs) (Camargo et al. 2019), and variational autoencoders (Graziosi et al. 2024). Recent advancements enable the generation of privacy-preserving synthetic event logs by learning patterns directly from real data, reducing the need for manual configuration. However, current approaches for generating synthetic multi-perspective event logs do not offer formal privacy guarantees (Li et al. 2024; Graziosi et al. 2024). Alternative privacy-preserving techniques for event logs are based on differential privacy (DP). These often focus solely on the control-flow and time perspectives, thereby neglecting multi-perspective logs (Elkoumy et al. 2021; Rafiei et al. 2023; Wangelik et al. 2025).

In contrast, (Fahrenkrog-Petersen et al. 2020) proposed a privacy-aware framework called PRIPEL that explicitly supports the anonymization of multi-perspective event logs by utilizing DP. The event log is anonymized in a two-step approach. First, the control-flow is anonymized by adding noise to the counts of trace variants. Second, DP is applied to the timestamps and remaining attributes. To retain utility, anonymized traces are selected based on their similarity to the original traces, measured using the Levenshtein distance. However, matching anonymized traces back to the original log to assess similarity can introduce privacy risks, as it may enable re-identification of individuals (Schulze et al. 2024). Additionally, PRIPEL lacks precise control over the output size of the event log and is computationally expensive, which can hinder its applicability to large or complex datasets unless substantial preprocessing or integration with complementary techniques is employed (Fahrenkrog-Petersen et al. 2020, 2023).

To address these limitations, this work makes two primary contributions. First, the Private Autoregressive Log Synthesizer (PALSYN) is introduced as a novel approach for synthesizing multi-perspective event logs, while offering formal privacy guarantees. To achieve this, PALSYN combines two privacy-preserving techniques, the training of DP autoregressive models and the application of DP clustering algorithms. Thereby

ensuring robust trace-level privacy throughout the event log synthesis process. Second, PALSYN is benchmarked against the only state-of-the-art method for anonymizing multi-perspective event logs with DP (Fahrenkrog-Petersen et al. 2020). The evaluation is conducted on real-world event logs with varying complexity. Thus, providing a comprehensive analysis of the trade-offs between a classical anonymization method and a method generating synthetic data to preserve privacy.

In summary, the evaluation shows that PALSYN effectively generates differentially private multi-perspective event logs that retain statistical and behavioral characteristics of the original event log. Compared to PRIPEL, PALSYN offers greater scalability, improved preservation of statistical event log properties and event attributes, and enhanced model generalization, particularly for large datasets. However, results also confirm a privacy-utility trade-off, with smaller logs more affected by stricter privacy settings for PALSYN. These findings highlight practical considerations for selecting between synthetic generation and anonymization approaches in the process mining domain. The remainder of the paper is organized as follows: Section Background provides background information on event logs, DP, and deep learning. Section Related work reviews related work, while Section Private autoregressive log synthesizer introduces the PALSYN approach for synthesizing private event logs. Section Evaluation presents the evaluation, followed by a discussion of results and limitations in Section Discussion. The final section concludes the study and outlines further directions for future research.

Background

This section provides the necessary theoretical foundations for the methods presented in later chapters. The following subsections introduce the concepts of event logs, neural network architectures, and privacy-preserving techniques that form the basis for the proposed approach.

Event log

Event logs are multi-sets of cases, where each case consists of a sequence of events, known as a trace. Events are activity executions, where a single activity may be represented by multiple events. Beyond control-flow analysis, event logs can include attributes that represent different perspectives, such as the data perspective (Aalst 2016). These event logs are called multi-perspective event logs and extend traditional control-flow analysis by capturing diverse attributes such as organizational resources, timestamps, and event data, enabling analysis across multiple dimensions. Furthermore, a distinction must be made between *trace attributes*, which describe static attributes that are consistent throughout the trace and *event attributes*, which are event-specific and may vary for each event's execution. In the following, event logs, traces, and events are defined.

Definition 1 (Universes) We define the following universes used throughout this paper:

- $\mathcal{C}$: the universe of all possible case identifiers.
- $\mathcal{L}$: the universe of all possible event logs.
- $\mathcal{E}$: the universe of all possible event identifiers.
- $\mathcal{A}$: the universe of all possible activity identifiers.
- $\mathcal{AN}$: the universe of all possible attribute identifiers.

Definition 2 (Attributes and Classifier) Attributes are used to characterize events and cases. For instance, an event can be associated with a resource or have a timestamp. For any event $e \in \mathcal{E}$, case $c \in \mathcal{C}$, and attribute name $n \in \mathcal{AN}$, $\#_n(e)$ denotes the value of attribute n for event e, and $\#_n(c)$ denotes the value of attribute n for case c. If event e has no attribute n, then $\#_n(e) = \perp$. Similarly, if case c has no attribute n, then $\#_n(c) = \perp$. A classifier is a function that assigns a label to an event, typically used to identify the activity represented by the event. By default, the classifier is assumed to be $\underline{e} = \#_{activity}(e)$ (Aalst 2016).

Definition 3 (Trace and Case) Each case $c \in \mathcal{C}$ has a mandatory attribute called *trace*, denoted as $\hat{c} = \#_{trace}(c)$, where $\hat{c} \in \mathcal{E}^* \setminus \{\langle\rangle\}$. A trace is a finite sequence of events $\sigma \in \Sigma^*$, where Σ is the set of all possible events, and each event occurs only once within the trace. Formally:

$$\forall 1 \leq i < j \leq |\sigma| : \sigma(i) \neq \sigma(j).$$

The addition of an event e to a trace σ is denoted by $\sigma \oplus e = \sigma \cdot \langle e \rangle$. Additionally, for all events $e_i, e_j \in \sigma$, if $i \leq j$, then $\#_{time}(e_i) \leq \#_{time}(e_j)$ (Aalst 2016). This ensures that events are ordered chronologically.

Definition 4 (Event Log) An event log $L \in \mathcal{L}$ is defined as a set of cases $L \subseteq \mathcal{C}$, where each event appears only once in the log. If the event log includes timestamps, events within each trace must be ordered chronologically according to their timestamps (Aalst 2016). The set of all events appearing in the log L is denoted by $\hat{L}$ and is defined as:

$$\hat{L} = \{e \mid c \in L \wedge e \in \hat{c}\}.$$

Long short-term memory

Recurrent Neural Networks (RNNs) are a class of neural networks designed to process sequential data by maintaining a hidden state that captures information about previous elements in the sequence. Unlike feedforward neural networks, RNNs use their internal memory to process variable-length sequences, making them suitable for tasks such as time series analysis, natural language processing, and speech recognition (Schmidhuber 2015). However, traditional RNNs suffer from the vanishing gradient problem, which limits their ability to learn long-term dependencies in sequences. Long Short-Term Memory (LSTM) networks were introduced to address these limitations by incorporating a more sophisticated architecture with gating mechanisms (Hochreiter and Schmidhuber 1997). An LSTM unit replaces the simple hidden state of standard RNNs with a more complex structure that includes multiple gates to control the flow of information. These gates enable LSTMs to retain relevant information over longer sequences while discarding irrelevant details. The operation of an LSTM at each time step t in an input sequence $x = (x_1, x_2, \ldots, x_T)$ is defined by Eq. 1:

$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) && \text{(Input Gate)} \\
f_t &= \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) && \text{(Forget Gate)} \\
o_t &= \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) && \text{(Output Gate)} \\
g_t &= \tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g) && \text{(Cell State Update)} \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t && \text{(New Cell State)} \\
h_t &= o_t \odot \tanh(c_t) && \text{(Hidden State Output)}
\end{aligned} \tag{1}
$$

The input gate (i_t) determines how much of the new input x_t should influence the cell state. This decision is made using a sigmoid activation function along with the weights W_{ix}, W_{ih}, and a bias term b_i. The forget gate (f_t) decides how much of the previous cell state c_{t-1} should be retained. This operation is controlled by the weights W_{fx}, W_{fh}, and the bias b_f. The output gate (o_t) regulates how much of the current cell state contributes to the hidden state h_t. This is achieved using weights W_{ox}, W_{oh}, and the bias b_o. The cell state update (g_t) computes candidate updates to the cell state using a hyperbolic tangent ($\tanh$) activation function and the weights W_{gx}, W_{gh}, along with a bias term b_g. The new cell state (c_t) is then formed by combining the retained information from the forget gate with the new information from the input gate. Finally, the hidden state output (h_t) is produced by applying the output gate's result to the hyperbolic tangent of the updated cell state, which forms the final output for the current time step. The gating mechanisms allow LSTMs to selectively decide what information to store, update, or discard at each time step, making them well-suited for learning long-term dependencies in sequential data (Hochreiter and Schmidhuber 1997; Gers et al. 2000). A simplified alternative to LSTMs is the Gated Recurrent Unit (GRU) (Cho et al. 2014), which merges the input and forget gates into a single update gate and combines the cell and hidden states, resulting in a more compact architecture with fewer parameters while often achieving similar performance on sequential learning tasks.

Differential privacy

Differential privacy (DP) provides a mathematical framework designed to define and quantify the privacy of individual data points within a dataset while still enabling the analysis of the overall dataset. It requires two privacy parameters, ϵ and δ. The privacy budget, $\epsilon \geq 0$, limits how much the inclusion or exclusion of a single data point can affect an algorithm's output. A small ϵ provides a strong level of privacy, indicating a lower likelihood that any single data point will significantly influence the output. Conversely, δ quantifies the probability of a privacy failure, where the guarantee of ϵ might not hold. Typically, δ is constrained such that $0 < \delta < 1$ (Dwork et al. 2006).

The DP mechanism is defined as a randomized algorithm $\mathcal{A}$, that satisfies (ϵ, δ) -DP. This property holds if, for any two neighboring datasets D and D', differing by at most one element, and for all subsets S of possible outputs, the following inequality is satisfied:

$$
\Pr[\mathcal{A}(D) \in S] \leq e^{\epsilon} \cdot \Pr[\mathcal{A}(D') \in S] + \delta \tag{2}
$$

This inequality helps to quantify that the presence or absence of any single individual in the dataset does not significantly increase the likelihood of any particular output, thus providing robust privacy protections. DP provides several key theorems that establish how privacy guarantees are maintained or degrade under various operations. These

include the *Sequential Composition Theorem*, the *Parallel Composition Theorem*, and the *Post-Processing Theorem* (Dwork et al. 2006).

Theorem 1 (Sequential Composition) When multiple DP mechanisms are applied to the same dataset, the overall process remains differentially private, but the privacy parameters degrade (Dwork and Roth 2014). A basic composition bound states that the resulting privacy parameters (ϵ', δ') are the sums of the individual privacy parameters (ϵ_i, δ_i). Formally, if $A_1, \ldots, A_t$ are t mechanisms where the i-th mechanism satisfies (ϵ_i, δ_i)-DP, then the combined mechanism, i.e., $(A_1, \ldots, A_t)$, satisfies (ϵ', δ')-DP with:

$$\epsilon' = \sum_{i=1}^{t} \epsilon_i, \quad \delta' = \sum_{i=1}^{t} \delta_i.$$

Theorem 2 (Parallel Composition) In parallel composition, unlike sequential composition where all mechanisms are applied to the same dataset, the dataset is divided into disjoint subsets, with each mechanism operating on a unique subset. Let $A_1, \ldots, A_t$ represent the set of mechanisms, where the i-th mechanism satisfies (ϵ_i, δ_i)-DP. Parallel composition ensures that the combined mechanism, $(A_1, \ldots, A_t)$, satisfies $(\max_i \epsilon_i, \max_i \delta_i)$-DP. This guarantee is stronger than that of sequential composition because, in parallel composition, each record in the dataset is used only once, whereas in sequential composition, the same records can be used multiple times (Dwork et al. 2006).

Theorem 3 (Post-Processing Invariance) Any data-independent transformation applied to the output of a DP mechanism preserves its privacy guarantees, under the same privacy parameters. This property has two key implications: (1) An attacker cannot weaken or compromise the DP guarantees by manipulating the mechanism's output and (2) it allows for simplification in the design and analysis of complex DP systems, as post-processing does not affect the privacy guarantees (Dwork and Roth 2014).

Differential privacy in deep learning

DP plays a crucial role in deep learning for protecting individual data privacy, particularly when sensitive data is used to train models. In this context, DP defines privacy by introducing randomness during the training process. This is achieved through Differentially Private Stochastic Gradient Descent (DP-SGD), which extends the traditional Stochastic Gradient Descent (SGD) algorithm with DP. SGD is a fundamental optimization technique used in machine learning to minimize a loss function, which measures the difference between the predicted outputs of the model and the actual outcomes of the data. At each iteration, DP-SGD performs the following steps: (1) Sample a mini-batch of data, (2) compute individual gradients for each example, (3) clip each gradient according to the clipping threshold $C \in (0, \infty)$, (4) aggregate the clipped gradients, and (5) add calibrated Gaussian noise before applying the update to model parameters. The integration of DP into SGD involves modifying the algorithm to enforce that each update to the model's parameters minimizes the risk of exposing any individual data point in the training set (Abadi and Al. 2016). This is achieved through two main modifications: *Gradient Clipping* and *Noise Addition*. Gradient clipping ensures that no single data point has a disproportionately large effect on the computed gradient, effectively bounding the

influence of each example and thus protecting privacy. The clipping operation is defined as follows:

$$g' = \frac{g}{\max\left(1, \frac{\|g\|_2}{C}\right)} \tag{3}$$

$$\overline{g} = g' + \mathcal{N}(0, \sigma^2) \tag{4}$$

Here, g is the gradient computed during training, $\|g\|_2$ is the L2 norm of the gradient, and C is a predefined clipping threshold (Abadi and Al. 2016). After clipping, Gaussian noise is added to the gradients in a process called *Noise Addition*. The amount of noise added is calibrated based on the privacy budget ϵ and the sensitivity of the gradients, which is controlled by the clipping threshold C. In Eq. 4, $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise with mean zero and variance σ^2 (Abadi and Al. 2016). The privacy budget (ϵ, δ) quantifies the allowed leakage of information about individual data points. Lower values of ϵ imply stronger privacy, but typically require more noise or fewer training steps (see Sect. Privacy accounting). By combining these two mechanisms, DP-SGD enables deep learning models to train effectively while providing strong privacy guarantees for individual data points.

Privacy accounting

Rényi Differential Privacy (RDP) offers an alternative mathematical framework to the traditional (ϵ, δ)-DP formulation presented in Section Differential privacy. RDP uses Rényi divergence to quantify the privacy loss between probability distributions resulting from the inclusion or exclusion of an individual element. This approach enables tighter and more accurate privacy accounting, particularly suited for iterative algorithms such as DP-SGD, which accumulate privacy loss over multiple steps (Mironov 2017).

Definition 5 (Rényi Divergence) Let P and Q be two probability distributions over the same domain $\mathcal{X}$. The Rényi divergence of order $\alpha > 1$ is defined as:

$$D_\alpha(P\|Q) = \frac{1}{\alpha - 1} \log \mathbb{E}_{x\sim Q}\left[\left(\frac{P(x)}{Q(x)}\right)^\alpha\right]$$

Definition 6 (Rényi Differential Privacy) A randomized mechanism $\mathcal{M}$ is said to satisfy (α, ϵ)-RDP if for any two neighboring datasets D and D', the Rényi divergence of order α between the output distributions $\mathcal{M}(D)$ and $\mathcal{M}(D')$ is bounded by ϵ:

$$D_\alpha(\mathcal{M}(D)\|\mathcal{M}(D')) \leq \epsilon$$

The parameter α controls the trade-off between privacy and utility, with larger values of α corresponding to stronger privacy guarantees but potentially reduced utility. In contrast to standard DP, which expresses guarantees using ϵ and δ as privacy parameters, RDP characterizes privacy loss as a function $\epsilon(\alpha)$ that defines privacy loss for each order α. This functional relationship provides a more detailed privacy loss across different values of α.

RDP guarantees can be converted to standard (ϵ, δ)-DP guarantees using Theorem 4. This conversion allows practitioners to use RDP for analysis and accounting while expressing final privacy guarantees in the more widely-used (ϵ, δ)-DP framework. Similar to standard DP, RDP possesses important composition properties that make it particularly suitable for iterative algorithms. The sequential composition property (Theorem 5) is particularly useful in the context of DP-SGD, where noise is repeatedly added over multiple training steps, requiring cumulative accounting of privacy loss. Building upon the DP-SGD framework introduced in Section Differential privacy in deep learning, RDP provides a more precise way to track privacy loss during training. The Gaussian noise mechanism used in DP-SGD (as shown in Eq. 4) has well-characterized RDP guarantees (Mironov 2017).

Theorem 4 (RDP to DP Conversion) If a mechanism $\mathcal{M}$ satisfies (α, ϵ)-RDP, then for any $\delta > 0$, it also satisfies $(\epsilon + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$-DP.

Theorem 5 (Sequential Composition for RDP) If mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfy (α, ϵ_1) -RDP and (α, ϵ_2)-RDP respectively, then their composition $\mathcal{M}(D) = (\mathcal{M}_1(D), \mathcal{M}_2(D))$ satisfies $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.

Theorem 6 (RDP for Gaussian Mechanism) Let f be a function with sensitivity $\Delta_2 f$. The Gaussian mechanism $\mathcal{M}(x) = f(x) + \mathcal{N}(0, \sigma^2 \cdot \mathbb{I})$ satisfies $(\alpha, \alpha \Delta_2 f^2/(2\sigma^2))$-RDP.

In the context of DP-SGD, gradient sensitivity is bounded by the clipping threshold C (as defined in Eq. 3, resulting in a sensitivity of $\Delta_2 = C$. Therefore, each iteration of DP-SGD with noise scale σ satisfies $(\alpha, \alpha C^2/(2\sigma^2))$-RDP according to Theorem 6. When applying DP-SGD over multiple training iterations, the total privacy loss can be calculated using the sequential composition property of RDP (Theorem 5). For T iterations with the same noise scale σ and clipping threshold C, the cumulative RDP guarantee is:

$$\epsilon_{RDP}(\alpha) = \frac{\alpha C^2 T}{2\sigma^2} \tag{5}$$

$$\epsilon_{DP} = \frac{\alpha C^2 T}{2\sigma^2} + \frac{\log(1/\delta)}{\alpha - 1} \tag{6}$$

This RDP bound can then be converted to an (ϵ, δ)-DP guarantee using Theorem 4. By optimizing over different values of α, one can find the tightest ϵ_{DP} guarantee for a given δ (Mironov 2017). This refined accounting framework enables practitioners to achieve improved privacy-utility trade-offs in differentially private deep learning, by reducing overestimation of privacy loss while maintaining formal guarantees (Ponomareva and Al. 2023).

DP-k-means (EUGkM)

EUGkM (Extended Uniform Grid k-Means) is a non-interactive approach for differentially private k-means cluster analysis that extends the Uniform Grid (UG) method from

two-dimensional to higher-dimensional data spaces (Su et al. 2016). The method partitions the d-dimensional data space into M equal-sized grid cells and publishes a noisy count of data points contained in each cell, where Laplace noise with a scaling parameter is added to ensure ϵ-DP. The critical parameter M is determined through theoretical Mean Squared Error (MSE) analysis, which balances the trade-off between approximation error from spatial discretization and noise variance from DP, ensuring optimal accuracy for subsequent clustering analyses. This synopsis is created once and can subsequently be used for various k-means analyses with different parameters without further consuming the privacy budget. Experimental evaluations demonstrate that EUGkM exhibits superior performance compared to interactive methods such as DPLloyd, particularly under low privacy budgets (small ϵ values) (Su et al. 2016).

Related work

The growing demand for event logs has driven the development of synthetic data generation and privacy-preserving techniques in process mining. Thus, this section reviews key approaches to synthetic event log generation, encompassing both knowledge-driven and data-driven methods. Additionally, it examines privacy-preserving techniques, with a focus on DP.

In process mining, knowledge-driven generators produce event logs based on predefined process models. These approaches are well-suited for generating synthetic logs from models created with domain knowledge, but developing such models requires significant effort (Grüger et al. 2023). Most existing knowledge-driven event log generators focus primarily on the control flow, like (Burattin et al. 2022; Esgin and Karagoz 2019; Burattin and Sperduti 2010; Ciccio et al. 2015). However, there are examples of multi-perspective generators, including approaches based on multi-perspective Declare (Bergami 2023), probabilistic programming (Kuhn et al. 2024) and data-aware Petri nets (Grüger et al. 2023). Since these approaches rely on manual configuration, they can produce synthetic data that lacks realism and fails to serve its intended purposes (Grüger et al. 2023; Kuhn et al. 2024). Additionally, these methods are incompatible with use cases involving real event logs that demand strict privacy guarantees.

Data-driven generators take a fundamentally different approach by leveraging real event logs to replicate patterns found in the event log. One prominent example is proposed by Lin et al., which utilizes a LSTM to predict the next event and its categorical attributes (Lin et al. 2019). By assigning weights based on attribute importance, this method can improve its predictive power. However the method lacks an implementation to generate synthetic traces probabilistically and only supports categorical attributes. Also, building on the use of LSTMs, Camargo et al. introduced a method that incorporates timestamps and roles (Camargo et al. 2019). This approach ensures variability by sampling the next event based on the predicted probability distribution, though it occasionally produces traces inconsistent with the overall process distribution. Similarly, Evermann et al. proposed a LSTM with activity embeddings to reduce dimensionality issues while generating traces (Evermann et al. 2017). However, this approach does not support numerical attributes or timestamps. Beyond autoregressive approaches, GANs have been explored for synthetic data generation. For instance, the approach proposed by Van Dun et al., is a novel GAN-based method designed to generate synthetic traces for generating ideas to restructure existing processes (van Dun et al. 2023). This

approach supports the generation of traces with control-flow, timestamps, and activity durations. Another GAN-based approach is proposed by Taymouri et al., which combines GANs and LSTMs to predict the next event and its timestamp (Taymouri et al. 2020). This method combines recurrent layers with generative adversarial techniques to enhance trace generation capabilities. The Conditional Variational Autoencoder (CVAE) proposed by Graziosi et al. extends synthetic log generation by incorporating trace information (Graziosi et al. 2024). Moreover, CVAE lacks formal privacy guarantees, though it should be noted that the study's focus was not on data sharing but rather on what-if analysis and conditional log generation. Another approach is ProcessGAN, proposed by Li et al., which focuses on synthesizing event logs with support for control-flow, timestamps as well as event and trace attributes (Li et al. 2024). It incorporates contextual information, through a transformer-based generator. This component learns associations between traces and their attributes during training and produces synthetic contexts alongside synthetic traces, ensuring that generated data mirrors real-world complexity. While this approach generates event logs from noise to preserve privacy, it does not offer formal privacy guarantees. This limits its use in scenarios requiring stringent privacy protections.

There are several methods in the process mining domain that apply methods for privacy protection to event logs with the aim of releasing or sharing them. The method proposed by Rafiei et al. combines GANs with DP-SDG to synthesize event logs (Rafiei et al. 2023). Another method for generating synthetic DP event logs was proposed by Wangelik et al. and is based on Denoising Diffusion Probabilistic Models (DDPMs) that synthesize trace variants from noise using Markov chains (Wangelik et al. 2025). Both of the approaches can only synthesize traces that are part of the training data, thus avoiding the generation of fake trace variants. However this imposes that extremely rare or unique variants tied to individuals might be regenerated and are potentially re-identifiable (Wangelik et al. 2025). By avoiding the generation of fake trace variants the utility for process mining tasks is well preserved even if strong privacy guarantees are utilized (Rafiei et al. 2023; Wangelik et al. 2025). These two methods do not allow for the synthetization of multi-perspective event logs, thus focusing on the control-flow only.

Elkoumy et al. (2021) propose an approach for generating a directly-follows graph (DFG) by adding calibrated noise to the DFG. While they emphasize that their method does not involve the addition or removal of arcs, this assumption can be problematic in scenarios where individual trace variants influence the presence of specific activity pairs. A subsequent work from Elkoumy et al. (2022), present a method that tries to increase the utility by using subsampling techniques to reduce the required privacy budget. Furthermore, the event log is anonymized by selectively removing traces based on assumed attacker knowledge. Both of the approaches target the control-flow and timestamp for anonymization but neglect other perspectives thus are not capable for synthesizing multi-perspective event logs. (Fahrenkrog-Petersen et al. 2023) developed a method to release an anonymized distribution of trace variants utilizing DP. From this distribution, a prefix-tree is derived. However, the activity pairs included in the prefix-tree are preselected based on a scoring function. This function evaluates the inclusion of each activity in a branch based on predefined rules derived from domain knowledge or observed process behavior. While this approach aims to ensure DP, its reliance on information extracted from the original event log to design the scoring function can compromise

privacy guarantees. (Mannhardt et al. 2019) applied DP to event logs by introducing it for directly-follows relations (DFR) and trace variant frequencies. The Laplace mechanism is used to add noise, directly to DFR frequencies. However, their approach is computationally limited to short trace lengths and also is computationally expensive when used on larger event logs (Mannhardt et al. 2019).

Furthermore, Fahrenkrog-Petersen et al. proposed a framework for privacy-aware event log publishing called PRIPEL that explicitly considers the inclusion of additional attributes, thus making it possible to anonymize multi-perspective event logs (Fahrenkrog-Petersen et al. 2020). The event log is anonymized in a two step approach. First, the control-flow is modeled using a prefix-tree, where noise is added to the counts of trace variants. The noise is added utilizing DP. Afterwards, these anonymized trace variants are matched to the original event log, based on the trace similarity calculated by the Levenshtein distance. In a second step, DP is applied to the timestamp by introducing random shifts. Also, DP is applied to each other attribute separately. This assumes independence between attributes, which can limit their applicability to real-world scenarios. PRIPEL is computationally expensive and can result in long run times similarly to (Fahrenkrog-Petersen et al. 2020; Mannhardt et al. 2019), if strong privacy protection is needed or large event logs are anonymized. Furthermore, matching anonymized traces back to original traces can introduce vulnerabilities, as attackers may infer the presence of specific individuals according to the original traces, thus introducing re-identification risks. Another limitation of PRIPEL is that through the anonymization of the trace variant counts the number of traces in the final event log can drop or increase significantly compared to the original log (Fahrenkrog-Petersen et al. 2020).

Table 1 reveals that while numerous approaches exist for synthetic event log generation, only PRIPEL currently provides formal privacy guarantees while supporting multi-perspective event logs. However, PRIPEL also suffers from the shortcomings discussed earlier, which limit its applicability in certain scenarios. Currently, no approach exists, which uses an autoregressive model for multi-perspective synthesis with formal privacy guarantees. The PALSYN approach proposed in this publication, addresses this gap by being the first autoregressive model to achieve comprehensive multi-perspective

Table 1 Comparative analysis of state-of-the-art synthetic event log generation methods across control-flow (CF), timestamps (TS), event attributes (EA), and privacy mechanisms (PM)

Publication	Method	CF	TS	EA	PM
Lin et al. (2019)	LSTM	✓	×	✓[2]	×
Camargo et al. (2019)	LSTM	✓	✓	×	×
Evermann et al. (2017)	LSTM	✓	×	✓[2]	×
Van Dun et al. (2023)	GAN	✓	✓	×	×
Li et al. (2024)	GAN	✓	✓	✓	×
Graziosi et al. (2024)	CVAE	✓	✓	✓	×
Rafiei et al. (2023)	GAN	✓	×	×	✓
Wangelik et al. (2025)	DDPMs	✓	×	×	✓
(Mannhardt et al. 2019)	DFR	✓	×	×	✓
Elkoumy et al. (2021)	DFG	✓	✓	×	✓
Elkoumy et al. (2022)	DFG	✓	✓	×	✓
Fahrenkrog-Petersen et al. (2023)	Prefix-tree	✓	×	×	✓
Fahrenkrog-Petersen et al. (2020)	Prefix-tree	✓	✓	✓	✓
Our Approach	LSTM	✓	✓	✓	✓

[2] Limited to categorical attributes only

✓ = supported, × = not supported

synthesis with formal privacy guarantees. Thus, making it a contribution to the current state-of-the-art for privacy-preserving synthetization of event logs in process mining.

Private autoregressive log synthesizer

This section outlines the PALSYN approach, which effectively generates multi-perspective event logs while ensuring formal privacy guarantees. One of the main assumptions of PALSYN is that through the use of autoregressive models, the temporal dependencies hidden in the event log between events and attributes can be learned to a certain degree during training. Furthermore, PALSYN aims to retain statistical and behavioral properties across the entire event log, thereby supporting meaningful statistical analyses and process mining tasks.

Methodological design

The methodological design of PALSYN is motivated by seven findings from Section Related work:

(1) Knowledge-driven generators can produce multi-perspective event logs but require handcrafted models that are time-consuming to configure.
(2) Even when configured, these logs generated by knowledge-driven generators often lack realism.
(3) Data-driven generators based on GANs, although powerful, are prone to training instability and mode collapse.
(4) Many data-driven generators either do not support multi-perspective logs or do not implement formal privacy guarantees.
(5) Existing DP-based generators typically regenerate only known trace variants or rely on access to the original, unprotected event log for similarity calculations, introducing re-identification risks.
(6) DP-based anonymization techniques for multi-perspective logs are often computationally expensive and can result in long run times.
(7) DP-based anonymization techniques for multi-perspective logs do not guarantee the desired number of traces needed for process mining tasks.

PALSYN addresses these findings with following design choices. First, PALSYN utilizes autoregressive models, addressing finding (1) and finding (2) by enabling the automatic learning of both short- and long-term temporal relationships between events and their attributes. This avoids the need for manual modeling and improves realism, as demonstrated in prior work on process mining tasks (Camargo et al. 2019; Lin et al. 2019). Also by choosing autoregressive models instead of GANs, (3) is addressed as the problems inherent to GANs like instability of training and mode collapse are avoided. (4) is addressed by training the model with DP-SGD, ensuring that (ϵ, δ)-DP is formally achieved, when multi-perspective event logs are used for training. (5) Third, to eliminate the re-identification risks introduced by variant-based or similarity-based methods, PALSYN combines multiple DP mechanisms under the Sequential and Parallel Composition Theorems. Concretely, three complementary components are integrated: (a) DP-SGD for learning the model, (b) DP-K-Means for clustering numerical and time-related attributes during preprocessing, and (c) probabilistic sampling from a DP-compliant Gaussian distribution for generating start timestamps during post-processing.

Components (a) and (b) are applied sequentially, while component (c) is applied independently in parallel. Component (a) adopts the relaxed notion of (δ, ϵ)-DP, while components (b) and (c) adhere to pure ϵ-DP; consequently, the overall privacy guarantee of PALSYN must be expressed in relaxed form, because the presence of any (δ, ϵ)-DP component dictates the final bound. (6) by utilizing autoregressive models the complexity is set to the complexity of the chosen deep-learning architecture and can also be accelerated by using GPU computing. Lastly, (7) by relying on synthetic data generators the desired amount of traces can easily be generated.

Furthermore, PALSYN is designed to preserve specific statistical properties. (i) control flow is characterized by a realistic number of traces in the log, by statistics of activity occurrences, by the distribution of trace variants, by the distribution of trace lengths, and by similar fitness, precision, generalization, and simplicity of the process models discovered from the synthetic logs compared to models discovered from the real logs. (ii) timestamps are characterized by a similar distribution of case throughput times, defined as the time from the first to the last event of a case. (iii) event and trace attributes are characterized by similar per attribute distributions for categorical, boolean, and numerical attributes. (iv) privacy is measured by the achieved differential privacy guarantees based on (δ, ϵ)-DP as explained earlier.

These properties are preserved as follows. First, control-flow is preserved by training an autoregressive model on tokenized traces to predict the next event together with its event and trace attributes. Sampling from the learned next step probabilities aims to reproduce realistic control-flow structures and induces activity, variant, and length distributions consistent with those observed in the real log while preserving the process mining dimensions. Second, timestamps are preserved by representing absolute times as time between events during preprocessing and by reconstructing absolute times in postprocessing through cumulative summation, which should preserve the implied distribution of throughput times. Third, event and trace attributes are preserved by generating attribute values jointly with each next event using the autoregressive model. Numerical attributes are discretized during preprocessing and reconstructed during postprocessing, which should provide better statistical preservation of numerical values (Xu et al. 2019). Fourth, privacy is preserved via the composition theorems, parallel composition and sequential composition, thus the desired (δ, ϵ)-DP can be achieved. Thus, for (δ, ϵ)-DP, a formal guarantee is given. PALSYN does not explicitly enforce constraints to guarantee preservation of particular statistics. Instead, it learns these properties from data and uses targeted pre- and postprocessing to facilitate learning realistic behavior, which can result in preservation of statistical properties. Thus, it must be empirically validated if the statistical properties are preserved by the PALSYN approach.

The next sections of this chapter will define the architecture of the PALSYN approach and provide technical details on how it operates. This includes a detailed explanation of the preprocessing, training, sampling, and post-processing steps involved in the approach.

Numerical attribute encoding

Numerical attributes can present challenges for probabilistic generative models. Unlike categorical data, numerical values form continuous distributions that are challenging for discrete probabilistic models. This issue can be addressed through discretization, which

transforms numeric attributes into categorical ones, making probabilistic generation straightforward. PALSYN utilizes DP-K-Means, as proposed by (Su et al. 2016), to discretize numerical and time-related attributes into categorical attributes. DP-K-Means is specifically chosen to ensure compliance with DP guarantees, preserving the privacy of sensitive data. Additionally, the granularity of discretization can be adjusted by controlling the number of clusters, making this method adaptable to various datasets and requirements. By clustering the values of each numerical attribute into a predefined number of groups, each cluster is assigned a categorical label. In contrast to traditional methods that require imputation for missing numerical values, this approach incorporates missing values directly into the synthetization process by designating a specific cluster for them. An overview of this process is illustrated in Fig. 1. First, the numerical attributes are discretized using clustering. During this step, the statistical properties of each cluster such as minimum, maximum, mean, and standard deviation are calculated by using DP-K-Means. These properties are crucial for accurately modeling the data but remain protected under DP. During training, the model uses the categorical cluster labels as inputs, effectively working with a transformed representation of the numerical data. In the sampling phase, the generated clusters are transformed back into numerical values based on the statistical properties of the corresponding cluster. Since each

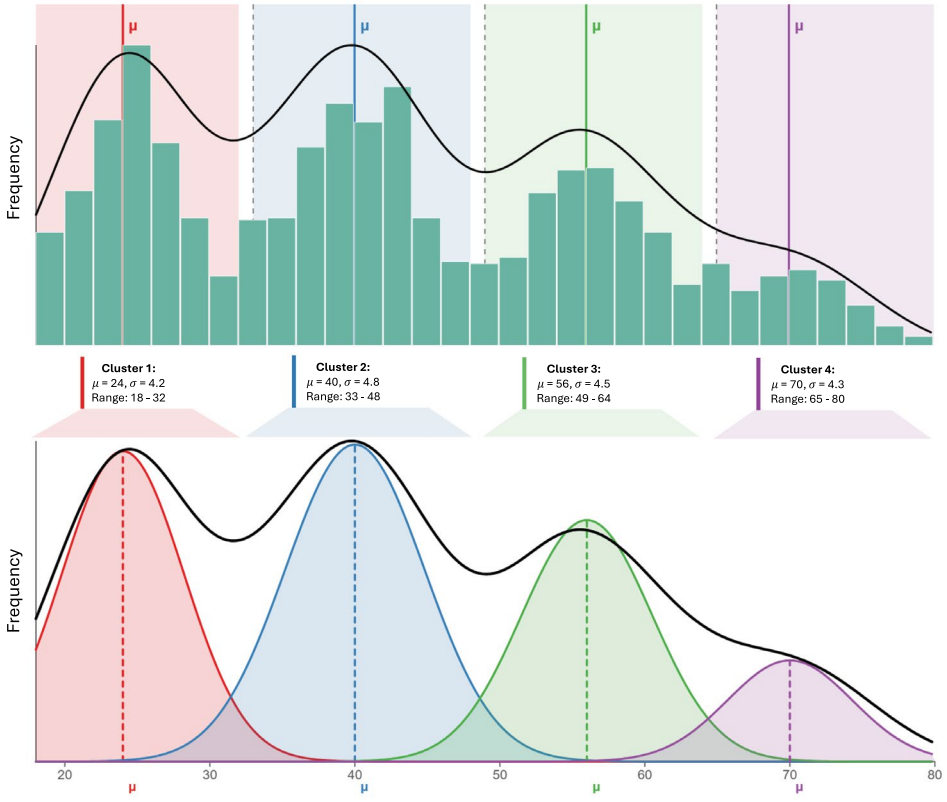

Fig. 1 This figure depicts the discretization of numerical and time-related attributes into categorical labels. This transformation makes it possible to learn probabilistic characteristics of the numerical attributes. DP-K-Means is applied to group attribute values into a predefined number of clusters (n), ensuring DP. Each colored distribution represents a distinct cluster. For each cluster, its statistical properties, specifically mean (μ) and standard deviation (σ), are calculated and stored. This effectively models the attribute's distribution as a mixture of Gaussian distributions. The resulting categorical cluster labels are used as tokens during training. In the sampling phase, these generated cluster labels are then transformed back into realistic numerical values by sampling from their corresponding Gaussian distributions, utilizing μ and σ for each cluster

cluster represents one component of a mixture model, the generated numerical values follow the same overall statistical behavior as the original attribute. The combination of clusters allows the synthetic data to approximate the distribution of the original numerical attribute while maintaining privacy guarantees. This concept shares similarities with techniques used in CTGAN, which leverage Gaussian Mixture Models instead of DP-K-Means (Xu et al. 2019).

Architecture

Figure 2 illustrates the PALSYN approach and its four phases. First, the preprocessing phase (see Sect. Preprocessing) converts each trace in the log into a sequence of tokens, ensuring they are a valid input for the RNN. This involves the discretization of numerical attributes to enable their representation as tokens (see Sect. Numerical attribute encoding). Second, the training phase processes the tokenized sequences and trains the RNN (see Sect. Training). Third, the sampling phase generates a synthetic sequence of tokens by drawing from the probability distributions learned by the RNN, as explained in Section Sampling. In the final phase, post-processing converts the generated token sequences back into a structured event log, as described in Section Post-processing.

The deep learning architecture of the PALSYN approach can be seen in the middle of Fig. 2. The first layer of the model is an embedding layer, designed to transform the input data into dense vectors that encapsulate the semantic characteristics of the input. The dimensionality of these vectors can be adjusted according to the complexity of the input data. Next, a masking layer follows, which addresses the presence of variable-length sequences by excluding padded positions from subsequent computations, thereby ensuring that only valid data contribute to the model's learning process.

The next part is composed of differentially private recurrent layers (RNN, GRU, LSTM). These layers are adapted to ensure DP in gradient updates using the DP-SGD implementation from (Abadi and Al. 2016) (see Sect. Differential privacy in deep learning). To improve training stability and model performance, a batch normalization layer follows, normalizing activations from the recurrent layers. The final component consists of multiple dense layers with softmax outputs, where each event attribute in the event

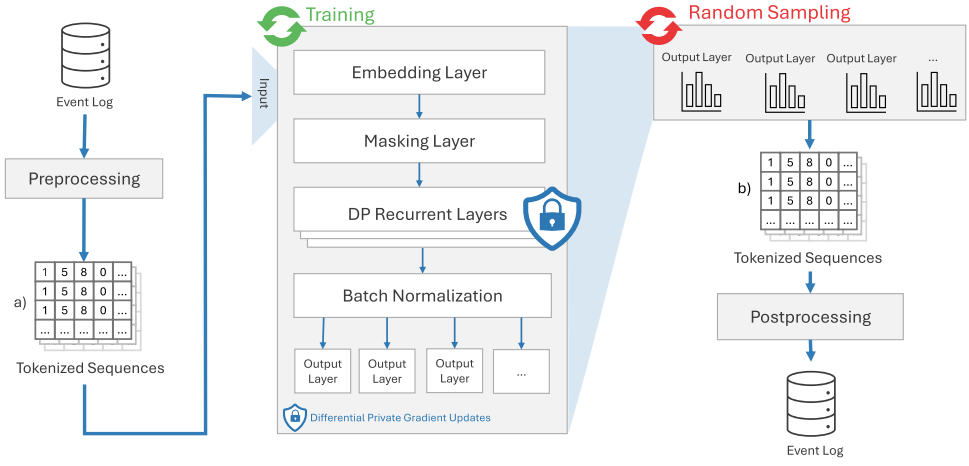

Fig. 2 Architectural overview of the PALSYN approach, illustrating the four-phase process: preprocessing, training, sampling, and post-processing. The figure depicts the transformation pipeline from input event logs through the neural network architecture to the generation of synthetic event logs

log is assigned its own output layer. Each layer computes a probability distribution over the target classes for its respective attribute, ensuring that the approach generates an entire event along with all its attributes.

Preprocessing

The preprocessing phase in PALSYN is crucial for converting raw event logs into a structured, numerical format suitable for RNN input. RNNs require uniform input sequences. However, standard event logs do not meet this requirement, as they contain diverse data types such as categorical and numerical attributes, and traces of varying lengths. Preprocessing is applied to handle this heterogeneity. In particular, numerical and time-related attributes are not directly compatible with probabilistic sequence modeling using autoregressive RNNs. This phase also integrates privacy-preserving mechanisms into the data transformation steps. For example, it ensures that discretization follows DP guarantees and that variable trace lengths are managed to create consistent RNN input. The method used to achieve this is DP-K-Means for discretization. PALSYNs preprocessing consists of three key steps: calculating time durations, clustering numerical attributes in a privacy-preserving way, and tokenizing traces. The preprocessing function is defined as follows.

Definition 7 (Preprocessing Function) *Let* $\mathrm{preprocess} : \mathcal{L} \times \mathbb{N} \to \mathcal{T}$ be defined as a function that executes a sequence of transformations on an event log $L \in \mathcal{L}$ by using predefined functions:

$$\mathrm{preprocess}(L, n) = token(discretization(tbe(L), n))$$

where $\mathcal{T}$ represents the set of all possible token sets, $T \in \mathcal{T}$ is the resulting set of tokens that capture the semantic representation of activities in the process, and $n \in \mathbb{N}$ denotes the number of clusters used in the discretization step.

Definition 8 (Time Between Events) Let $\mathrm{tbe} : \mathcal{L} \to \mathcal{L}$ be a function that calculates the time duration between consecutive events within each trace, yielding a transformed log $L' \in \mathcal{L}$, where timestamps are replaced by time differences. For each trace $\sigma = \langle e_1, e_2, \ldots, e_n \rangle \in L$, where $L \in \mathcal{L}$, each event e_i has an associated timestamp t_i, such that $t_i > t_{i-1}$. For all $i > 1$, the time between events d_i is defined as:

$$d_i = \begin{cases} t_i - t_{i-1}, & \text{if } i > 1 \\ 0, & \text{if } i = 1 \end{cases}$$

This transformation results in a modified event log $L' \in \mathcal{L}$, where each event's timestamp is replaced by its time difference d_i relative to the preceding event. The time difference for the first event in each trace is defined as zero.

Definition 9 (Discretization of Numerical Attributes) Let $discretization : \mathcal{L} \times \mathbb{N} \to \mathcal{L} \times \mathcal{K}$ be a function that transforms an event log with numerical attributes to one with categorical labels using DP-K-Means, where $\mathcal{K}$ represents the universe of all possible cluster metadata collections. For an event log $L' \in \mathcal{L}$ that is the output of the tbe function, and cluster count $n \in \mathbb{N}$, the function produces:

$$discretization(L', n) = (L'', K)$$

where $L'' \in \mathcal{L}$ is the transformed event log with all numerical and time-related attributes replaced by their respective cluster labels. $K \in \mathcal{K}$ is a collection of cluster statistics organized by attribute and cluster. Specifically, for each numerical attribute $a \in A$ and each cluster $k \in \{1, 2, \ldots, n\}$, K stores the corresponding mean $\mu_{a,k}$ and standard deviation $\sigma_{a,k}$.

The discretization function transforms numerical attributes in event logs using DP-K-Means clustering, chosen specifically to enforce privacy guarantees based on the sequential composition theorem, thus protecting sensitive data. This function processes all numerical and time-related attributes in the input event log $L' \in \mathcal{L}$, replacing each value with its corresponding cluster label. The function outputs a transformed event log $L'' \in \mathcal{L}$ with discretized numerical attributes, along with set $K \in \mathcal{K}$ that stores statistical properties of each cluster. For every attribute a and cluster k, K contains the mean $\mu_{a,k}$ and standard deviation $\sigma_{a,k}$, which are essential for later stages of event log generation. By adjusting the cluster count n, users can control the granularity of this transformation. The underlying approximation principle allows each numerical attribute's distribution to be represented as a mixture of normal distributions, with each cluster corresponding to a component in the mixture model. Together, these components approximate the original attribute's statistical behavior, as illustrated in Fig. 1.

Definition 10 (Tokenization and Padding) Let $\text{token} : \mathcal{L} \to \mathcal{T}$ be a function that tokenizes and applies padding to each trace in an event log to ensure uniform size.

The final step involves tokenizing and padding all the traces in $L'' \in \mathcal{L}$. Each unique attribute value is mapped to a unique token, thereby transforming all traces into token sequences. Additionally, padding is applied to traces to ensure valid processing during training and sampling. Special tokens mark the start and end of each trace. Additionally, dedicated tokens are introduced to represent missing values. Through these three steps, the input event log $L \in \mathcal{L}$ is transformed into a structured, tokenized format $T \in \mathcal{T}$ suitable for further processing by the RNN. This preprocessing phase ensures the preservation of essential process characteristics while maintaining compliance with DP guarantees, as ensured by applying Theorem 1 to the DP-K-Means clustering step. Figure 3 shows the process of transforming one event with its attributes to the tokenized representation. Since it is possible to transform the DP guarantees from the accounting method to standard DP guarantees, it is possible through Theorem 1 to summarize the guarantees for DP-K-Means and for the training of the model which are calculated by the accounting method.

Training

In this section, the training of the RNN, shown in Fig. 2, is explained in more detail. During training, the model processes the tokenized sequences $T \in \mathcal{T}$ as input. The primary objective of the model is to predict the next tokens in the sequence, thereby respecting the sequential dependencies and contextual information. Each attribute is associated with a distinct output layer to accommodate their individual prediction tasks.

The set Θ represents the complete parameter space of all possible weight and bias configurations for the RNN architecture. In our specific case, $\theta \in \Theta$ denotes the particular

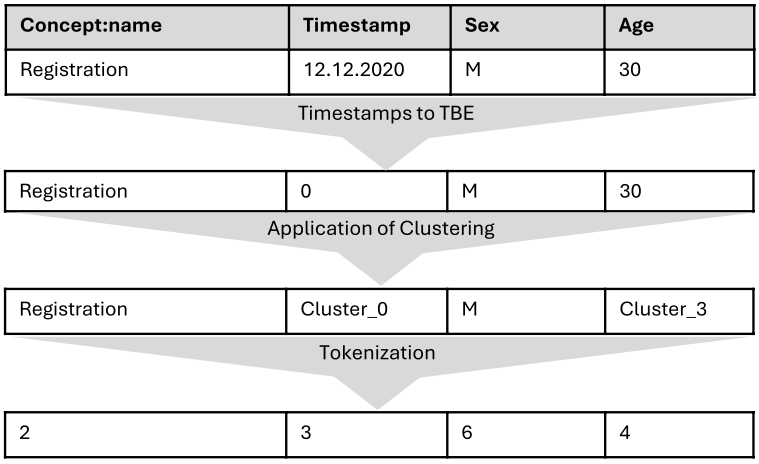

Fig. 3 Visualization of the preprocessing divided into the steps executed for each function. The preprocessing is applied to one event and its attributes. In the PALSYN approach, each event in the trace is tokenized

set of all trainable weights W and biases b in this RNN according to Eq. 1. Training involves calculating the gradients from the cross-entropy loss, which is used to measure the discrepancy between the predicted probabilities of the next token and the actual token. The cross-entropy loss is computed as follows:

$$Loss = -\sum_{t=1}^{|T|}\sum_{c=1}^{C} y_{t,c} \log p_{t,c}, \tag{7}$$

where t denotes the time step in the tokenized sequence T, C the number of possible tokens, $y_{t,c}$ the true label at time t for token class c, and $p_{t,c}$ the predicted probability for class c at time t.

Gradients g are computed for each time step t by back propagating the cross-entropy loss through the network. To ensure DP, each gradient g corresponding to θ is then adjusted according to Eq. 4, implementing DP-SGD (Abadi and Al. 2016). The parameters of the recurrent layers are updated using the differentially private gradients $\widetilde{g}$, which are averaged over the batch size B, yielding $\widetilde{g} \leftarrow \frac{1}{B}\sum_{i=1}^{B} \overline{g}_i$. The update rule in each training step for a parameter θ is given by $\theta \leftarrow \theta - \eta\widetilde{g}$, where η denotes the learning rate. After updating the parameters, the model continues its computations with these differentially private updates for the next iteration of training. The overall DP guarantees for the final model are calculated using the RDP accounting method, as detailed in Section Privacy accounting.

Sampling

Once the model completes training, it can be utilized to generate synthetic tokenized sequences, denoted as $T_s \in \mathcal{T}$, which follow the same format as the real tokenized sequences T. The sampling process starts with an artificial start token for each output layer. Using this token as input, the model predicts the next token based on the learned probability distributions. To introduce variability, predictions are sampled probabilistically rather than deterministically. The newly generated token, along with the previous tokens, serves as input for the next prediction. This iterative process continues until either a predefined stopping condition, such as an end token or a maximum sequence

length, is met. The sampling mechanism ensures that the generated sequences closely resemble the patterns of the training data while maintaining randomness. Notably, the sampling process is conducted entirely without direct access to the original training data. Instead, the model relies solely on structural knowledge of attribute dependencies, which are learned during training. All post-processing steps done after the sampling that are not using data from the real event log are valid transformations that do not compromise the DP guarantees of the sampling. This is given by the post-processing invariance theorem (see Theorem 3).

Post-processing

The post-processing phase in PALSYN reverses the transformations applied during preprocessing. This phase transforms the sampled tokenized sequences $T_s \in \mathcal{T}$ into a fully reconstructed synthetic event log $L_s \in \mathcal{L}$. The post-processing function is defined as follows:

Definition 11 (Post-processing Function) Let $\mathrm{postprocess} : \mathcal{T} \times \mathcal{K} \to \mathcal{L}$ be a function that reverses the transformations applied during preprocessing to convert a tokenized sequence into a synthetic event log. The function is defined as:

$$\mathrm{postprocess}(T, K) = tbe^{-1}(cluster^{-1}(token^{-1}(T), K))$$

where $T \in \mathcal{T}$, $K \in \mathcal{K}$, and $L \in \mathcal{L}$ are specific instances of tokenized sequences, cluster metadata, and an event log, respectively.

Definition 12 (Tokenization Reversal) Let $\mathrm{token}^{-1} : \mathcal{T} \to \mathcal{L}$ be a function that reverses the tokenization and padding process, producing a structurally correct event log consisting only of categorical attributes.
The first step, token^{-1}, reverses the tokenization and padding. Each token in $T \in \mathcal{T}$ is mapped back to its original value using the token dictionary created during preprocessing. Padding tokens, including start and end indicators, are removed during this step. Furthermore, the tokens indicating missing values are also removed.

Definition 13 (Cluster Reversal) Let $\mathrm{cluster}^{-1} : \mathcal{L} \times \mathcal{K} \to \mathcal{L}$ be a function that reconstructs numerical attribute values from cluster labels by sampling from a Gaussian distribution defined by μ and σ for each cluster.
Next, the function $\mathrm{cluster}^{-1}$ restores numerical values by sampling from cluster-specific Gaussian distributions, as defined in $K \in \mathcal{K}$. For each cluster label, a numerical value is sampled from a Gaussian distribution defined by the mean μ and standard deviation σ of the corresponding cluster. This process ensures that the reconstructed numerical attributes approximate the overall statistical properties of the original data, as explained in Fig. 1.

Definition 14 (Time Between Events Reversal) Let $\mathrm{tbe}^{-1} : \mathcal{L} \to \mathcal{L}$ be a function that reconstructs synthetic timestamps as follows:

$$t_i = \begin{cases} t_{\mathrm{start}}, & \text{if } i = 1 \\ t_{i-1} + d_i, & \text{if } i > 1 \end{cases}$$

where t_{start} is sampled from a DP Gaussian distribution of start timestamps from the original event log incorporating Theorem 2 since the real values for the start timestamp are not accessed during preprocessing or training. This is because in Definition 8, the first timestamp is replaced with 0. Furthermore, d_i is defined as the time interval between events.

Finally, the function tbe^{-1} reconstructs timestamps for each event by reversing the time interval transformation. The initial timestamp for each trace is sampled from the DP Gaussian distribution of start timestamps in the original event log $L \in \mathcal{L}$, while subsequent timestamps are calculated by summing the time intervals between consecutive events. This step ensures that the temporal structure of the synthetic data matches that of the original event log and the sequential order of generated events is maintained.

By sequentially applying the three functions token^{-1}, cluster^{-1}, and tbe^{-1}, the post-processing phase transforms the tokenized sequences $T_s \in \mathcal{T}$ into the final synthetic event log $L_s \in \mathcal{L}$. This ensures that the synthetic data preserves the temporal and structural dependencies of the original event log $L \in \mathcal{L}$ while adhering to DP guarantees. Figure 4 illustrates the workflow of this phase and how each transformation is reversed.

Evaluation

To evaluate PALSYN, the tool was implemented in Python using a range of specialized libraries. PM4Py[1] was utilized for event log management, enabling efficient handling of event logs (Berti et al. 2023). TensorFlow[2] was used to implement the deep learning models, while TensorFlow Privacy[3] provided tools for integrating DP into model training (Abadi et al. 2015). Additionally, diffprivlib[4] was employed for implementing DP-K-means and calculating DP bounds for minimum and maximum values (Holohan et al. 2019). The complete PALSYN implementation is publicly available on GitHub[5]. To evaluate the effectiveness of the PALSYN approach, it is benchmarked against PRIPEL (Fahrenkrog-Petersen et al. 2020). Although PRIPEL is primarily an privacy-aware

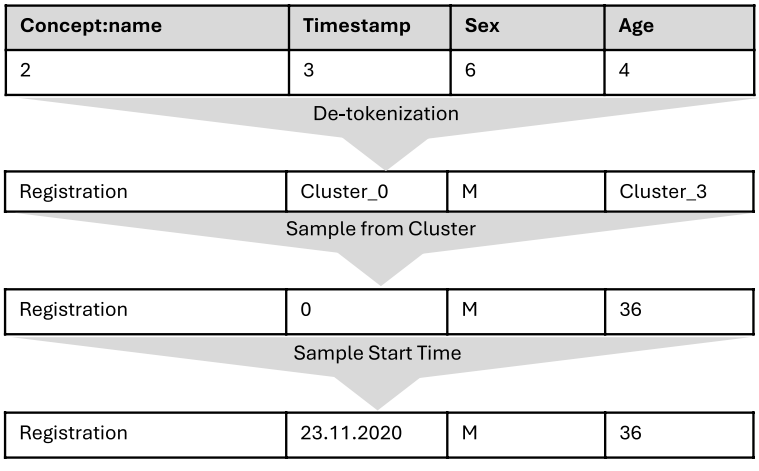

Fig. 4 Visualization of the post-processing, illustrating the sequential application of the post-processing functions

[1] https://github.com/process-intelligence-solutions/pm4py

[2] https://github.com/tensorflow/tensorflow

[3] https://github.com/tensorflow/privacy

[4] https://github.com/IBM/differential-privacy-library

[5] https://github.com/martinkuhn94/PALSYN

framework rather than a data synthesis method, it was chosen because it is, to our knowledge, the only available tool capable of anonymizing multi-perspective event logs under DP. As shown in Table 1, this makes PRIPEL the sole baseline for multi-perspective DP event log anonymization. For our evaluation, we used the PRIPEL implementation provided in PM4Py (Kirchmann et al. 2022).

The evaluation is aligned with the methodological design in Section Methodological design and focuses on the statistical properties defined there, including control flow, timestamps, event and trace attributes, and privacy. Guided by established literature on the assessment of synthetic privacy-preserving data (Hernandez et al. 2023; Liu et al. 2025; Lautrup et al. 2024), the analysis is organized into three dimensions Resemblance, Utility, and Privacy. These dimensions frame the empirical validation and link the methodological goals with the reported results. Resemblance measures how closely the privacy-preserving data replicates the statistical properties of the original data, by analyzing distributions, frequencies, and statistical metrics. It should be noted that PRIPEL assumes attribute independence (Fahrenkrog-Petersen et al. 2020). Consequently, the resemblance evaluation likewise adopts this assumption when analyzing single-attribute distributions, ensuring methodological consistency and comparability between PALSYN and PRIPEL. The utility dimension focuses on the privacy-preserving data's applicability for process mining tasks, determining if meaningful results similar to the real data can be achieved. Thus, the structural dependencies in the control-flow of the privacy-preserving log are evaluated against those of the original log. The privacy dimension assesses the strength of privacy protection through the evaluation of ϵ values provided by the DP mechanism. Lower ϵ values indicate stronger privacy protections. To quantify the DP guarantees, the *Moments Accountant* method described by Abadi et al. is utilized to derive an ϵ if DP-SGD is used during training (Abadi and Al. 2016). This method evaluates the privacy cost at each time the training data is accessed. This cost is then aggregated throughout the duration of the training process. The calculation of ϵ is dependent on the number of data points, batch size, number of epochs, δ and the noise multiplier.

To conduct this evaluation, three event logs of varying sizes and complexities were selected to ensure the robustness of the approach across diverse scenarios. The evaluation aims to test the approach in settings with complex control-flow patterns, multiple data perspectives, and a high number of traces, demonstrating its applicability to real-world data. The Sepsis Cases event log was chosen for its representation of complex medical processes and its high number of event attributes (Mannhardt 2016). This log contains 1050 traces with 846 unique trace variants, reflecting the high variability typical in healthcare settings. It includes 31 distinct event attributes, spanning data types such as timestamps, categorical, boolean, and numerical. The Hospital Billing event log was selected for its high complexity in control-flow and large number of traces. It includes 100,000 traces with 1020 unique trace variants and 10 attributes, comprising timestamps, categorical, boolean, and numerical data types. Finally, the Road Fines event log was included for its simpler control-flow structure but large volume of traces. This log contains 150,370 traces with 231 unique trace variants and a significant number of numerical attributes, along with timestamps, categorical, and boolean attributes. All the event logs are provided in the XES format, a widely used standard in the process mining domain (IEEE 2016). Table 2 provides a summary of the metadata for the utilized

Table 2 Structural comparison of real-world event logs used for model training and evaluation, detailing trace characteristics and attribute type distribution

Metadata	Sepsis Cases	Hospital Billing	Road Fines
Number of Traces	1,050	100,000	150,370
Number of Trace Variants	846	1,020	231
Total Number of Attributes	31	22	16
Float Attributes	3	1	4
Integer Attributes	1	1	3
Boolean Attributes	22	9	0
String attributes	5	10	8
Date time attributes	1	1	1

event logs, showing the number of traces, attributes, and the numeric and categorical attributes.

PALSYN setup

For each of the selected event logs, PALSYN is used to train a model to generate synthetic data based on the training data. The architecture of the model is flexible, allowing for the use of various neural network variants, such as GRU, LSTM, and RNN, as well as their bidirectional configurations. These options provide a range of possibilities for capturing sequential dependencies in the data. However, a comparison in (Camargo et al. 2020) between LSTM and GRU-based methods for next activity prediction shows that, on average, the LSTM-based approach outperforms the variant using GRU layers. Therefore, although multiple configurations involving RNN, GRU, LSTM, and their combinations with bidirectional layers could be explored and are available for implementation, this study focuses solely on LSTM layers to ensure a more straightforward evaluation.

The architecture of the models is configured with a single LSTM layer containing 32 units. Training is performed with a batch size of 128 and an embedding dimension of 128. The number of clusters for numerical attributes is set to 10, and the number of epochs varies depending on the event log. To optimize the amount of padding and training time, only a subset of traces is considered by selecting the top quantile of traces by length, as specified for each event log. For example, for the model of the Road Fines event log, a higher trace quantile and lower amount of epochs is chosen to prevent overfitting since the event log has simple control-flow patterns and a high volume of traces. These configurations ensure that the models are appropriately tailored to the characteristics of the respective event logs. Nevertheless, for each ϵ value, a general architecture is chosen to make the models more comparable. However, in real-world applications, hyperparameter tuning can be performed to optimize the privacy-utility trade-off for specific datasets and use cases (Arous et al. 2023). Since our framework allows specifying other neural networks like RNN and GRU as well as bi-directional layers, it might also be possible that some other architectures are better suited for specific event logs.

Table 3 provides an overview of the parameters used for training the models using PALSYN. Furthermore, the event logs are used without any preprocessing or data cleaning. By preprocessing and cleaning, for example outlier removal, the privacy-utility trade-off could also be improved. For the privacy guarantees, four variants of the privacy parameter ϵ are evaluated: ∞ (no DP), 10, 1, and 0.1. For the privacy parameter δ, the value $\frac{1}{N^{1.1}}$ is chosen, where N is the number of events in the training data, following recommended practices for DP (Dwork and Roth 2014). Thus, in summary, four models

Table 3 Hyperparameter configuration for synthetic log generation models across three different real-world event logs

Parameter	Sepsis Cases	Hospital Billing	Road Fines
Batch Size	128	128	128
Embedding Dimension	128	128	128
LSTM Units	32	32	32
Epochs	10	10	5
Epsilon	[∞, 10, 1, 0.1]	[∞, 10, 1, 0.1]	[∞, 10, 1, 0.1]
Trace Quantile	0.8	0.8	0.9

are trained for each event log, each one having the same architectural parameters but varying regarding the ϵ. One of the models serves as a control model and is trained without privacy guarantees ($\epsilon = \infty$). Furthermore, from each trained model, five synthetic event logs are generated to ensure statistical robustness through repeated sampling. These settings allow for analyzing the effectiveness of the approach across varying levels of privacy protection. During the sampling process, the number of generated traces is matched to the quantity present in the original event log, thereby facilitating a meaningful evaluation. It is important to note that, no hyperparameter tuning was executed. This choices was made to keep the evaluation transparent and comparable across event logs. Nevertheless, targeted tuning and filtering rare variants or handling outliers could further improve results by achieving a more favorable privacy–utility trade-off.

PRIPEL setup

For each selected event log, a privacy-preserving event log was created using PRIPEL, with the objective of achieving the optimal outcome within a reasonable time frame. PRIPEL requires configuration of three key parameters: (1) n, defining the maximum considered trace length, (2) p the pruning parameter for the prefix-tree, (3) ϵ which defines the level of DP. These parameters are set following the recommendations in the publication of Fahrenkrog-Petersen et al. (2020). Thus, the parameter n was set to the 95th percentile of the trace length. The ϵ values were aligned with those used in PALSYN for comparability, specifically $\{0.1, 1, 10\}$. Furthermore, it is important to note that for PRIPEL, an epsilon of infinity does note make sense since PRIPEL does not create synthetic data. Also as pointed out by Schulze et al., PRIPEL introduces privacy risks and thus it is harder to compare it to the PALSN approach regarding the privacy guarantees (Schulze et al. 2024). Furthermore, PRIPEL adheres to the stronger ϵ-DP, while PALSYN uses the relaxed form of (δ, ϵ)-DP (See Sect. Private autoregressive log synthesizer). While the DP guarantees differ a comparative evaluation can still be meaningful. It enables assessment of how each method performs under similar privacy budgets in practice, highlighting trade-offs in resemblance and utility for process mining tasks. For the pruning parameter p, an exhaustive search was conducted to identify the lowest possible value, which according to Fahrenkrog-Petersen et al. lead to higher data utility (Fahrenkrog-Petersen et al. 2020). This involved iterating through p values in the set of powers of two $\{2, 4, 6, 8, 16, 32, 64, \ldots, N\}$ while keeping n and ϵ constant, until a solution was found that executed in under four hours. In (Fahrenkrog-Petersen et al. 2020), the experiments were aborted if execution exceeded one hour. The lowest p value was executable is then selected. For each configuration of valid parameters five logs are created. In this case for the Hospital Billing ($\epsilon = 0.1$) and Road Fines Event Log it was not possible to find reasonable values for p, given the described setting. It was only possible

to find working configurations but they have a unrealsitically high values of p which can be seen in Table 4. Thus, the comparison of these logs with the ones generated by PALSYN can be misleading. Still because of transparency reasons these results are shown. It is crucial to highlight that PRIPEL did not underwent extensive preprocessing, or integration with other techniques that could boost the performance. For example, PRIPEL could be combined with other control-flow anonymization techniques like SaPa to achieve better realism in specific settings (Fahrenkrog-Petersen et al. 2023).

Event log resemblance

This section assesses how closely the privacy-preserving logs replicate the statistical properties of the original logs. Resemblance metrics and their normalization are defined in Section Resemblance metrics, and results for PALSYN and PRIPEL across all utilized event logs and privacy levels are reported in Section Resemblance results.

Resemblance metrics

To quantify the resemblance between the privacy-preserving and original event logs across the four different privacy models, an overall resemblance score is calculated by averaging the resemblance values for all event attributes. This is done for both categorical and numerical attributes. The average resemblance is computed using the following formula:

$$R = \frac{1}{n} \sum_{i=1}^{n} r_i$$

where R denotes the overall resemblance score, n is the total number of attributes, and r_i represents the resemblance score of the i-th attribute.

To determine the resemblance score r_i for individual attributes, different metrics are applied based on the type of attribute. For numerical attributes, the *Kolmogorov-Smirnov (KS)* statistic is used. This measures the distance between the cumulative distribution functions of the original and privacy-preserving data. A lower value of the KS statistic indicates a higher resemblance.

To ensure consistent interpretation across all metrics in this study, the KS statistic is normalized according to $r_i = (1 - \text{Value of the KS statistic})$. A normalized score of $r_i = 1$ indicates perfect resemblance, whereas $r_i = 0$ indicates no resemblance to the original data. For categorical and boolean attributes, the *Hellinger Distance (HD)* is used to measure deviations in frequency distributions. Boolean attributes are treated as categorical attributes in this context. Similar to the KS statistic, the Hellinger distance is

Table 4 PRIPEL configuration parameters for experimental evaluation across three event logs

Event Log	ϵ	k	p
Sepsis Case	0.1	30	16
	1	30	2
	10	30	2
Road Fine	0.1	6	65536
	1	6	65536
	10	6	65536
Hospital Billing	0.1	8	32768
	1	8	64
	10	8	32

normalized so that higher values indicate better resemblance, with $r_i = 1$ representing perfect resemblance and $r_i = 0$ representing no resemblance. These metrics are applied to all trace and event variables except for the concept:name and time:timestamp attribute, which are calculated separately due to their special significance.

Furthermore, it is evaluated how the occurrence of events is distributed in the real data compared to the synthetic data. Therefore, the Hellinger distance is also employed. This metric is also applied to evaluate the distribution of trace length. The throughput time, which measures the time required to complete a trace, is analyzed using the KS statistics, ensuring that the privacy-preserving data accurately reflects the variability and complexity of timings observed in the real event log.

Resemblance results

Figures 5, 6, 7 show the privacy-preserving event logs' resemblance to the original logs for PRIPEL and PALSYN across all privacy budgets. It can be seen that for smaller event logs like Sepsis Case, PRIPEL and PALSYN have mostly similar values for the observed metrics, especially when the average resemblance is considered. For the event logs with more traces, like Road Fines and Hospital Billing, PALSYN achieves better overall resemblance scores and generally higher values on numerical and temporal views (e.g., KS Statistic and Throughput Time), while PRIPEL achieves good results on some categorical distributions at higher ϵ. Important to mention is that PRIPEL was not able to create a good privacy-preserving version of the event logs in several settings. It is also not able to generate the desired amount of traces and, in some occasions, the number of generated traces drops significantly due to high pruning. For both PALSYN and PRIPEL, stronger privacy (smaller ϵ) corresponds to a decrease in resemblance. In contrast, larger ϵ values tend to improve resemblance across datasets.

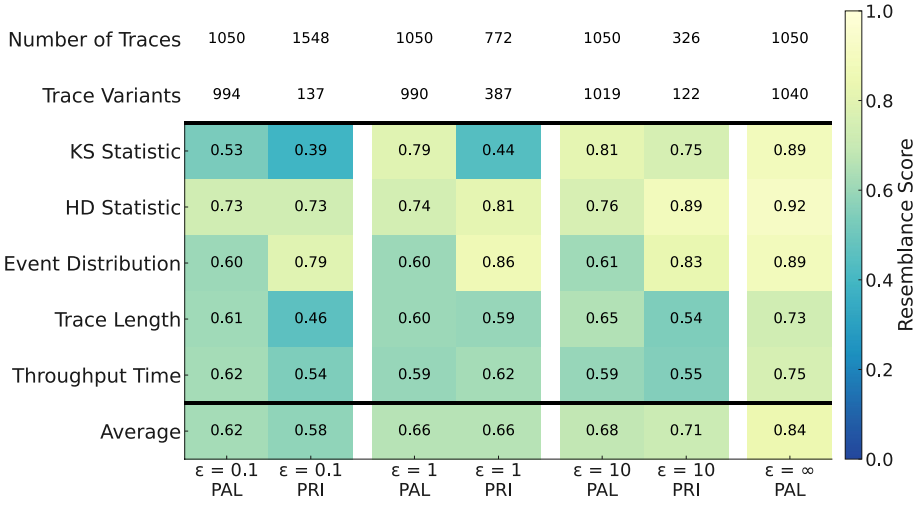

Fig. 5 Resemblance evaluation of the sepsis case event log. The PALSYN approach is denoted by PAL and PRIPEL is denoted by PRI. The metrics number of traces and trace variants are absolute counts (not normalized) and are therefore shown as white cells. All other metrics are normalized and encoded by the color scale where 0.0 (blue) shows low resemblance and 1.0 (yellow) shows high resemblance. Values are means over five independent runs. The average standard deviation across all metrics for all runs is below 0.003 for PALSYN and 0.008 for PRIPEL

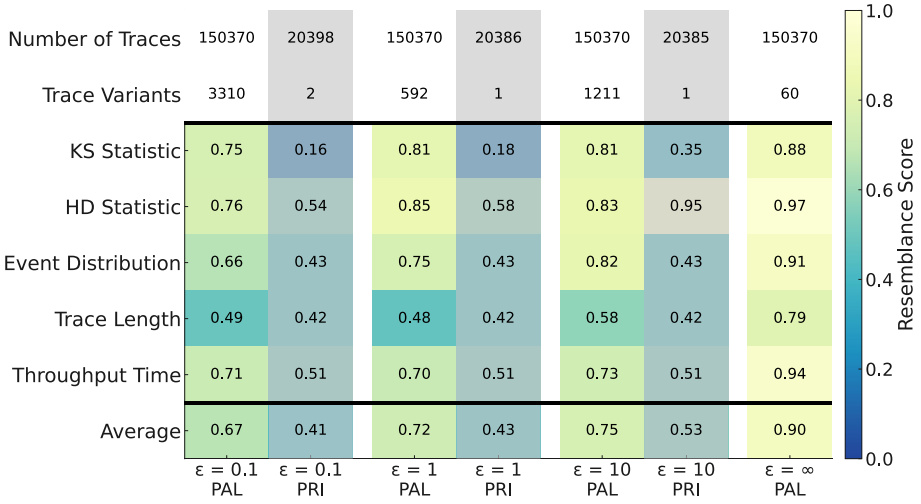

Fig. 6 Resemblance evaluation of the road fine event log. The PALSYN approach is denoted by PAL and PRIPEL is denoted by PRI. The metrics number of traces and trace variants are absolute counts (not normalized) and are therefore shown as white cells. All other metrics are normalized and encoded by the color scale where 0.0 (blue) shows low resemblance and 1.0 (yellow) shows high resemblance. Values are means over five independent runs. The average standard deviation across all metrics for all runs is below 0.004 for PALSYN and 0.002 for PRIPEL. Note, the gray overlay is used to indicate results that are not directly comparable due to high pruning parameters for PRIPEL

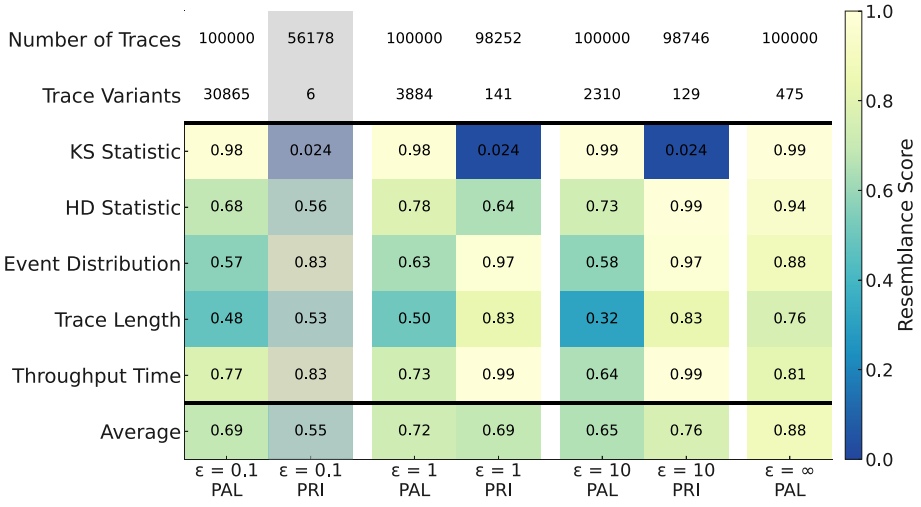

Fig. 7 Resemblance evaluation of the Hospital Billing event log. The PALSYN approach is denoted by PAL and PRIPEL is denoted by PRI. The metrics number of traces and trace variants are absolute counts (not normalized) and are therefore shown as white cells. All other metrics are normalized and encoded by the color scale where 0.0 (blue) shows low resemblance and 1.0 (yellow) shows high resemblance. Values are means over five independent runs. The average standard deviation across all metrics for all runs is below 0.002 for PALSYN and 0.0004 for PRIPEL. Note, the gray overlay is used to indicate results that are not directly comparable due to high pruning parameters for PRIPEL

Process mining utility

This section examines how effectively privacy-preserving logs support core process mining tasks. The evaluation protocol and metrics for discovery, conformance, and trace variant similarity are specified in Section Utility metrics, and empirical results across event logs and privacy budgets are reported in Section Utility results.

Utility metrics

For the evaluation of the utility dimension, typical process mining tasks are considered. First, process models are discovered from the privacy-preserving event logs and compared to real process models obtained from real event logs. Second, conformance checking is applied by using normative models created by experts to verify whether the privacy-preserving event logs conforms to these models. These models, created by Mannhardt in cooperation with domain experts, represent real-life behavior and serve as the ground truth for comparison (Mannhardt 2018). Three models are used, corresponding to the Sepsis Case, Hospital Billing, and Road Fines event logs. This analysis evaluates whether the privacy-preserving event logs align with expert-created process models, indicating its ability to mimic real-world behavior. Furthermore, the Earth Mover's Distance (EMD) is utilized to assess the similarity between the trace variant distributions and thus the stochastic languages of the real and synthetic event log. In process mining, the EMD quantifies the minimal effort required to transform one trace variant distribution into another, using the normalized string edit distances to calculate the similarity between trace variants derived from event logs (Leemans et al. 2021).

To evaluate the utility of the privacy-preserving event logs in terms of discovery, the Inductive Miner implemented in the PM4Py library, with a noise threshold of 0.2 is used to discover a process model based on the privacy-preserving event logs (Leemans et al. 2014). The Heuristics Miner implemented in PM4Py is also applied using the default parameter settings (Weijters and Ribeiro 2011). For clarity, this paper discusses only the Inductive Miner results in detail. The complete outputs of the Heuristics Miner are available in the Zenodo repository for transparency[6]. To evaluate the quality of the discovered process model, the four process mining metrics fitness, precision, generalization, and simplicity are used. Fitness measures how well the discovered model can reproduce the behavior observed in the event log. Precision assesses the extent to which the model restricts itself to only the behavior observed in the event log, penalizing any behavior seen in the model but not in the log. Generalization measures the model's ability to handle unseen but plausible behavior, ensuring it does not overfit to the event log. Simplicity evaluates how straightforward and comprehensible the discovered process model is, favoring less complex representations.

In this evaluation, these metrics are interpreted in the following way. Fitness is used to determine whether the privacy-preserving event logs can be used for a process model discovery, where the discovered model can accurately replicate behavior captured in the real log. A low difference in the fitness score indicates that the discovered model preserves the sequences and structure of the behavior observed in the real log. Precision indicates whether the model accurately reflects the observed behavior, avoiding extraneous or irrelevant activities. A high difference in the precision score suggests that the model discovered on the privacy-preserving event logs either introduces more behavior into the model not seen in the log, or introduces less behavior into the model. Generalization is used to assess whether the model discovered on privacy-preserving event logs generalizes to unseen but plausible behavior, avoiding overfitting to specific patterns in the real log. Additionally, simplicity evaluates the structural complexity of the discovered process model.

[6] https://zenodo.org/records/17571178

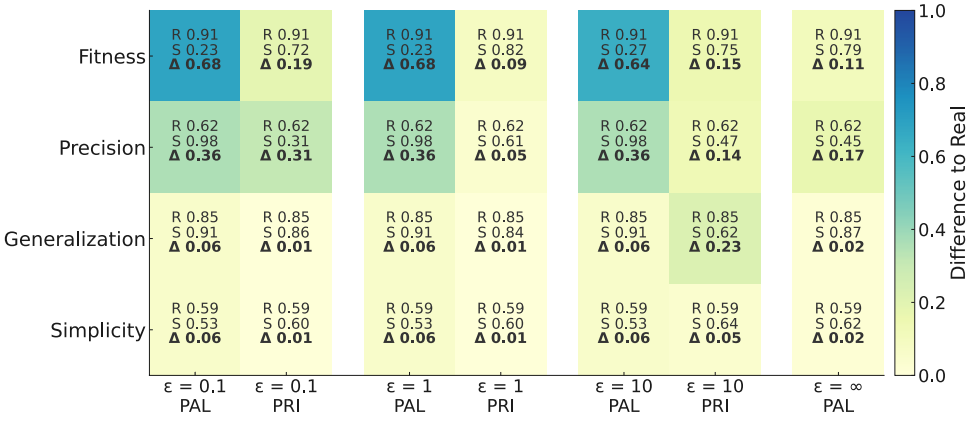

Fig. 8 Utility evaluation of the sepsis Case event log. The PALSYN approach is denoted PAL and PRIPEL by PRI. In each cell, Δ denotes the difference between the values achieved by the real (**R**) and synthetic (**S**) models. Values are means over five independent runs. The average standard deviation of Δ is below 0.007 for PAL and below 0.02 for PRI. Colors encode Δ from 0.0 (yellow, best) to 1.0 (blue, worst)

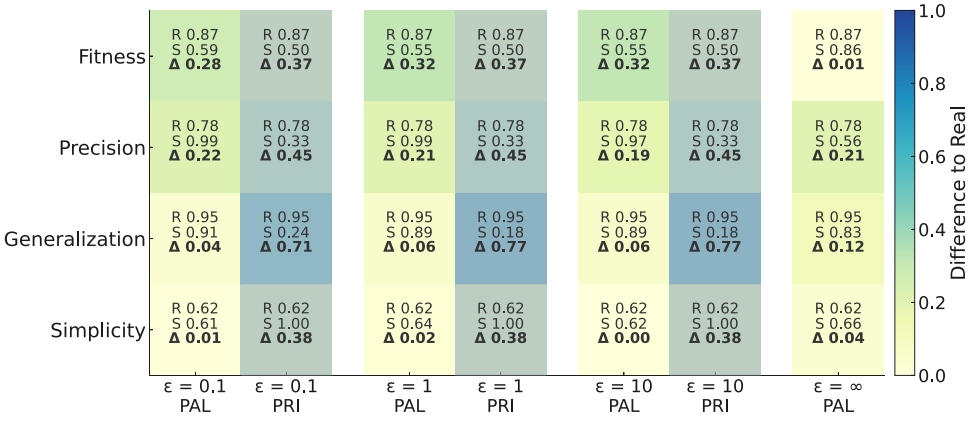

Fig. 9 Utility evaluation of the road fine event log. The PALSYN approach is denoted by PAL and PRIPEL by PRI. In each cell, Δ denotes the difference between the values achieved by the real (**R**) and synthetic (**S**) models. Values are means over five independent runs. The average standard deviation of Δ is below 0.04 for PAL and below 0.006 for PRI. Colors encode Δ from 0.0 (yellow, best) to 1.0 (blue, worst). Note, the gray overlay is used to indicate results that are not directly comparable due to high pruning parameters for PRIPEL

For this evaluation, initially, a Petri net was discovered from the real event log. This net's performance on the real event log was assessed using fitness, precision and generalization, calculated through an alignment. Then on each privacy-preserving event log a model is discovered and fitness, precision and generalization, are calculated through an alignment based on the real log. For simplicity, both the real Petri net and a Petri net discovered from the privacy-preserving event log were compared. The performance of the synthetic process model was evaluated against the real process model by calculating the absolute deviation for the four metrics. This was done for each privacy-preserving event log created by PALSYN and PRIPEL.

Utility results

Figures 8, 9 and 10 show the calculated values for fitness, precision, generalization, and simplicity for the privacy-preserving event logs. Across all datasets, PALSYN generally achieves lower deviations from the real logs in precision, generalization, and simplicity

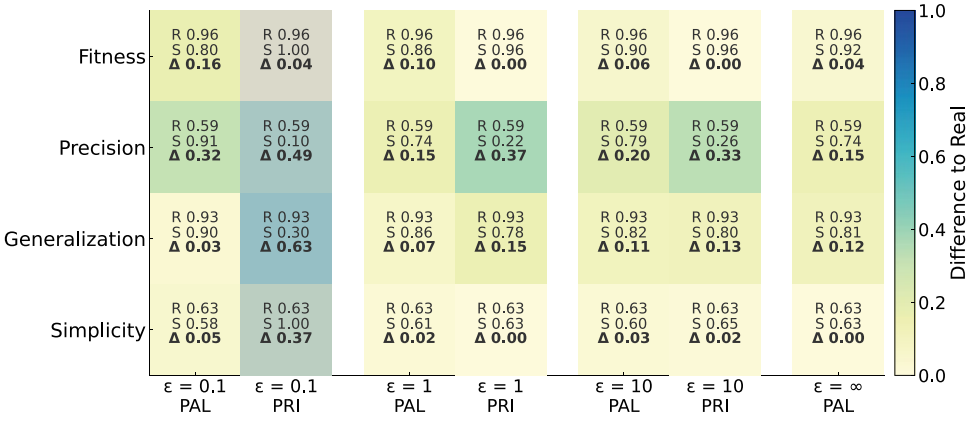

Fig. 10 Utility evaluation of the Hospital Billing event log. The PALSYN approach is denoted by PAL and PRIPEL by PRI. In each cell, Δ denotes the difference between the values achieved by the real (**R**) and synthetic (**S**) models. Values are means over five independent runs. The average standard deviation of Δ is below 0.01 for PAL and below 0.002 for PRI. Colors encode Δ from 0.0 (yellow, best) to 1.0 (blue, worst). Note, the gray overlay is used to indicate results that are not directly comparable due to high pruning parameters for PRIPEL

Table 5 Fitness scores of the privacy-preserving event logs against domain expert-designed normative process models for sepsis case, Road Fines, and Hospital Billing processes across different privacy levels. These models were created by Mannhardt (2018). Each reported value represents the mean aggregated over five independent runs. Across all settings, the standard deviation does not exceed 0.04 for PALSYN and 0.02 for PRIPEL. **Bolded values** indicate results that are not directly comparable due to high pruning parameters for PRIPEL

Event Log	$\epsilon = 0.1$		$\epsilon = 1$		$\epsilon = 10$		$\epsilon = \infty$	Original
	PAL	PRI	PAL	PRI	PAL	PRI	PAL	Log
Sepsis Case	0.40	0.90	0.40	0.95	0.42	0.99	0.78	0.99
Road Fines	0.67	**1.0**	0.79	**1.0**	0.76	**1.0**	0.92	0.99
Hospital Billing	0.53	**1.0**	0.58	0.99	0.50	0.99	0.99	0.99

compared to PRIPEL, particularly at higher ϵ values, indicating better preservation of process structure under weaker privacy constraints. For lower ϵ values and small event logs, PRIPEL tends to achieve smaller fitness deviations, though often at the cost of reduced precision (see Fig. 8).

Furthermore, the quality of the privacy-preserving event logs is assessed using normative models developed by domain experts. For evaluation, fitness is used, and it is calculated using the alignment-based method proposed in (Mannhardt 2018). Table 5 presents the results of this analysis. It can be seen that with higher privacy guarantees, the fitness drops significantly for all three event logs for PALSYN. It can be observed that logs that are more complex, like Hospital Billing and Sepsis Cases, have lower fitness values in comparison to Road Fines. For PRIPEL the fitness values are high across each level of ϵ showing no degradation despite high privacy guarantees.

The quality of the trace variant distributions is assessed using EMD by comparing the privacy-preserving event logs with the original event logs. The results of this analysis can be seen in Table 6. Across all event logs, higher privacy guarantees generally correspond to lower EMD values, indicating greater deviation from the original distributions. For PALSYN, the preservation of trace variant distributions varies notably across datasets. In the Sepsis Case and Hospital Billing event logs, PALSYN achieves comparatively low similarity scores, indicating weaker preservation of the trace variant distributions relative to PRIPEL. In contrast, for the Road Fines event log, PALSYN demonstrates

Table 6 EMD between the trace variant distributions of real and privacy-preserving event logs for the sepsis case, Road Fines, and Hospital Billing datasets across different privacy levels. Higher values indicate higher similarity since the values are normed. Each reported value represents the mean aggregated over five independent runs. Across all settings, the standard deviation does not exceed 0.06 for PALSYN and 0.02 for PRIPEL. **Bolded values** indicate results that are not directly comparable due to high pruning parameters for PRIPEL

Event Log	$\epsilon = 0.1$		$\epsilon = 1$		$\epsilon = 10$		$\epsilon = \infty$
	PAL	PRI	PAL	PRI	PAL	PRI	PAL
Sepsis Case	0.27	0.59	0.27	0.72	0.29	0.68	0.58
Road Fines	0.50	**0.51**	0.60	**0.51**	0.60	**0.51**	0.96
Hospital Billing	0.39	**0.84**	0.48	0.99	0.37	0.98	0.88

stronger performance, achieving higher values for EMD than PRIPEL. It is important to note that logs generated by PRIPEL are generally smaller than the original logs. As a consequence, the relative EMD values for PRIPEL may appear more favorable, but they may also be influenced by the reduced size of the event logs. Therefore, when comparing PALSYN and PRIPEL, the differences in log size should be taken into account.

Discussion

This section discusses the comparative performance of PALSYN and PRIPEL across the analyzed dimensions. The discussion highlights strengths and limitations of both approaches in preserving privacy while maintaining data quality.

Number of traces and trace variants

Regarding the number of traces, PALSYN consistently outperforms PRIPEL in preserving the trace count across varying privacy budgets. While PRIPEL often distorts trace volumes, as seen in all event logs, PALSYN consistently maintains the target trace number. For example, in the Sepsis Case at $\epsilon = 10$, PRIPEL produces only 326 traces (compared to 1050 in the original), whereas PALSYN can synthesize the desired number of traces. Moreover, drastic drops are observed in the Road Fines and Hospital Billing event log. Beyond the total number of traces, PALSYN also outperforms PRIPEL in maintaining a realistic number of trace variants across the analyzed event logs. In cases where the event logs are large or privacy guarantees are strong, it was not possible to find meaningful pruning parameters for PRIPEL. As a result, PRIPEL limits the number of trace variants, sometimes generating only 1, 2, or 6 variants. However, PALSYN also has its limitations. For example, in the Hospital Billing log at $\epsilon = 0.1$, the number of generated trace variants increased significantly, reaching around 30,000 variants, marking the only case with such a drastic deviation (see Fig. 7). Overall, PALSYN performs particularly well on event logs with high-volume and strong privacy guarantees, such as Road Fines, where PRIPELs reliance on pruning parameters leads to overly simplistic behavior and a substantial reduction in the number of trace variants. This represents a disadvantage of PRIPEL, as identifying parameter configurations that yield satisfactory utility can become challenging for large-scale event logs (Fahrenkrog-Petersen et al. 2020; Mannhardt et al. 2019).

Event log properties

Another important aspect in evaluating synthetic event logs is the extent to which they resemble statistical properties of real logs. PALSYN leads or matches PRIPEL in

resemblance scores across most cases (see Figs. 5 to 7), offering more realistic results for the event logs. PRIPELs lower performance can be attributed to to the limited number of trace variants influencing event attributes. In cases where meaningful pruning parameters could be identified for PRIPEL, the average resemblance scores do not differ from PALSYN (see Fig. 5). However, when no meaningful pruning parameters are available, PALSYN tends to achieve better average results (see Figs. 6 and 7). Furthermore, PALSYN consistently achieves better preservation of numerical and temporal attributes across all datasets when compared to PRIPEL. This advantage is reflected in higher resemblance metrics, particularly regarding the KS statistic and throughput time (see Figs. 5 to 7). The integration of DP-K-Means enables effective discretization of numerical attributes, enhancing the attribute similarity to the original log, while still being private. This highlights the effectiveness of the PALSYN methodology by adapting the method proposed by (Xu et al. 2019) (See Sect. Private autoregressive log synthesizer). For the remaining metrics, PALSYN generally performs better in preserving trace length and throughput time, whereas PRIPEL excels in maintaining the event distribution. The weaker performance of PRIPEL in preserving trace length can likely be attributed to its additional pruning parameter n, which limits the maximum trace length considered during processing. Since throughput time is often linked to trace length, this metric is also impacted by the same pruning parameter. This influence becomes particularly pronounced when dealing with large event logs or enforcing strict privacy guarantees. Thus, both pruning parameters can limit the applicability of PRIPEL. But, these limitations may be mitigated by combining PRIPEL with the SaPa technique (Fahrenkrog-Petersen et al. 2023). However, the method relies on local control-flow information, such integration could reduce utility for process mining tasks (Fahrenkrog-Petersen et al. 2023). Overall, this shows that PALSYN has the advantage of being less sensitive to parameter selection, despite applying strong privacy guarantees (see Figs. 9 and 10).

Process mining utility

An important aspect in evaluating synthetic event logs is their usability for process mining tasks. Although PRIPEL shows higher fitness scores, especially when meaningful pruning parameters are available (see Fig. 8), this advantage results from generating only a small number of trace variants and using a similarity function based on Levenshtein distance to align original and noised traces. Thus, the fitness values of PRIPEL needs to be interpreted carefully, because this mechanism can introduce privacy risks as explained in Section Related work. In cases where no suitable pruning parameters could be identified, PRIPEL achieves near perfect fitness across all privacy budgets (see Figs. 9 and 10). Furthermore, such high fitness is questionable when it reflects only one to six preserved variants. In contrast, PALSYN introduces new trace variants to enhance privacy, which inherently reduces fitness. This trade-off is most evident in smaller event logs such as the Sepsis Case, where PALSYN yields consistently low fitness values that may limit practical usability. For larger event logs, however, PALSYN achieves better fitness results. In terms of precision, PALSYN achieves better average values compared to PRIPEL. This trend is particularly evident in the Hospital Billing and Road Fines event logs. The results indicate that the models generated by PRIPEL are often overly simplified, restricting behavior too strongly. In contrast, PALSYN allows for greater behavioral flexibility due to the increased number of trace variants generated. While this improves

coverage, it may also reduce precision by permitting too much behavioral deviation in certain cases.

Also, PALSYN demonstrates stronger generalization capabilities across different event logs and privacy budgets, effectively avoiding overfitting while preserving realistic behavior. For example, in the Road Fines event log, PALSYN maintains generalization differences below 0.06, even under strict privacy settings such as $\epsilon = 0.1$. In contrast, PRIPEL exhibits significant generalization loss in several settings, with differences exceeding 0.70. This is largely due to PRIPELs pruning mechanism and its similarity function, which reduce behavioral diversity in the resulting models. As a result, the model's ability to generalize beyond the training data is limited, increasing the risk of overfitting to specific traces. Thus, PALSYN achieves better adaptability, making it more suitable for practical applications that require a balance between privacy and dynamic behavior of the event log. Overall, model simplicity is better preserved by PALSYN especially for larger event logs. PALSYN maintains a small deviation from the real model across all analyzed event logs, While PRIPEL achieves high deviations for simplicity scores for large event logs. This is often the result of significantly reduced trace variants caused by pruning (see Figs. 9 and 10). Thus, the process models derived from the event logs created with PRIPEL result in overly simplified models. Therefore, this simplicity scores should be interpreted with caution, as they may indicate a loss of meaningful behavioral detail rather than true model interpretability. To properly assess the practical relevance of simplicity, the resulting process models should be evaluated in real-world scenarios, including visual inspection by domain experts.

An important criterion is alignment with expert-created normative models, with Table 5 revealing clear differences between the two methods. For PALSYN, fitness values decrease as privacy guarantees increase describing a classical privacy-utility trade-off. This drop is especially pronounced for more complex event logs such as Hospital Billing and Sepsis Case. In contrast, PRIPEL maintains consistently high fitness scores across all privacy levels, with minimal or no degradation, even under strong privacy constraints. While these results suggest that PRIPEL aligns better with normative models, this behavior is largely due to the limited number of trace variants it generates in combination with the similarity function. The resulting high similarity to the original log increases fitness but also raises privacy concerns. PALSYN, on the other hand, introduces a broader behavioral range, which may lead to lower fitness but offers stronger privacy protection and better generalization for complex processes. Therefore, alignment should be interpreted with consideration of the trade-off between privacy and utility. It should be noted that for PALSYN ($\epsilon = \inf$) a good alignment with the normative models can be achieved, thus showing that the architecture in general is capable of learning process behavior correctly and the deviation can be attributed to the noise inserted by using DP. This is also true for the process mining utility analysis covering fitness, precision, generalization and simplicity. This indicates that PALSYN is capable of learning and reproducing realistic process structures. However, its performance under DP-induced noise could be enhanced. For PRIPEL it should be noted that since it is a privacy-aware framework, some mentioned limitations could be addressed by combining it with other methods like SaCoFa or SaPa which are specifically developed for the process mining domain (Fahrenkrog-Petersen et al. 2023). On the other hand, the same is true for PALSYN, through

log filtering or hyperparameter tuning, an improved privacy-utility trade-off could theoretically be achieved.

Limitations and future work

Regarding privacy, PRIPEL and PALSYN implement trace-level DP, treating each trace as an independent unit. However, this protection does not extend to user-level privacy, where multiple traces may belong to a single individual. As a result, an adversary with auxiliary information might still be able to reconstruct behavioral patterns across multiple traces, even if each individual trace is protected in isolation. Addressing such cumulative privacy risks remains an important direction for future work, for example, by extending DP-SGD or adopting more advanced privacy accounting methods. It is also important to recognize that the choice of ϵ depends on the specific event log and the intended use case. Different values may be considered appropriate or legally acceptable in different contexts. Nonetheless, legal compliance, such as adherence to the GDPR, must always be confirmed before sharing data, even when formal DP guarantees are in place (GDPR). A further limitation lies in the comparability of the privacy guarantees provided by the two methods. PALSYN and PRIPEL apply different variants of DP. PALSYN employs relaxed guarantees, while PRIPEL relies on stronger formulations. Therefore, the ϵ values reported for each method are not directly comparable and should only be interpreted as indicative. Despite this, the comparison remains valuable for understanding the practical strengths and limitations of each method, as discussed in earlier sections. To ensure robust protection, thorough risk assessments should be conducted. These should include empirical testing using membership inference attacks and simulations with more realistic adversary models (Giomi et al. 2023).

Regarding the evaluation, there is a clear need for a standardized framework to assess the utility of synthetic event logs. While this study employed several widely used metrics, they do not fully capture all relevant aspects, and some may not be perfectly suited for comparing privacy-preserving logs with their original counterparts. Future work should therefore focus on identifying and consolidating evaluation metrics that are specifically tailored to this task, ideally integrating them into a unified assessment framework. The evaluation in this study was conducted using five runs per configuration. Although a larger number of repetitions would improve statistical robustness, this was not feasible due to the runtime constraints of the employed algorithms and the computational cost associated with the chosen metrics. Furthermore, only three event logs were used for evaluation. To draw more generalizable conclusions about the performance of the proposed methods, a broader and more diverse set of event logs should be included in future studies.

Furthermore, while PALSYN maintains high utility for process mining tasks when no DP is applied, its utility drops significantly if DP is applied. This limitation could be addressed by exploring model enhancements, optimization techniques, and architectural changes. In this context, integrating attention mechanisms could be a promising direction to further optimize this trade-off (Vaswani et al. 2017). In addition, since no parameter tuning was performed for PALSYN in this study, future work could implement automated tuning procedures to maximize utility for given privacy constraints. Another promising direction to enhance PALSYNs utility under strict privacy settings is the integration of conditional sampling. This could guide log generation toward desired

outcomes without relying on information derived from the original event log, thereby ensuring compliance with privacy requirements. Techniques proposed by Graziosi et al. (Graziosi et al. 2024) could be adapted for this purpose. Moreover, the privacy–utility trade-off might be improved by combining PALSYN with established privacy-aware framework. One possible approach would be to first apply anonymization methods, such as those described in (Elkoumy and Dumas 2022; Fahrenkrog-Petersen et al. 2020, 2023; Mannhardt et al. 2019), before using the anonymized data as training input for PALSYN, and then generating privacy-preserving synthetic logs.

Another important aspect concerns the practical relevance of the privacy-preserving event logs. In this work, utility was assessed exclusively through automated process mining metrics. However, no evaluation involving domain experts or end users was conducted to determine whether the generated logs are interpretable, meaningful, or useful from a practitioner's perspective. Such evaluations could provide valuable insights into the real-world applicability of the generated data. Finally, real-world pilot studies are needed in sensitive domains such as healthcare or financial compliance to evaluate the practical feasibility and impact of sharing privacy-preserving event logs. Since the event logs used in this work were already public and anonymized, the evaluation did not account for applying the evaluated methods to non-anonymized event logs. Future work should therefore focus on applying these methods to real-world, high-risk contexts to test their robustness, utility, and compliance in practice.

Conclusion

This paper presented PALSYN, a novel approach for generating differentially private synthetic event logs while preserving the statistical characteristics of multi-perspective event logs. By leveraging deep learning techniques and incorporating DP, PALSYN addresses key limitations of existing methods and provides formal privacy guarantees on the trace-level. The evaluation demonstrates that PALSYN can generate synthetic event logs resembling the original data across various metrics. However, the results highlight the privacy–utility trade-off, where stricter privacy settings introduce noise, affecting process mining utility. Additionally, the complexity and size of the original event log impact synthetic event log quality, with smaller logs generally showing lower utility under DP settings.

Moreover, this study benchmarked PALSYN with PRIPEL to analyze the advantages and disadvantages of both methods. From a practical perspective, the choice between PALSYN and PRIPEL depends on the use case and log characteristics. PALSYN is particularly suited for large event logs with high trace numbers. Its ability to produce the desired number of traces, preserve temporal and numerical features, and avoid manual configuration can make it valuable for sharing privacy-preserving logs. In contrast, PRIPEL may be a better option for smaller event logs that require anonymization without generating entirely new behavior. Its prefix-tree approach retains fewer but highly similar variants and may be appropriate when strict adherence to original trace structures is prioritized over generalization or behavioral richness and if the introduced privacy risks are acceptable.

Author contributions

In the publication "PALSYN: A Method for Synthetic Multi-Perspective Event Log Generation with Differential Private Guarantees," Martin Kuhn, Joscha Grüger, and Ralph Bergmann contributed as follows: Martin Kuhn led the conceptualization, methodology, software implementation, validation, investigation, formal analysis, and data curation

for the PALSYN approach. He was responsible for conducting the experimental evaluation and analyzing the results. Additionally, Martin Kuhn prepared the original draft of the manuscript, created visualizations, and managed the public code repository and evaluation data. Supervision was jointly provided by Ralph Bergmann and Joscha Grüger, who guided the research direction. They also contributed to reviewing and critically editing the manuscript, ensuring scientific rigor and clarity of presentation. The work was conducted under the academic mentorship of Ralph Bergmann, who provided project administration and resources necessary for completing the research. CRediT Statement- Conceptualization: MK, RB, JG- Methodology: MK- Software: MK- Validation: MK- Formal Analysis: MK- Investigation: MK- Resources: RB- Data Curation: MK- Writing – Original Draft: MK- Writing – Review & Editing: RB, JG- Visualization: MK- Supervision: RB, JG- Project Administration: RB- Funding Acquisition: RB

Funding
Open Access funding enabled and organized by Projekt DEAL. The research is funded by the German Federal Ministry of Research, Technology and Space (BMFTR) and NextGenerationEU (European Union) in the project KI-AIM under the funding code 16KISA115K. The publication was funded / supported by the Open Access Fund of Universität Trier and by the German Research Foundation (DFG).

Data availability
The source code of PALSYN is available in our public GitHub repository under the following link, https://github.com/martinkuhn94/PALSYN. All evaluation results, including raw measurements and statistical analyses, are accessible through Zenodo under the following link, https://zenodo.org/records/17571178.

Declarations

Ethical approval
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 30 April 2025 / Accepted: 1 December 2025
Published online: 20 December 2025

References

Aalst WMP (2016) Data science in action. Springer, Cham, Switzerland
Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X et al (2015). TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org (https://www.tensorflow.org/
Abadi M, Al.: Deep learning with differential privacy. In: (2016) Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. CCS'16, pp. Association for Computing Machinery, New York, USA, pp 308–318
Arous A, Guesmi A, Hanif MA, Alouani I, Shafique M (2023): Exploring machine learning privacy/utility trade-off from a hyperparameters lens. In: 2023 International joint conference on neural networks (IJCNN). pp 01–10
Bergami G: Fast Synthetic Data-Aware Log Generation for Temporal Declarative Models (2023) In: Proceedings of the 6th joint workshop on graph data management experiences & systems (grades) and network data analytics (NDA). ACM, Seattle WA USA, pp 1–9. Accessed 2023-10-19 https://dl.acm.org/doi/10.1145/3594778.3594881
Berti A, Zelst S, Schuster D (2023) Pm4py: a process mining library for python. Software Impacts 17:100556. https://doi.org/10.1016/j.simpa.2023.100556
Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D: Language models are few-shot learners (2020). In: Proceedings of the 34th international conference on neural information processing systems. NIPS'20. Curran Associates Inc., Red Hook, NY, USA
Burattin A, Re B, Rossi L, Tiezzi F (2022): A Purpose-Guided Log Generation Framework. In: Di ciccio, C., Dijkman, R., Río ortega, A., Rinderle-Ma, S. Business process management. Lecture Notes in Computer Science, pp. Springer, Cham, pp 181–198
Burattin A, Sperduti A (2010): PLG: a framework for the generation of business process models and their execution logs. In: BPM 2010 workshops, vol 66. pp 214–219
Camargo M, Dumas M, González-Rojas O: Learning accurate lstm models of business processes (2019). In: Business process management: 17th international conference, BPM 2019, Vienna, Austria, September 1– 6, 2019, proceedings. Springer, Berlin, Heidelberg, pp 286–302
Camargo M, Dumas M, Rojas OG (2020) Discovering generative models from event logs: data-driven simulation vs deep learning. peerj Computer Science 7
Cho K, Merriënboer B, Bahdanau D, Bengio Y: On the properties of neural machine translation: Encoder–decoder approaches (2014). In: Wu, D., Carpuat, M., Carreras, X., Vecchi, E.M. Proceedings of SSST-8, eighth workshop on syntax, semantics and structure in statistical translation. Association for Computational Linguistics, Doha, Qatar, pp 103–111. https://doi.org/10.3115/v1/W14-4012
Ciccio CD, Bernardi ML, Cimitile M, Maggi FM: Generating event logs through the simulation of declare models (2015). In: Workshop on enterprise and organizational modeling and simulation. Springer, Cham, Switzerland, pp 20–36
Dwork C, Kenthapadi K, McSherry F, Mironov I, Naor M: Our data, ourselves: Privacy via distributed noise generation (2006). In: Advances in cryptology - EUROCRYPT 2006. Springer, Berlin Heidelberg, Germany, pp 486–503

Dwork C, Roth A (2014) The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. 9(3–4):211–407
Elkoumy G, Dumas M: Libra: High-utility anonymization of event logs for process mining via subsampling (2022). In: 4th international conference on process mining (ICPM). pp 144–151. https://doi.org/10.1109/ICPM57379.2022.9980619
Elkoumy G, Pankova A, Dumas M: Mine me but don't single me out: Differentially private event logs for process mining (2021). In: 3rd international conference on process mining (ICPM). pp 80–87. https://doi.org/10.1109/ICPM53251.2021.9576852
Esgin E, Karagoz P: Process profiling based synthetic event log generation (2019). In: International conference on knowledge discovery and information retrieval. pp 516–524
Evermann J, Rehse J-R, Fettke P (2017) Predicting process behaviour using deep learning. Decision Support Systems 100:129–140. https://doi.org/10.1016/j.dss.2017.04.003. Smart Business Process Management
Fahrenkrog-Petersen SA, Aa H, Weidlich M: PRIPEL: Privacy-preserving event log publishing including contextual information. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (2020) Business process management. Springer, Cham, pp 111–128
Fahrenkrog-Petersen SA, Kabierski M, van der Aa H, Weidlich M (2023) Semantics-aware mechanisms for control-flow anonymization in process mining. Information Systems 114:102169. https://doi.org/10.1016/j.is.2023.102169
GDPR. Official journal of the European union. Accessed: 29.04.2025
Gers F, Schmidhuber J, Cummins F (2000) Learning to forget: continual prediction with lstm. Neural Computation 12:2451–2471
Giomi M, Boenisch F, Wehmeyer C, Tasnádi B (2023). A unified framework for quantifying privacy risk in synthetic data. https://petsymposium.org/popets/2023/popets-2023-0055.php
Graziosi R, Ronzani M, Buliga A, Di francescomarino C, Folino F, Ghidini C, Meneghello F, Pontieri L: Generating the traces you need: A conditional generative model for process mining data (2024). In: 6th international conference on process mining (ICPM). pp 25–32. https://doi.org/10.1109/ICPM63005.2024.10680621
Grüger J, Geyer T, Jilg D, Bergmann R: Sample: A semantic approach for multi-perspective event log generation (2023). In: Process mining workshops. Springer, Cham, pp 328–340
Hernandez M, Epelde G, Alberdi A, Cilla R, Rankin D (2023) Synthetic tabular data evaluation in the health domain covering resemblance, utility, and privacy dimensions. Methods Of Information In Medicine 62
Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Computation 9:1735–1780
Holohan N, Braghin S, Aonghusa PM, Levacher K (2019). Diffprivlib: the IBM differential privacy library. CoRR abs/1907.02444 (https://arxiv.org/abs/1907.02444
IEEE (2016) Standard for eXtensible event stream (XES) for achieving interoperability in event logs and event streams. IEEE Std 1849–2016 1–50
Jordon J, Szpruch L, Houssiau F, Bottarelli M, Cherubin G, Maple C, Cohen SN, Weller A (2022). Synthetic data – what, why and how?
Karras T, Aittala M, Laine S, Härkönen E, Hellsten J, Lehtinen J, Aila T: Alias-free generative adversarial networks. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., Vaughan, J.W. (2021) Advances in neural information processing systems, vol 34. Curran Associates, Inc., Red Hook, NY, USA, pp 852–863
Kirchmann H, Fahrenkrog-Petersen SA, Kabierski M, Aa H, Weidlich M: Privacy-preserving process mining with PM4Py (extended abstract) (2022). In: ICPM doctoral consortium / demonstration track. pp 85–89
Kuhn M, Grüger J, Matheja C, Rivkin A: Data petri nets meet probabilistic programming (2024). In: 22nd International conference on business process management, vol 14940. Springer, Cham, Switzerland, pp 21–38
Lautrup AD, Hyrup T, Zimek A, Schneider-Kamp P (2024) Systematic review of generative modelling tools and utility metrics for fully synthetic tabular data. ACM Comput. Surv. 57(4). https://doi.org/10.1145/3704437
Leemans SJJ, Fahland D, Aalst WMP: Discovering block-structured process models from event logs containing infrequent behaviour (2014). In: Business process management workshops (BPM 2013). Lecture Notes in Business Information Processing, vol. 171. Springer, Cham, pp 66–78. https://doi.org/10.1007/978-3-319-06257-0_6
Leemans SJJ, van der Aalst WMP, Brockhoff T, Polyvyanyy A (2021) Stochastic process mining: earth movers' stochastic conformance. Information Systems 102:101724. https://doi.org/10.1016/j.is.2021.101724
Li K, Yang S, Sullivan TM, Burd RS, Marsic I (2024) Processgan: generating privacy-preserving time-aware process data with conditional generative adversarial nets. ACM Trans. Knowl. Discov. Data 18(9). https://doi.org/10.1145/3687464
Lin L, Wen L, Wang J: Mm-pred: A deep predictive model for multi-attribute event sequence (2019). In: Berger-Wolf, T.Y., Chawla, N.V. Proceedings of the 2019 SIAM international conference on data mining, SDM 2019, Calgary, Alberta, Canada, May 2-4, 2019. SIAM, Philadelphia, PA, USA, pp 118–126. https://doi.org/10.1137/1.9781611975673.14
Liu Y, Acharya UR, Tan JH (2025) Preserving privacy in healthcare: a systematic review of deep learning approaches for synthetic data generation. Computer Methods And Programs In Biomedicine 260:108571. https://doi.org/10.1016/j.cmpb.2024.108571
Mannhardt F (2016). Sepsis Cases - event log (version 1)
Mannhardt F (2018). Multi-perspective process mining. PhD thesis, Technische Universiteit Eindhoven
Mannhardt F, Koschmider A, Baracaldo N, Weidlich M, Michael J (2019) Privacy-preserving process mining: differential privacy for event logs. Business Information Systems Engineering 61:595–614
Mironov I: Rényi differential privacy (2017). In: IEEE 30th computer security foundations symposium (CSF). pp 263–275. https://doi.org/10.1109/CSF.2017.11
Munoz-Gama J, Others (2022) Process mining for healthcare: characteristics and challenges. Journal Of Biomedical Informatics 127:103994
Ponomareva N, Al. (2023) How to dp-fy ml: a practical guide to machine learning with differential privacy. Journal Of Artificial Intelligence Research 77:1113–1201
Rafiei M, Wangelik F, Pourbafrani M, Aalst WMP: Travag: Differentially private trace variant generation using gans (2023). In: Research challenges in information science: information science and the connected world. Springer, Cham, pp 415–431
Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B: High-resolution image synthesis with latent diffusion models (2022). In: IEEE/CVF conference on computer vision and pattern recognition (CVPR). pp 10674–10685
Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Networks 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003
Schulze M, Zisgen Y, Kirschte M, Mohammadi E, Koschmider A: Differentially private inductive miner (2024). In: 6th international conference on process mining (ICPM). pp 89–96. https://doi.org/10.1109/ICPM63005.2024.10680684

Su D, Cao J, Li N, Bertino E, Jin H: Differentially private k-means clustering (2016). In: Proceedings of the sixth ACM conference on data and application security and privacy. CODASPY'16, pp. Association for Computing Machinery, New York, NY, USA, pp 26–37

Taymouri F, Rosa ML, Erfani S, Bozorgi ZD, Verenich I: Predictive business process monitoring via generative adversarial nets: The case of next event prediction. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (2020) Business process management. Springer, Cham, pp 237–256

van Dun C, Moder L, Kratsch W, Röglinger M (2023) Processgan: supporting the creation of business process improvement ideas through generative machine learning. Decision Support Systems 165:113880. https://doi.org/10.1016/j.dss.2022.113880

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I: Attention is all you need. In: Guyon, I., Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (2017) Advances in neural information processing systems, vol 30. pp 5998–6008

Wangelik F, Rafiei M, Pourbafrani M, Aalst WMP (2025). Releasing differentially Private event logs using generative models. https://arxiv.org/abs/2504.06418

Weijters AJMM, Ribeiro JTS: Flexible heuristics miner (fhm) (2011). In: IEEE symposium on computational intelligence and data mining (CIDM). pp 310–317. https://doi.org/10.1109/CIDM.2011.5949453

Xu L, Skoularidou M, Cuesta-Infante A, Veeramachaneni K (2019) Modeling tabular data using conditional GAN. Curran Associates Inc., Red Hook, NY, USA

Publisher's Note