GEDI: Generating Event Data with Intentional Features for Benchmarking Process Mining

Andrea Maldonado[1,3], Christian M. M. Frey[2], Gabriel Marques Tavares[1,3], Nikolina Rehwald[1], and Thomas Seidl[1,2,3]

[1] Database Systems and Data Mining, LMU Munich, Munich, Germany
{maldonado, tavares, seidl}@dbs.ifi.lmu.de
[2] Fraunhofer IIS, Division Analytics, Nuremberg, Germany
{christianmaxmike, nikolina.rehwald}@gmail.com
[3] Munich Center for Machine Learning, Munich, Germany

Abstract. Process mining solutions include enhancing performance, conserving resources, and alleviating bottlenecks in organizational contexts. However, as in other data mining fields, success hinges on data quality and availability. Existing analyses for process mining solutions lack diverse and ample data for rigorous testing, hindering insights' generalization. To address this, we propose Generating Event Data with Intentional features, a framework producing event data sets satisfying specific meta-features. Considering the meta-feature space that defines feasible event logs, we observe that existing real-world datasets describe only local areas within the overall space. Hence, our framework aims at providing the capability to generate an event data benchmark, which covers unexplored regions. Therefore, our approach leverages a discretization of the meta-feature space to steer generated data towards regions, where a combination of meta-features is not met yet by existing benchmark datasets. Providing a comprehensive data pool enriches process mining analyses, enables methods to capture a wider range of real-world scenarios, and improves evaluation quality. Moreover, it empowers analysts to uncover correlations between meta-features and evaluation metrics, enhancing explainability and solution effectiveness. Experiments demonstrate GEDI's ability to produce a benchmark of intentional event data sets and robust analyses for process mining tasks.

Keywords: Data Generation · Benchmarking · Event Log Features · Hyperparameter Optimization

1 Introduction

In today's digital age, data plays a pivotal role in organizational decision-making. Information systems meticulously record business events, creating extensive event data (ED). Process Mining (PM) aims to enhance operational processes through ED analysis, providing insights into performance, bottlenecks, and improvement opportunities. [1]. However, the success of PM critically hinges on ED quality, as highlighted by Andrews et al. [4]. High-quality ED may still

not fully represent relevant business processes, as noted by Beerepoot et al. [8], or may face GDPR. Beyond data quality, Weytjens et al. [36] highlight the pervasive lack of ED standardization, citing issues like undocumented preprocessing, reliance on non-public knowledge, and flawed training-test separation, which hinder generalizable process mining solutions. This standardization gap affects various PM tasks [11, 31], including process discovery (PD), and predictive monitoring [36]. Additionally, [33] highlights the imperative need for process diversity, but existing frameworks often struggle with small sample sizes, limiting result robustness [22]. The scarcity of real ED and inherent lack of reference models further complicate the analysis [23, 14], as available ED often fail to capture the full event log space due to their limited quantity and dubious quality.

Jouck et al. [23] advocate using artificial ED for empirical evaluation to compare model quality and understand how ED characteristics affect method efficiency. To enable meaningful comparisons of process mining solutions, the ED used needs to be reproducible and encompass potential characteristics beyond the current confines. The absence of standardization and diversity in benchmark data limits the reliability of process mining solutions and the validity of our community's research findings. Hence, we propose a framework for creating reproducible and comprehensive benchmark ED, addressing the deficiencies of randomly generated and scarce real-world ED. In our work, we focus on the control-flow perspective of event logs. Nevertheless GEDI can be extended to additional event attributes, such as resource or event payloads, among others.

Our proposed methodology combines established ED-level features, which cover statistical and entropy aspects, to generate more comprehensive event datasets, as shown in fig. 1. On the left, a coordinate grid fills the feature space with potential combinations of feature values, as input for GEDI. *ED Feature 1* and *ED Feature 2* represent two characteristics, i.e., features of event datasets. Using Bayesian optimization, we produce an event dataset (EDS) for each point in the grid. Orange diamonds represent generated EDS with desired feature values, while grey crosses represent unfeasible feature values, due to definition constraints. E.g. the value for the minimum number of events for a process execution cannot be larger than the maximum number of events for a process execution of the same event log. For comparative evaluation purposes, on the right side of fig. 1 we depict a collection of real ED, as blue circles, sourced from publicly available databases in the feature space. By combining both sides, we establish a novel comprehensive benchmark ED collection and compare descriptors of real and generated EDS. This process enriches the space reproducibly, covering a larger area than the current real ED, as depicted in the middle of fig. 1. Our framework enables the creation of a comprehensive EDS collection, which can be used for benchmarking new solutions and stressing algorithms in corner cases. Using GEDI for benchmarks, we reach a deep understanding of how feature sets relate to evaluation metrics, allowing the community to create methods tailored for specific downstream tasks. By defining the scope of GEDI benchmarks — including features, metrics, and candidates for a task — we can find approaches that generally perform well or excel in specific areas, including

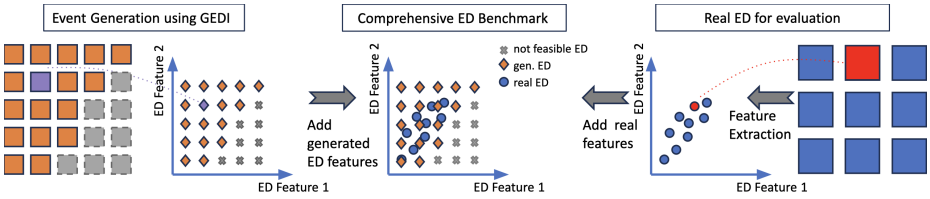


Fig. 1: GEDI [d͡ʒˈɛdaɪ] generates ED which enriches the current event log collection space in terms of multiple features.

multiple real-world scenarios. Additionally, GEDI can generate event data similar to real-world cases and comply with GDPR.

This paper addresses the challenge of obtaining high-quality ED for algorithm evaluation and their limited representation in the event log space by proposing two contributions: (1) A framework for intentional ED generation leveraging event log features and optimization to intentionally populate a grid in the feature space, and thus unexplored regions. (2) A properly standardized comprehensive benchmark ED for comparative algorithm evaluation. Consequently, we offer the community reproducible unexplored ED. Our results assess the benchmark ED's quality by measuring the distance between desired and resulting features and demonstrate the application with PD algorithms. Additionally, we map how ED features correlate with algorithm performance, shedding light on current challenges that require attention.

2 Related Work

We provide a literature overview for generating event log data (cf. section 3.1).

Simulation-based. These approaches rely on stochasticity for modeling a business process and/or event log generation. In [15], random processes are modeled using a stochastic context-free grammar, with the user controlling process complexity parameters such as the occurrence probability of well-known process patterns used as productions. [13] extends this approach to produce multi-perspective event logs and event streams with concept drifts. [14] proposes a framework for event log generation tailored for downstream tasks like process discovery, using guided simulation to incrementally generate traces from an existing process model. [9] introduces a recovering framework to generate deterministic traces, which maximize the similarity between a recovered log and a record that reflects the real occurrence of activities. [7] and [23] present event log generators with various modules: E.g. Process modeling, stochastic sampling, simulation, as well as user intervention. Several of these contributions [9, 13, 23, 15] use probability values, whereas our approach focuses on generating processes with intentional, data-driven features via optimization. As opposed to [7, 9, 13, 14], GEDI requires neither an input log/model nor the discovery of stochastic/process models to generate new ED with desired characteristics. In [23], the

authors generate random process trees and simulate them into corresponding event logs. Adjustable parameters offer some control over the resulting event log via process model characteristics. Nevertheless, these parameters are limited mostly to control-flow characteristics of process models, and thus indirectly to inter-activities relations. Therefore, the link between selected parameters and the resulting (meta-)features on an event log level is unclear. Similarly, GEDI uses features that directly describe ED. Beyond that, GEDI extends features impacts on the generated event log without a process model. GEDI's novelty lies in optimizing a data generator's input parameters so that event data with desired features will be generated.

Augmentation-based. In [24], the authors enhance next activity prediction by iteratively adding random noise-based transformations to input ED, rather than generating new datasets. [20] presents a framework that generates counterfactual sequences via evolutionary algorithms without requiring domain knowledge. The authors of [32] mine decisions from time series data using features for decision-making, similar to our feature-based approach. Recently, [19] proposed a privacy-preserving framework for multi-perspective process mining through data generalization, maintaining dependencies, and ensuring k-anonymity. Unlike [19, 24], GEDI does not require input event logs. Instead of generalizing ED behavior or enriching ED with noise-based transformations, GEDI comprehensively explores the (meta-)feature space. Similar to [20, 19, 32], our approach focuses on fulfilling objectives, like producing desired feature values. While [20] targets one counterfactual scenario at a time and [19] ensures k-anonymity and dependency preservation, GEDI can simultaneously optimize multiple criteria constraints regarding the (meta-)features (see section 5.4). The pre-assumption of separable effects, as in [32], is not necessary for GEDI but can be investigated as constraints between features (see section 5.2).

Deep Learning-based. Recently, Deep Learning enhanced generative models have been explored in the process mining domain. [16] combines data-driven simulation with Deep Learning for process simulation model discovery, to analyze what-if scenarios. In [18], the authors propose ProcessGAN, which involves: (1) input and data pre-processing, (2) the GAN model for Business Process Improvement, and (3) output and data post-processing. However, ample real event logs are needed for the discriminator to distinguish between real and fake samples. In our approach, we explore the feasible feature space without requiring real event logs. Additionally, different than in GEDI, DL approaches like [16, 18] use encoded features, which typically lack interpretability, as explained in [28].

In summary, as opposed to previous works, our approach, GEDI, uses intentional ED-level features to interpretably describe an event log, avoiding reliance on random processes or augmentation techniques. We explore the feasible feature space and steer the data generation process towards specific feature combinations via hyperparameter optimization, eliminating the need for a real event log to train a model. GEDI provides the first link between ED generation and ED characteristic features and can target multiple objectives simultaneously.

3 Preliminaries

3.1 Event Logs

In PM, an *event log* $\mathcal{L}$ is an event data set that consists of multiple events and describes the flow of a process. An *event* e is the smallest unit of an event log. It describes a single execution of a step in a process and has at least three components: *Case identifier* $c(e)$ identifies the process instance an event belongs to. An *activity* $a(e)$ describes the class of an event. Lastly, the *timestamp* $t(e)$ of an event indicates the timing of this occurrence. Aggregatively, a *trace* T is the collection of events with the same case id. We denote the number of traces in an event log as $|\mathcal{L}|$. And a *variant* $\upsilon(T)$ is the sequence of activities in a trace T and $\upsilon(\mathcal{L})$ the collection of variants in $\mathcal{L}$, as in [1]. Event data (ED) is characterized by its granularity, with the basic unit being an event. Aggregations like traces, activities, and variants offer different granularity levels and insights into process behavior. These varying levels complicate the use of traditional data mining algorithms, which require numerical vectors as input. Thus, to apply data mining and machine learning to event data, it needs to be mapped into a numerical space.

3.2 Event Data Features

In many scenarios, features of input data are informative for stakeholders for their decision-making, as through them they can interpret event data behavior. To apply machine learning-enhanced models, it is common practice to translate data into a numerical embedding space. In our work, we extract meta features on event-log level, using FEEED [28], which includes statistical ratios, and complexity measures based on graph entropy. The translation of ED to a vectorial feature space is described as follows:

Definition 1 (Feature extraction). *For an event log $\mathcal{L}$, feature extraction is a function f_e that maps $\mathcal{L}$ to a feature space, i.e., $f_e : \mathcal{L} \to \mathbb{R}^n$ where $\mathbb{R}^n$ is an n-dimensional real vector space.*

For this work, we focus on two types of ED-level features: Ratios, which are known for their simplicity, as well as EPA-based features, which have also been identified to correlate with the quality of discovered process models. [6]

Event-log ratios. Ratio-based features are simple statistical descriptors relating multiple counts of an event log to each other. Let $\upsilon(\mathcal{L})$ represent the variants from an event log $\mathcal{L}$, which are the sequences of activities in each trace (see section 3.1). A common statistic is the ratio of the first k unique variants to the total number of variants. Another feature is the *ratio of variants to the number of traces*. Hence, we define:

$$rmcv = \frac{|\upsilon(\mathcal{L})@1|}{|\mathcal{L}|} \quad (1) \qquad rt10v = \frac{|\upsilon(\mathcal{L})@10\%|}{|\mathcal{L}|} \quad (2) \qquad rvpnot = \frac{|\upsilon(\mathcal{L})|}{|\mathcal{L}|} \quad (3)$$

where $|v(\mathcal{L})@k|$, $|v(\mathcal{L})@k\%|$ denote the number of occurrences of the k top variants or top k percentage of the input data, respectively.

Extended Prefix Automaton (EPA) based features. The authors of [6] offer various measures that exhaustively capture multiple perspectives, based on graph entropy. An extended prefix automaton $EPA = (S_+, T, A, C, seq,\ root)$ is defined by a extended set of states $S_+ = S \cup \{root\}$, a set of activities A and transitions $T \subseteq S_+ \times A \times S$. A partitioning function $C \in S_+ \to \mathbb{N}_0 \cup \{\perp\}$ assigns each state to a partition, and a function $seq \in S \to p(E)$ maps each state to the set of events with the same prefix. The $root \in S_+$ state serves as the starting point with an empty prefix. The *variant entropy* is defined as:

$$E_v = |S| * \log(|S|) - \sum_{i=1}^{\max(C)} |\{s \in S | C(s) = i\}| \cdot \log(|\{s \in S | C(s) = i\}|) \quad (4)$$

Sequence entropy is a measure that captures the frequency of events and their prefixes, based on the count of events within a partition:

$$E_s = |seq(S)| \cdot \log(|seq(S)|) - \sum_{i=1}^{\max(C)} |seq_i(S)| \cdot \log(|seq_i(S)|), \quad (5)$$

where $seq(S) = \bigcup_{s \in S} seq(s)$ and $seq_i(S) = \bigcup_{s \in S | C(s)=i} seq(s)$. Furthermore, the *normalized variant entropy (enve)* and *normalized sequence entropy (ense)* are given by:

$$\bar{E}_v = \frac{E_v}{|S| \cdot \log(|S|)} \quad (6) \qquad \bar{E}_s = \frac{E_s}{|seq(S)| \cdot \log(|seq(S)|)} \quad (7)$$

We can calculate sequence entropy by incorporating the concept of forgetting, where older events contribute less to the entropy than newer ones. We use two methods to determine the weight of an event e [34]:

$$w_{lin}(e) = 1 - \frac{t_{max} - t(e)}{\Delta_t} \quad (8) \qquad w_{exp}(e) = \exp\left(-k \frac{t_{max} - t(e)}{\Delta_t}\right) \quad (9)$$

where $\Delta_t \coloneqq t_{max} - t_{min}$ denotes a sequence's temporal horizon, and k defines a forgetting coefficient $k > 0$. The choice of weighting determines the type of sequence entropy: *normalized sequence entropy linear forgetting (enself)* or *normalized sequence entropy exponential forgetting (enseef)*. For a detailed discussion on EPA-based features, see [6, 34].

3.3 Hyperparameter Optimization (HPO)

Using event log generators, we can guide the generation towards desired descriptors, i.e., the produced event log implies features defined as optimization goals. The hyperparameter optimization (HPO) problem is defined as [21]:

Definition 2. *Let Λ be the set of hyperparameter configurations of algorithm A. Given $\lambda \in \Lambda$, the function $\ell(\cdot, \cdot)$ computes the loss of A_λ when applied to a dataset $\mathcal{D}$. The* ***hyperparameter optimization problem*** *is finding $\lambda^* \in \Lambda$ that minimizes the loss function. Hence, we define:*

$$\lambda^* \in \arg\min_{\lambda \in \Lambda} \frac{1}{k} \sum_{i=1}^{k} \ell(A_\lambda, \mathcal{D}) \tag{10}$$

In our framework, the algorithm A is an event log generation algorithm. Its hyperparameters can be configured in a way to produce an event log with desired characteristics. Finding the optimal configuration becomes the HPO problem. There are several approaches for HPO in the literature such as grid and random searches, evolutionary algorithms, and Bayesian optimization (BO), among others [21]. For the scope of this paper, we chose BO due to its state-of-the-art performance in many current applications [27].

BO employs a probabilistic model to understand the link between hyperparameter configurations and their outcomes, utilizing the exploration-exploitation balance [21]. Exploration seeks areas of uncertainty in the objective function, while exploitation targets areas expected to perform well [10]. BO iteratively selects and evaluates promising hyperparameters, aiming to minimize objective function evaluations for efficiency, especially when the function is expensive to compute. It uses prior evidence to inform the posterior distribution over the function space, characterizing aspects like potential maxima and smoothness [10]. BO often uses Gaussian processes and excels in continuous spaces, suitable for this paper's context.

4 GEDI - Generating ED with Intentional Features

We argue that major characteristics of a given event log are captured by its m (meta-)features $F \coloneqq \{f_1, \ldots, f_m\}$, $\forall f_i \in \mathbb{R}$. In our framework, we aim to explore untapped regions. Therefore, let $F^\square$ be a bounded solution space w.r.t. feature definitions with:

$$F^\square \coloneqq [f_1^l, f_1^u] \times \ldots \times [f_m^l, f_m^u], \quad \forall f_i \in F : f_i \in \mathbb{R}, \tag{11}$$

where f_i^l, f_i^u denotes the respective lower and upper bound of the i-th feature. We use step size $\eta_i \in \mathbb{N}^+$ to scan the search space for the i-th feature, i.e. the sampling rate is given as $(f_i^u - f_i^l)/\eta_i$.

Hence, we start by defining a grid space that is given by $[f_1^l, f_1^u] \times \ldots \times [f_m^l, f_m^u]$ with their respective sampling rates. We introduce our data-generating pipeline $\mathcal{G} : F^\square \to \hat{\mathcal{L}}$ that enables the generation of new event logs $\hat{\mathcal{L}}$ whose features are closely aligned to grid points defined in the bounded space $F^\square$. Our pipeline is illustrated in Figure 2. Intuitively, our generating process $\mathcal{G}(g_k)$ GEDI minimizes the distance between the respective grid points $g_k \in F^\square$ signifying a query feature combination and $f_e(A_\lambda)$, where A_λ denotes an exchangeable module for generating new event logs with hyperparameter configuration λ. In

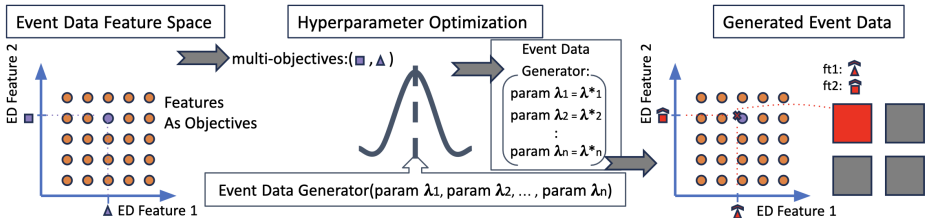


Fig. 2: GEDI generates event data for multiple objectives in a grid search space using Event Data Generator and HPO to configure its parameters.

our framework, we employ the Tree Generator proposed by Jouck et al. [23]. The parameters λ of this generator module are optimized by BO (cf. eq. (10)). GEDI's optimization problem for a given feature combination defined by grid point $g_k \in F^{\square}$ is then given as:

$$\mathcal{G}(g_k) = \min_{\lambda \in \Lambda} d(f_e(A_\lambda), g_k)) \tag{12}$$

The function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ calculates the distance in feature space, where in the following we use $n = 7$ features that we extract from the event logs and which are described in Section 3. As distance metric we apply the Euclidean distance $d(p, q) = \|p - q\|$.

Figure 2 shows a multi-objective workflow for two features, ED Feature 1 and 2. We create a 2D search space, setting step size η_i, and bounds f_i^l and f_i^u to define the grid of possible feature combinations. Using BO for each grid target, HPO adjusts the generator's parameters to match both feature objectives until an optimum is found or a termination condition, like reaching a maximum number of iterations, is met. The final iteration produces a new event log based on the optimized feature values.

5 Evaluation

We aim to answer the following research questions:

- **R1:** *How well does the generator produce ED with specific feature values?*
- **R2:** *How well does generated ED represent the ED space?*
- **R3:** *How well does generated ED align with real ED in a downstream task?*

5.1 Setup

To evaluate, we benchmark representative event logs and compare real and generated data for similarity. Our code is available on GitHub[4].

Hyperparameter Configuration. We generate ED using the Process Tree

[4] https://doi.org/10.5281/zenodo.11416771

Generator [23] with specified hyperparameter intervals: activities_{min, mode, max}= [5, 20], {sequence, choice, parallel, loop, silent, lt_dependency}: [0.01, 1], num_traces: [10, 10001], {duplicate, or}: [0].

Features. For simplicity, we consider features with values between $[0, 1]$, including 4 EPA-based features [6] — *enve*, *ense*, *enself*, and *enseef* — and variant-based indicators — *rmcv*, *rvpnot*, and *rt10v* — as introduced in section 3.

Datasets. Following [6], we utilize publicly available datasets[5], comprising BPIC12, BPIC13cp, BPIC13inc, BPIC15f(1-5), SEPSIS, RTFMP, BPIC20(a-e), without any preprocessing. Additionally, we incorporate BPIC13op, BPIC14di_p, BPIC14dc_p, BPIC14dia_p, BPIC16wm_p, BPIC16c_p, BPIC17, BPIC17ol [17], and BPIC19, representing further BPI Challenges Challenges [5]. ED marked with "_p" have been converted from ".csv" files to ".xes" files. We also include RWABOCSL [12] and HD [30].

Methods. For process discovery (PD) as a downstream task, we use Inductive Miner Infrequent (IMF) [26], Heuristics Miner (HM) [35] and ILP Miner [37].

Evaluation Metrics. We assess PD methods with generated event logs, examining fitness, precision, F-score, size, and CFC, per [6]. Alignment-based *fitness/recall* and *precision* scores, range from 0 to 1 [3, 2]. In BPMN[6] process models, *size* equals node count, *CFC* measures branching from split gateways [29]. Lower *size* and *CFC* indicate better measures.

5.2 R1 - Feasibility

First, we examine the quality of GEDI w.r.t. the distances between a set of features given by real-world ED, and by GEDI-generated event datasets using grid targets (GenGrid). For that, we constructed a grid for each pairwise feature value combination. For each point in the grid, we generated a novel EDS, intending to be as close to a given point as possible. Choosing k features out of seven features available for objectives, we produce $\binom{7}{k} \cdot |p \times q|$ EDS in total, where $7 > k \in \mathbb{N}$; $p, q \in \mathbb{R}$, and p, q denote the resolution in each feature dimension, i.e., the number of grid points being examined for a specific feature. For simplicity and readability we focus our experiments $p, q \in [0, 1]$, with 0.1 steps and $k \in \{1, 2\}$.

Figure 3 depicts the distances between feature values from the generated event logs to the real-world ones. The diagonal shows the range of values, where GEDI produced ED with sensible distances for a single objective. Brighter (darker) colors indicate lower (higher) average distances and, thus, better (worse) results in meeting intended targets. We observed especially for *enve* only values between $[0.1, 0.6]$, which is sensible considering that avoiding OR operators has a direct impact on the relative length of traces, and thus in the upper limit for entropy over the variant per trace distribution. We observe similar restrictions for several features in fig. 3. On the upper diagonal half of fig. 3, we show the average distances w.r.t. each pair of features. The explanation for deviating results

[5] https://www.tf-pm.org/competitions-awards/bpi-challenge

[6] bpmn.org

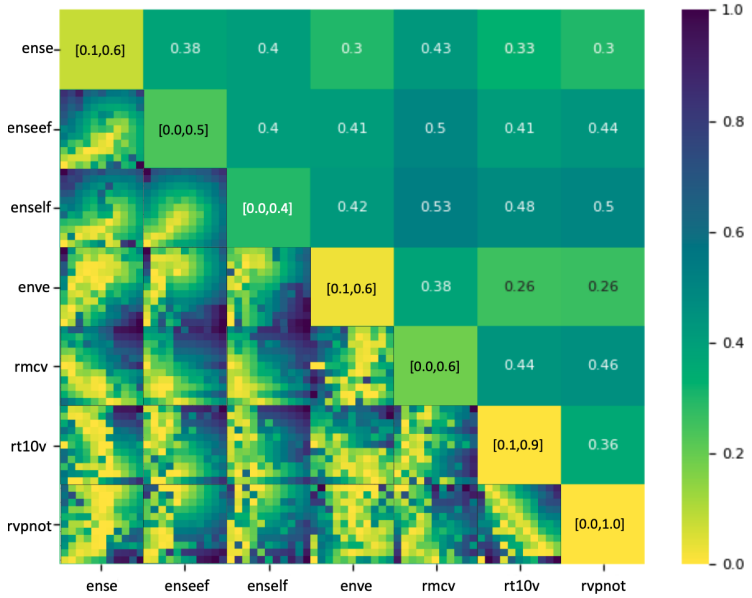


Fig. 3: In the diagonal we show ranges with sensible distances; above the diagonal, we show average distances; and below it, distances between targets and generated event log feature values for each feature combination.

lies, on the one hand, in the restrictions of each feature standalone, as discussed in the single objective experiments above, and in the relation between features.

Highly correlated features such as *rmcv* and *rvpnot* demonstrate, on average, lower feasibility for intentional ED generation. The bottom diagonal half of fig. 3 shows distances from generated ED to the target feature space for various feature value combinations in greater detail. Herein, we fix two feature values on the x- and y-axis. Lighter (darker) colors encode again smaller (larger) distances. The combination of *rmcv* and *enself* exemplarily shows with a bright bottom left corner, how restrictions of both individual features impact the feasibility of an EDS with their combination. Also worth mentioning is the combination of *enseef* and *enself*, which shows a clear positive correlation in its bright diagonal. Regarding **R1**, as demonstrated by the figures, we can define ranges for each feature, where generating intended ED is feasible, also in a multi-objective scenario. The data generator covers larger areas of the search space. Furthermore, we can analyze the search spaces for single and multi-objective generation, and thus learn about relations between features in a k-dimensional space.

5.3 R2 - Representativeness

For further analysis, we select a qualitative sample of GenGrid, filtering for event logs with a distance between targets and generated event log feature values below a threshold $\theta < 0.1$, resulting in 467 event logs. Figure 4 shows the

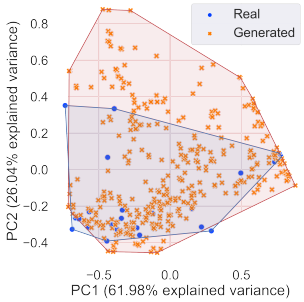


Fig. 4: GenGrid benchmark covers and enriches the feature space.

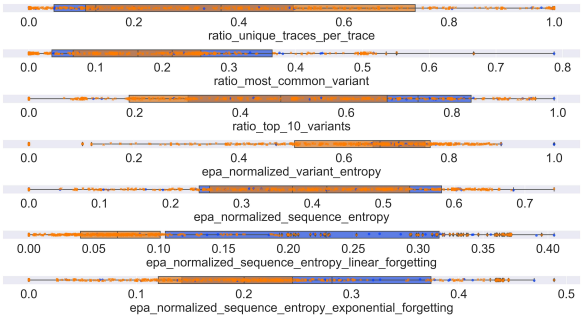


Fig. 5: Features' distribution from 34 real world datasets compared to 467 event logs generated by GEDI.

feature space being covered by real event logs compared to logs generated by GEDI. We conduct a PCA to gain insights into the explained variance. The first and second principal components yield $\approx$ 88% of the variance of the feature space. Furthermore, the illustration shows the convex hull of both, the original ED and the GenGrid. Not only does GenGrid cover the feature space given by the real ED, but also covers a larger search space despite the single and pairwise feature restrictions. Figure 5 shows the distributions of selected features extracted from 26 real ED compared to 467 event logs produced by GEDI. The boxplots show that GEDI is capable of capturing the feature distributions of real competitors resembling real characteristics. This coverage is of uttermost importance to use GenGrid to benchmark methods in various realistic scenarios. Nevertheless, feature distributions of GenGrid spread differently than in the real ED, e.g. for *enself*, which motivates further feature space exploration beyond the current benchmarks' limits. Hence, answering **R2**, GEDI can generate ED containing values w.r.t. given features, which are not provided by existing benchmark datasets.

5.4 R3 - Benchmarking process mining tasks

This section exemplarily offers a downstream task analysis, here Process Discovery (PD), using generated ED. First, fig. 6 shows distribution comparisons of evaluation metrics, obtained running three methods – IMF [26], HM [35] and ILP Miner [37] – on real ED and our GenGrid. Blue boxplots and circles represent real ED results and orange boxplots and crosses represent GenGrid. On the one hand, real ED results show a relatively high fitness for all methods. Adding GenGrid to these results, we observe similar to fig. 5 that GenGrid offers a wider range of metric values than real ED. This validates the merit in the representativeness of GenGrid over real ED for these features. In contrast, complexity feature values are predominantly lower and narrower spread in GenGrid than in real ED. Metrics exposing a better performance for GenGrid than real

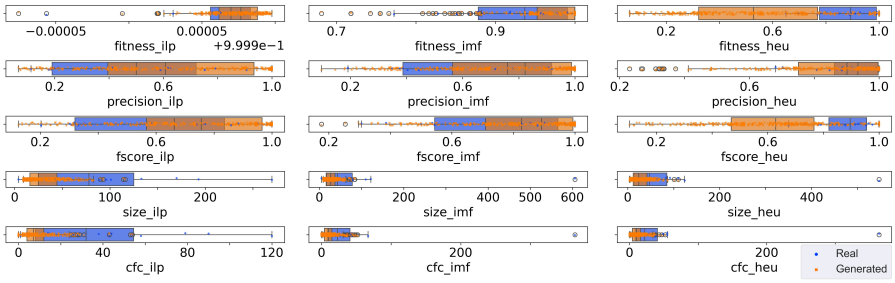
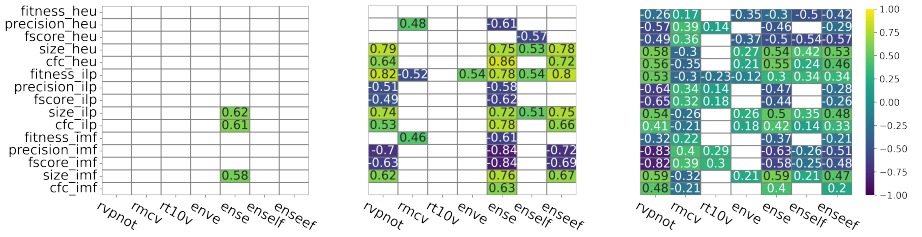


Fig. 6: Quality Metrics for PD on 34 real and 467 GenGrid event logs.

ED can be indicators of the limitations of the chosen configuration space. For our experiments we chose it, considering that higher complexity leads to longer execution time, as observed in [5]. Particularly, the combinations of *precision* or *f-score*, and *HM* show that GenGrid tends to have worse *precision* values and *f-scores* than real ED. One reason for this could be *HM* over-fitting real ED homogeneous feature values. In cases, where most GenGrid measurements have higher performance than most real ED ones, we can recognize predominantly high suitability of the corresponding method for that metric. In any case, thanks to the reproducibility and intended characteristics of GenGrid, we can now challenge the current state-of-the-art to discover new insights regarding a larger, yet unknown space of possible ED.

Next, we explore the relations between features and metrics. As in [6], we utilize correlation analysis on relations between specific features and PD evaluation metrics, to identify how these relations change with more diverse generated ED. For this analysis, besides real ED and previously introduced GenGrid, generated from grid objectives, we use GEDI with real ED feature values as a 7D multi-objective, to create another collection of generated ED (GenRT). We first use the Pearson correlation test, which evaluates the likelihood of a pre-existing linear relationship between two sets of measurements. Figure 7 shows results of each pair of feature and evaluation metric that yield a p-value equal to or below 0.05 in the Pearson correlation test indicating statistical significance. In fig. 7a we observe three significant pairs for real ED. Figure 7b shows correlation values of real ED and GenRT. If real ED were to be perfectly reproduced, the correlation test would deliver the same results for both fig. 7a and fig. 7b. Nonetheless, GEDI produced ED with the same characteristics in fig. 7b, discovering 39 pairs of correlated features and metric combinations, consolidating the pairs encountered in real ED with increased correlation values. The reason GEDI doesn't perfectly reproduce the correlation values of real ED and GenRT might be due to unaccounted degrees of freedom in independent features or restrictions in duplication and configuration space. Nevertheless, our results for *enve* and *ense* combined with metrics employing the *IMF* method, are consistent with findings in [6]. Moreover, diversity in generated ED results in different relations between PD metrics and features than those presented by real ED only, suggesting that



(a) Real ED

(b) GenRT using real targets

(c) GenGrid using grid targets

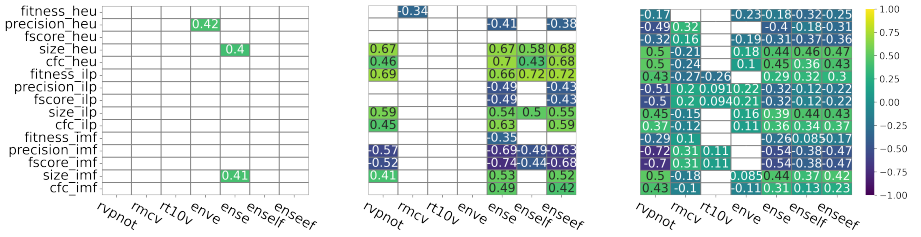
Fig. 7: **Pearson** correlation test for features and PD metrics using p-value<=0.05

real ED exists beyond these included features. Furthermore, fig. 7c shows 84 pairs of correlation measures between features and evaluation metrics. Correlations for real ED are reinforced in fig. 7c since corresponding pairs are found again. Interestingly *size_ilp* and *cfc_ilp* correlation values decrease with the GenGrid. A possible reason for this is that *ense* covers a larger range of values in GenGrid (cf. fig. 5) introducing corresponding metric values.

As a second test, we use the Kendall-tau [25] correlation test to assess the relationship between rank and similarity, regardless of linearity. Similar to the Pearson test, fig. 8 shows correlation values for each pair of features and evaluation metrics with a p-value <= 0.05, indicating statistical significance. In fig. 8a, three significant pairs are observed for real ED. The *ense* with *size* combination using the *IMF* appears in both correlation tests. Figure 8b shows correlation results for real ED and GenRT, revealing 40 correlated feature-metric pairs, with 80% (32 pairs) overlapping with the Pearson test in fig. 7b. The combinations of *envе* and *ense* with metrics using the *IMF* are consistent with findings in [6]. fig. 8c shows 89 correlated pairs, with 96% (87 pairs) overlapping with the Pearson test in fig. 7c. Correlations for real ED are reinforced for two pairs in fig. 8c as they reappear. Notably, the *envе* with *precision* combination using the *HM* is lost between real and generated ED, possibly due to over-fitting, consistent with findings in fig. 6.

6 Discussion and Conclusion

Machine learning models exhibit bias towards characteristics present in the input data, and thus its (meta-)features. We actively address and mitigate these biases through careful data generation to ensure intentional and diverse event outcomes. For that, we propose a novel framework for generating event data with intentional features, called GEDI. Our framework systematically explores a discretized feature space. In the next step, we apply Bayesian Optimization to obtain optimized hyperparameters for any data-generating module (here: process tree generator). The evaluation shows that the generated ED resembles the



(a) Real ED

(b) GenRT using real targets

(c) GenGrid using grid targets

Fig. 8: **Kendalltau** correlation test for features and PD metrics using p-value<=0.05

characteristics of real-world ED and covers a larger area in feature space. Our benchmark analysis shows evaluation results of process discovery models on generated data compared to real-world data. GEDI provides a deep understanding of how feature sets relate to evaluation metrics, allowing the community to create methods tailored for specific downstream tasks. By defining the scope of GEDI benchmarks—including features, metrics, and candidates for a task—we can find approaches that generally perform well or excel in specific areas. Additionally, GEDI can generate event data similar to real-world cases and comply with GDPR.

GEDI for training purposes. GEDI enables to pre-train models on datasets with varying characteristics, the ML engineer can draw on well-trained reserves, which do generalize much better than models being solely trained on data which cover only a specific area in the meta-feature space, see fig. 4. GEDI, thus, represents an extension of data augmentation or simulation of training data. Our evaluation emphasizes the point that process discovery techniques and specifically their metrics are interrelated when training models on existing benchmark datasets and when trained on an enriched data input space (see fig. 7 and fig. 8). Through GEDI the community gains the ability to craft methods specially designed for various downstream tasks.

GEDI for testing purposes. Although GEDI can generate representative ED in terms of features, i.e. it can reproduce desired features from ED (see fig. 3), reproducing benchmark results for PD via selected meta-features remains an open direction. Given a representative set of features for PD and knowledge about such feature values for external valid test data, e.g. from an industry partner, unable to publish their data, GEDI could reproduce the same behavior in PD evaluation metrics. GEDI enables and opens new exciting directions for reproducible benchmarking to mine the potential behind feature and evaluation metrics relations for various downstream tasks.

Limitations arise from the crucial role of feature selection in defining the framework's robustness. This duality serves as both a strength and a weakness. Users benefit from the ability to specify criteria precisely, but an arbitrary feature selection introduces multiple challenges. In scenarios where ED is generated based on all possible combinations of Eq. 11, convergence of the Hyperparameter Optimization (cf. Section 3.3) becomes negatively correlated with the number of optimizing features, as complexity escalates and a larger feature set may introduce combinations leading to unfeasible solutions. Limiting the Hyperparameter Optimization by a maximal number of epochs may interrupt convergence, yielding unfeasible solutions for specific configurations.

In future work, we plan to investigate a broader set of features, explore other parametrizable ED generators, and examine generated event logs on a wider range of benchmarks, as downstream tasks. Additionally, GEDI is extensible to integrating additional information where statistical descriptors for attributive parameters formulate further side constraints on generated event log data. Furthermore, adapting to human-in-the-loop approaches could prove promising for further GEDI applications.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer Berlin Heidelberg (2011)
2. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.: Conformance checking using cost-based fitness analysis. In: 2011 IEEE 15th International Enterprise Distributed Object Computing Conference. pp. 55–64. IEEE (2011)
3. Adriansyah, A., Munoz-Gama, J., Carmona, J., Van Dongen, B.F., Van Der Aalst, W.M.: Measuring precision of modeled behavior. Information Systems and e-Business Management **13**, 37–67 (2015)
4. Andrews, R., Emamjome, F., ter Hofstede, A.H., Reijers, H.A.: An expert lens on data quality in process mining. In: 2020 2nd International Conference on Process Mining (ICPM). pp. 49–56 (2020)
5. Augusto, A., Conforti, R., Dumas, M., La Rosa, M., Maggi, F.M., Marrella, A., Mecella, M., Soo, A.: Automated discovery of process models from event logs: Review and benchmark. IEEE Transactions on Knowledge and Data Engineering **31**(4), 686–705 (2018)
6. Augusto, A., Mendling, J., Vidgof, M., Wurm, B.: The connection between process complexity of event sequences and models discovered by process mining. Information Sciences **598**, 196–215 (2022)
7. Bakir, C., Yuzkat, M., Karabiber, F.: Creating a data generator and implementing algorithms in process analysis. Elektronika ir Elektrotechnika **28**(5), 68–79 (2022)
8. Beerepoot, I., Di Ciccio, C., Reijers, H.A., Rinderle-Ma, S., Bandara, W., Burattin, A., Calvanese, D., Chen, T., Cohen, I., Depaire, B., Di Federico, G., Dumas, M., van Dun, C., Fehrer, T., Fischer, D.A., Gal, A., Indulska, M., Isahagian, V., Klinkmüller, C., Kratsch, W., Leopold, H., Van Looy, A., Lopez, H., Lukumbuzya, S., Mendling, J., Meyers, L., Moder, L., Montali, M., Muthusamy, V., Reichert, M., Rizk, Y., Rosemann, M., Röglinger, M., Sadiq, S., Seiger, R., Slaats, T., Simkus,

M., Someh, I.A., Weber, B., Weber, I., Weske, M., Zerbato, F.: The biggest business process management problems to solve before we die. Computers in Industry **146**, 103837 (2023)
9. Bogdanov, E., Cohen, I., Gal, A.: SKTR: Trace recovery from stochastically known logs. In: 5th International Conference on Process Mining (ICPM). IEEE (2023)
10. Brochu, E., Cora, V.M., de Freitas, N.: A tutorial on Bayesian Optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. CoRR **abs/1012.2599** (2010)
11. vanden Broucke, S.K., De Weerdt, J., Vanthienen, J., Baesens, B.: A comprehensive benchmarking framework (cobefra) for conformance analysis between procedural process models and event logs in prom. In: 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). IEEE (2013)
12. Buijs, J.: Receipt phase of an environmental permit application process (WABO), CoSeLoG project (2022). https://doi.org/10.4121/12709127.V2
13. Burattin, A.: PLG2: multiperspective process randomization with online and offline simulations. In: Azevedo, L., Cabanillas, C. (eds.) Proceedings of the BPM Demo Track 2016 Co-located with the 14th International Conference on Business Process Management (BPM 2016), Rio de Janeiro, Brazil, September 21, 2016. CEUR Workshop Proceedings, vol. 1789, pp. 1–6. CEUR-WS.org (2016)
14. Burattin, A., Re, B., Rossi, L., Tiezzi, F.: A purpose-guided log generation framework. In: Di Ciccio, C., Dijkman, R., del Río Ortega, A., Rinderle-Ma, S. (eds.) Business Process Management. pp. 181–198. Springer International Publishing, Cham (2022)
15. Burattin, A., Sperduti, A.: Plg: A framework for the generation of business process models and their execution logs. In: Business Process Management Workshops: BPM 2010 International Workshops and Education Track, Hoboken, NJ, USA, September 13-15, 2010, Revised Selected Papers 8. pp. 214–219. Springer (2011)
16. Camargo, M., Dumas, M., González-Rojas, O.: Learning accurate business process simulation models from event logs via automated process discovery and deep learning. In: International Conference on Advanced Information Systems Engineering. pp. 55–71. Springer (2022)
17. van Dongen, B.: BPI Challenge 2017 - offer log (2017). https://doi.org/10.4121/UUID:7E326E7E-8B93-4701-8860-71213EDF0FBE
18. van Dun, C., Moder, L., Kratsch, W., Röglinger, M.: Processgan: Supporting the creation of business process improvement ideas through generative machine learning. Decision Support Systems **165**, 113880 (2023)
19. Hildebrant, R., Fahrenkrog-Petersen, S.A., Weidlich, M., Ren, S.: PMDG: Privacy for multi-perspective process mining through data generalization. In: International Conference on Advanced Information Systems Engineering. pp. 506–521. Springer (2023)
20. Hundogan, O., Lu, X., Du, Y., Reijers, H.A.: Created: Generating viable counterfactual sequences for predictive process analytics. In: International Conference on Advanced Information Systems Engineering. pp. 541–557. Springer (2023)
21. Hutter, F., Kotthoff, L., Vanschoren, J. (eds.): Automated Machine Learning - Methods, Systems, Challenges. Springer (2019)
22. Jouck, T., Bolt, A., Depaire, B., de Leoni, M., van der Aalst, W.M.: An integrated framework for process discovery algorithm evaluation. ArXiv **abs/1806.07222** (2018)
23. Jouck, T., Depaire, B.: Generating artificial data for empirical analysis of control-flow discovery algorithms. Business & Information Systems Engineering **61**(6), 695–712 (Dec 2019)

24. Käppel, M., Jablonski, S.: Model-agnostic event log augmentation for predictive process monitoring. In: International Conference on Advanced Information Systems Engineering. pp. 381–397. Springer (2023)
25. Kendall, P.M.: Rank correlation methods. Charles Griffin Book, Oxford University Press, London, England, 5 edn. (Dec 1990)
26. Leemans, S.J., Fahland, D., Van Der Aalst, W.M.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Business Process Management Workshops: BPM 2013 International Workshops, Beijing, China, August 26, 2013, Revised Papers 11. pp. 66–78. Springer (2014)
27. Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., Hutter, F.: Smac3: A versatile bayesian optimization package for hyperparameter optimization. J. Mach. Learn. Res. **23**(1) (jan 2022)
28. Maldonado, A., Tavares, G.M., Oyamada, R.S., Ceravolo, P., Seidl, T.: FEEED: feature extraction from event data. In: van der Werf, J.M.E.M., Cabanillas, C., Leotta, F., Genga, L. (eds.) Doctoral Consortium and Demo Track 2023 at the International Conference on Process Mining 2023 co-located with the 5th International Conference on Process Mining (ICPM 2023), Rome, Italy, October 27, 2023. CEUR Workshop Proceedings, vol. 3648. CEUR-WS.org (2023)
29. Mendling, J.: Metrics for process models: empirical foundations of verification, error prediction, and guidelines for correctness, vol. 6. Springer Science & Business Media (2008)
30. Polato, M.: Dataset belonging to the help desk log of an italian company (2017), https://data.4tu.nl/articles/_/12675977/1
31. Rozinat, A., de Medeiros, A.K.A., Günther, C.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The need for a process mining evaluation framework in research and practice. In: ter Hofstede, A., Benatallah, B., Paik, H.Y. (eds.) Business Process Management Workshops. pp. 84–89. Springer Berlin Heidelberg (2008)
32. Scheibel, B., Rinderle-Ma, S.: Decision mining with time series data based on automatic feature generation. In: International Conference on Advanced Information Systems Engineering. pp. 3–18. Springer (2022)
33. Schreiber, C.: Exploring the impact of process diversity on business process performance. In: ICPM-D 2021: Proceedings of the ICPM Doctoral Consortium and Demo Track 2021 – co-located with 10th International Conference on Process Mining (ICPM 2021); Eindhoven, The Netherlands, November, 2021. CEUR Workshop Proceedings, vol. 3098, pp. 17–18. CEUR-WS.org (2021)
34. Vidgof, M., Wurm, B., Mendling, J.: The impact of process complexity on process performance: A study using event log data. In: Di Francescomarino, C., Burattin, A., Janiesch, C., Sadiq, S. (eds.) Business Process Management. pp. 413–429. Springer Nature Switzerland, Cham (2023)
35. Weijters, A., Aalst, van der, W., Alves De Medeiros, A.: Process mining with the HeuristicsMiner algorithm. BETA publicatie : working papers, Technische Universiteit Eindhoven (2006)
36. Weytjens, H., De Weerdt, J.: Creating unbiased public benchmark datasets with data leakage prevention for predictive process monitoring. In: Marrella, A., Weber, B. (eds.) Business Process Management Workshops. pp. 18–29. Springer International Publishing, Cham (2022)
37. van Zelst, S.J., van Dongen, B.F., van der Aalst, W.M., Verbeek, H.: Discovering workflow nets using integer linear programming. Computing **100**, 529–556 (2018)