Probabilistic Learning of Temporal Uncertainties in Business Processes

Michel Kunkler[0000−0002−1920−7322], Stefanie Rinderle-Ma[0000−0001−5656−6108]⋆

Technical University of Munich, Germany
TUM School of Computation, Information, and Technology
{michel.kunkler,stefanie.rinderle-ma}tum.de

Abstract. Business processes consist of process activities that must be executed to reach a business goal. The processing times of process activities, as well as the waiting times preceding them, are often influenced by inherent uncertainties, resulting in variability in the overall processing duration of the business process. Current data-driven business process simulation approaches utilize historical data of waiting and activity processing times to fit simple single-peaked probability distributions, from which samples are drawn during the simulation. Such probability distributions might be too simplistic and lead to poor simulation results. Probabilistic learning techniques enable the modeling of uncertainties as non-parametric probability distributions, whose shapes dynamically adapt to influencing factors. This work examines the applicability of a recently proposed probabilistic learner, DR-BART, to express uncertainties of activity processing and waiting times. We train multiple DR-BART models using different combinations of input features on different data sets and sample from these models in a business process simulator. We compare the simulation results with those obtained by sampling from parametric probability distributions. Our results show that DR-BART models can be used to improve business process simulation.

Keywords: Probabilistic Learning · Business Process Simulation · Business Process Management · Process Mining

1 Introduction

During their execution, business processes (processes for short) are exposed to uncertainties caused by internal and external reasons such as resource unavailabilities or compliance constraint violations [21]. Often, uncertainties are time-related, e.g., it cannot be predicted (with certainty) when an external stakeholder will deliver a part, or how long a resource will take to conduct a process

⋆ © Springer Nature Switzerland AG 2025. This is the author's accepted manuscript of a paper published in: R. Guizzardi, L. Pufahl, A. Sturm, H. van der Aa (Eds.), *Enterprise, Business-Process and Information Systems Modeling*, Lecture Notes in Business Information Processing (LNBIP), vol. 502, Springer, Cham, 2025. The final authenticated version is available online at: https://doi.org/10.1007/978-3-031-95397-2_12

activity. Quantifying such temporal uncertainties appropriately is key for different tasks of process intelligence, i.e., business process simulation (BPS) [1] and predictive process monitoring (PPM) [22]: In BPS and generative PPM approaches, processing times of activities and waiting times preceding their execution are typically modeled as probability distributions from which samples are drawn to simulate the further course of a process. Modeling the processing times of activities is challenging, especially when human resources conduct them [1]. Data-driven BPS approaches use historical data to set up simulation models. To express the uncertainty inherent to the processing time of an activity, current data-driven BPS approaches take the historical processing times of that activity to fit a simple single-peaked parametric probability distribution [15, 20].

In reality, such simple probabilistic models might not fit the underlying data well. Consider a process as depicted in Figure 1 with three activities. The first activity *waits for all parts to arrive*, which are delivered by a parcel delivery service that arrives every morning at around 9 a.m. The resulting activity completion times can be described by a multi-peaked probability distribution, as shown below the activity. Assume further that the *quality control* activity can be executed faster with every time it is executed again, e.g. because less checks have to be done in consecutive executions. Then, a fitting probability distribution would not only depend on the name of the activity but also on its previous number of executions in the running process (see distribution below the activity).

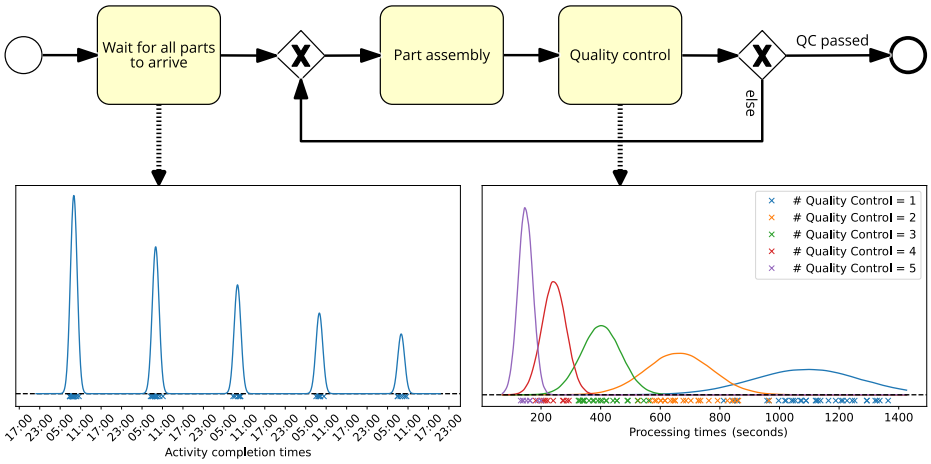

Fig. 1. Historical samples of the processing times of two activities and probability density functions

In this work, we address the learning of probability distributions for processing and waiting times of process activities on historical process data. Learning probability distributions has been addressed in statistics and machine learning

as distribution(al) regression [12, 13], or probabilistic learning [12]. Klein [12] distinguishes the term probabilistic learning from distributional regression by its ability to learn higher-order dependencies inherent in the data by employing machine learning techniques.

A recently proposed probabilistic learning model is *Density Regression - Bayesian Additive Regression Trees* (DR-BART) [19]. DR-BART is a non-parametric tree-based ensemble model. For a given input, DR-BART yields a Gaussian Mixture Model (GMM), which can approximate any smooth probability density function to a desired degree. Furthermore, due to its tree-based structure, DR-BART can "capture complex, nonlinear relationships and interactions" [19] in the input data that may influence the distributions of processing or waiting times. Because DR-BART can learn multi-peaked probability distributions whose shapes can depend on context data, DR-BART has the potential to overcome the aforementioned limitations of current approaches.

However, training probabilistic DR-BART models on process data can be challenging in practice due to the following reasons: First, DR-BART requires fixed-sized input data. Traces in process event logs reflecting the execution of different process cases can be of varying length due to, e.g., alternative branchings or loop structures in the underlying process models. Hence, it is unclear how process traces should be encoded to function as input to DR-BART models. Second, because real-world event logs can consist of large numbers, e.g. millions, of events, it is unclear whether this results in intractable training times for DR-BART models. Third, because DR-BART is a non-parametric Bayesian method, i.e., it can adapt the numbers of parameters during training, it can be prone to overfitting. DR-BART utilizes regularization hyperparameters to mitigate overfitting, but the impact of these parameters has been subject to limited empirical investigation [12].

In this work, we examine the applicability of DR-BART models for sampling processing and waiting times in BPS models. We propose to apply feature encoding techniques to encode event log data to a fixed-sized input size. We then use different combinations of features from the encoded data to train multiple DR-BART models on three different event logs and apply the trained DR-BART models for sampling processing and waiting times in processes. We evaluate the application of our trained DR-BART models in a BPS model by comparing our DR-BART models with currently used (parametric models) for sampling processing times and waiting times. Our results show that DR-BART can improve the precision of a BPS model when appropriate features are encoded to DR-BART.

This work is structured as follows: In Section 2, we present related work and fundamentals. We present our process data encoding approach for the application of DR-BART in Section 3 and describe our evaluation method for examining the applicability of DR-BART in BPS in Section 4. Afterwards, we present our results in Section 5 and discuss the results and conclude our work in Section 6.

2 Related Work

This section discusses existing work on uncertainties, probabilistic learning, and BPS and introduces fundamentals required for the proposed approach.

Uncertainties: In machine learning, a distinction has been made between two types of uncertainties, i.e., aleatoric and epistemic uncertainties [11]. Aleatoric uncertainties are considered irreducible as the uncertainties stem from inherently random effects. In contrast, epistemic uncertainties are referred to as "uncertainty due to a lack of knowledge about the perfect predictor" [11] and hence are considered reducible uncertainties. Epistemic uncertainties can be further divided into approximation and model uncertainties. Approximation uncertainties refer to uncertainties due to a lack of data for selecting appropriate parameters for a predictor model. In general, approximation uncertainties can be reduced by obtaining more training samples. Model uncertainties refer to uncertainties due to a model's insufficient approximation capabilities. Models with high capacity allow more flexibility which can lead to disappearing model uncertainties [11]. However, approximation uncertainty can be challenging when training models with a high capacity. As models with little capacity make stronger model assumptions, i.e., stronger assumptions about the underlying data, they can require less data to fit the model. Different representations for aleatoric uncertainties exist, where probability distributions are the most general and complex representation [7].

Probabilistic Learning aims at learning aleatoric uncertainties by leveraging machine learning techniques to capture complex interactions of the input data [12]. The goal of training probabilistic models is usually to minimize a specific loss function, which is often based on proper scoring rules [11].

DR-BART is a recently proposed non-parametric tree-based ensemble learning method for training probabilistic models [19]. It yields a continuous probability distribution for a given input data. The returned continuous probability distribution of DR-BART is a GMM, which can approximate any smooth probability density function to a desired degree.

A DR-BART model combines two tree-based ensemble models: The leaves in the first tree-based ensemble model represent mean values of Normal distributions, while the leaves of the other tree-based model represent variances. DR-BART leverages a latent variable such that multiple pairs of means and variances can be returned for a given input data, which then constitute the normally distributed components in the returned GMM. In each of the two tree-based ensembles, DR-BART uses a predefined number of trees: In the implementation from Orlandi et al. [19], the default number of mean trees is set to $m_{mean} = 200$, and the number of variance trees to $m_{var} = 100$.[1]

For training a DR-BART model, the likelihood of the training samples is maximized via Gibbs sampling. At each Gibbs step, one of four possible modifications (a grow, prune, change, or swap modification) to a tree in the ensemble tree models is proposed and tested. Since maximizing the likelihood alone would quickly result in overfitting and degenerate Gaussian components where,

[1] `https://github.com/vittorioorlandi/drbart/`

e.g., each training sample has its distinct leaf with a matching mean value and zero variance, DR-BART regularizes the tree structure and requires a minimum amount of observations in every leaf node. In the implementation of Orlandi et al. [19], at least 5 observations are required in every leaf node. The tree structure is regularized via α and β parameters (see [5]). Orlandi et al. [19] use in default $\alpha = 0.95$ and $\beta = 2$, which rewards a "bushy" tree shape [5].

DR-BART itself is an extension to BART [6], which is a tree-based ensemble learning model for mean regression tasks. For BART, it is acknowledged that due to its multi-tree structure, it is robust against converging to local minima during training [6]. Therefore, running the Gibbs sampler once for sufficiently many iterations seems sufficient for training a DR-BART model.

Business Process Simulation (BPS) is considered one of the "most established analysis techniques" [2] in Business Process Management. As setting up simulation models by hand can be cumbersome, data-driven BPS approaches leverage historical process data to learn a BPS model using process mining techniques. In their seminal work on data-driven BPS, Rozinat et al. [20] consider several process perspectives separately, i.e., they discover the control-flow, decision points, roles, and processing and waiting times and integrate them into a single BPS model. In their work, they exclusively fit Normal distributions to the processing and waiting times of each activity. However, they note that it might be meaningful to train different distributions [20].

Martin et al. [16] reviewed data-driven BPS approaches: They notice that processing times in BPS models are either sampled from a parametric probability distribution or from mathematical expressions, i.e., formulas that yield a deterministic value. They suggest combining these approaches such that processing times of some components are calculated based on mathematical expressions and drawn from probability distributions for other components. Some recent data-driven BPS approaches have built upon this concept: For example, Meneghello et al. [17] propose a BPS approach in which processing and waiting times are derived from either probability distributions or obtained from mathematical expressions that take multiple input variables into account. In the simulator used in [14], mean regression is first used to predict the expected processing times of an activity. Then, a normally distributed error term is added to the prediction to account for variability.

In a recent work, López-Pintado et al. [15] build on the assumption that an activity's processing time is affected by the resource performing it. They fit multiple (single-peaked) probability distributions for each resource-activity combination and select the best-fitting one. They refer to this approach as *resource differentiation*. This differentiation approach could be adapted to other factors, e.g., case attributes or context data, but combining multiple factors would pose a challenge: Due to the curse of dimensionality, the number of observations would rapidly decrease. Furthermore, differentiation does not work directly for continuous attributes, necessitating a binning strategy and appropriately chosen bin sizes.

To the best of our knowledge, probabilistic learning has not been used to learn processing or waiting times in BPS.

3 Data Encoding

In this section, we present our approach for aggregating sequence-based data into a fixed-sized input that can be used to train DR-BART models.

3.1 Event Log Data

Table 1. Example event log

case	timestamp	label	resource
1	2024-10-31 07:00	A	Bob
1	2024-10-31 07:15	B	Alice
1	2024-10-31 08:30	B	Felix
2	2024-10-31 09:00	A	Alice
2	2024-10-31 09:15	B	Felix
1	2024-10-31 09:45	C	Bob

Historical process data is often represented in event logs [3]. In this work, we assume that an event has at least three attributes, i.e., a case identifier which links an event to a process case, a timestamp attribute which expresses the time at which an event happened, and an event label which links the event to a class of event types, such as to the start or completion of a distinct process activity. An exemplary event log can be seen in Table 1. Take the event in the first row: It occurred when executing case 1, refers to an activity with label *A*, and was processed by resource *Bob*.

3.2 Feature Engineering and Prefix Encoding:

The event log data needs to be transformed for training and inferring probability distributions from DR-BART models. In particular, we derive the target data, the processing and waiting times from the event log, and apply feature engineering techniques to obtain additional features. Additionally, we apply prefix encoding techniques to encode the history of a case into the feature data.

Deriving processing and waiting times: Many event logs only record the completion of activities. When only completion timestamps (or conversely, only the start timestamps) are available, it can become challenging to obtain the actual processing time of an activity and its preceding waiting time. Intuitively, the timestamp of an event and the timestamp of its preceding event from the same case can be taken to obtain a duration. Taking this duration as processing time

comes with three problems: First, this approach assumes that the activities are executed in a sequential order. If activities are actually performed in parallel, the calculated duration may underestimate the actual processing time. Second, it is not possible to determine the duration of the first activity. Third, the durations may include both waiting times and actual processing times. Henceforth, special care must be taken when working with such event logs: For example, activities that run in parallel must be identified, and the duration must be taken between the activities' actual preceding activities. Other works have addressed decomposing the duration between the completion of two activities into a waiting and processing time [23].

Other event logs store each activity's internal state. Oftentimes, event logs are stored in the eXtensible Event Stream (XES) standard and use the XES lifecycle extension. The XES lifecycle extension itself implements the Business Process Analytics Format (BPAF) state model [18]. In the BPAF state model, it is, e.g., logged when an activity is ready for execution and when the execution has started and ended. When event logs use the XES lifecycle extension, we derive processing or waiting times, respectively, by calculating the time between the lifecycle transitions of an activity.

Prefix encoding: To encode the history of a running case, but reduce the sequence-based event log data to a fixed-sized input, required for DR-BART, we use prefix encoding techniques. Verenich et al. [22] identify two prefix encoding techniques that are applicable to DR-BART: Last m-states encoding and aggregation encoding. In the last m-states encoding, the m variable specifies the number of previous events of a case that are encoded. However, [22] note that the majority of publications choose $m = 1$, i.e., do only encode the most recent event and no previous events. We also select $m = 1$ in this work, as choosing a larger m would strongly increase the input size. We provide information on previous events instead by using aggregation encoding.

Aggregation encoding adds additional attributes to the event log that aggregate information about the case's previous events. For example, information about a numerical attribute can be aggregated by adding a new attribute that represents, e.g., the sum, average, minimum, or maximum value of the previous values. For categorical attributes, for each value, an additional column can be created with the number of occurrences of the categorical attribute value. In this work, we examine count aggregations for activity and resource attributes. In Table 2, the columns `A`, `B`, `C` represent count aggregations on the activity label attribute, and the columns `Bob`, `Alice`, `Felix` count aggregations on the resource attribute.

Feature engineering Additionally, we apply feature engineering, i.e., obtaining new feature attributes from other features in the event log. In particular, we conduct feature engineering based on the timestamp attributes. As performances of human resources have been shown to differ over time [1], or as waiting times

might also depend on the time, we added the `day of the week` and the `seconds in the day` attributes.

Table 2 shows the encoded event log from the original event log in Table 1.

Table 2. Encoded event log

case	timestamp_start	timestamp_end	label	res.	A	B	C	Bob	Alice	Felix	seconds in the day	day of week	dur.
1	2024-10-31 07:00	2024-10-31 07:15	B	Alice	1	1	0	1	1	0	25200	4	900
1	2024-10-31 07:15	2024-10-31 08:30	B	Felix	1	2	0	1	1	1	26100	4	4500
1	2024-10-31 08:30	2024-10-31 09:45	C	Bob	1	2	1	2	1	1	30600	4	4500
2	2024-10-31 09:00	2024-10-31 09:15	B	Felix	1	1	0	0	1	1	32400	4	900

4 Evaluation Method

In this section, we describe our method to evaluate the applicability of DR-BART models in BPS. First, we describe the event logs that we used for training DR-BART models and how we applied the DR-BART models for BPS. Second, we describe the metrics we used to evaluate the simulation results. We implemented our approach and the evaluation in Python, which is publicly available.[2]

4.1 Evaluation Datasets

To evaluate the applicability of DR-BART models for expressing waiting and processing times, we train DR-BART models on one artificial and two real-life data sets. Properties of the three data sets are depicted in Table 3.

The artificial data (AR)[3] set describes a sequential process with three activities, resembling a repair shop in the manufacturing domain. It has five different resources that have different properties: Some resources are faster at conducting tasks in the morning; some resources occasionally take breaks during the processing of a task that is not logged, but this increases the processing times; two resources are not able to work well with each other. If the other resource has conducted a previous activity, the resource is likely to take longer on a subsequent task. Because the data set is artificial, we can compare DR-BART to an optimal probabilistic model.

The second data set (PCR)[4] is a real-world data set of a coronavirus testing laboratory that conducts Polymerase Chain Reaction (PCR) tests. This process has been under the active control of a workflow engine, and an explicit process model exists. The resources that have conducted activities have not been tracked for this data set.

[2] `https://github.com/ltsstar/TaskExecutionTimeMining/`

[3] AR: `https://github.com/ltsstar/TaskExecutionTimeMining/blob/main/data/artificial_event_log_2.xes`

[4] PCR: `https://doi.org/10.5281/zenodo.11617408`

The third data set (BPIC-2017)[5] is a real-world data set from the financial domain. It is a loan application process from a Dutch bank and has been widely investigated in the Business Process Intelligence Competition (BPIC) 2017. It consists of more events, cases, and resources than the other two data sets.

Table 3. Dataset Properties

	Cases	Events	Variants	Event labels	Resources	Case Length Mean (Std.Dev.)	Case Duration Mean (Std.Dev.)
Artificial	1802	16 209	1	3	5	9.00 (0.00)	12.18 (5.49) hours
PCR	6166	117 703	1213	8	-	19.09 (3.37)	5.52 (7.74) hours
BPIC-17	31 509	1 202 267	15 930	26	159	38.16 (16.72)	21.9 (13.17) days

To examine whether DR-BART models will overfit the event log data, we conduct a train/test split based on the process case identifiers, such that 80% of the cases were assigned to the training data set and 20% to the testing data set. The evaluation was then conducted on both the test and the training data sets.

4.2 Training Probabilistic Models

The encoded training data sets are used to train DR-BART models. We train models with different combinations of attributes and two different numbers of iterations. For each data set, we select the number of training iterations such that on recent hardware, the models with few iterations could be trained within a few hours, and the models with a larger number of iterations within a few days. The number of iterations can be seen in Table 4. On the BPIC-2017 data set, some DR-BART models could not be trained: Four models ran into an error during the training due to numerical instabilities, and for the larger model, two attribute combinations were shown to be computationally infeasible, i.e., they could not be trained within a week. The long training durations for the two models are possibly due to a growing number of Gaussian components in the DR-BART models, which leads to overly long evaluation times on the training samples.

For comparison, we train resource-differentiated probabilistic models as proposed in [15], using their publicly available implementation.[6] Additionally, we use the same code to train probabilistic models for each activity, i.e., without the resource differentiation. For the AR data set, we know the underlying probabilistic model and hence also evaluated on that model for comparison.

4.3 Business Process Simulator

We apply our trained probabilistic models in a BPS model. In particular, we sample from the probabilistic models to simulate cycle times of process cases,

[5] BPIC-17: `https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b`

[6] `https://github.com/AutomatedProcessImprovement/pix-framework`

i.e., the time from the beginning to the end of process cases. As we focus on uncertainties of processing and waiting times in this work (and do not address control flow or resource uncertainties), our simulator replays the events of a process case and samples the processing or waiting time for each event from the used probabilistic model. Similarly, when an event has a resource label, that resource is passed to the probabilistic model.

In the AR and the BPIC-2017 data set, the simulator obtains a cycle time by summing up the processing and waiting time samples of the replayed events. Since an explicit process model exists for the PCR data set, where some process activities are executed in parallel, we leverage this information to aggregate processing times: E.g., when two process activities are in parallel, processing time samples are drawn from both activities and proceed with the sample with the higher value.

4.4 Monte Carlo Sampling

Deriving a probability distribution of cycle times analytically can quickly become computationally intractable. Therefore, we use Monte Carlo (MC) sampling to approximate a probability distribution of cycle times.

Because sampling sufficiently many MC trials is crucial for MC sampling, we chose to draw 10 000 MC samples for evaluating the PCR and AR data sets. On the BPIC-2017 data set, this number has proven to be computationally too expensive: Because we replay every event in a data set, choosing 10 000 MC trials on the BPIC-17 data set means sampling $1\,202\,267 \times 10\,000 = 12\,022\,670\,000$ times from each of the tested probabilistic models. For each of these samples, the individual trees of the DR-BART model have to be traversed to eventually draw a value. Therefore, we reduced the number of MC trials to 1000 for this data set.

4.5 Evaluation Metrics

We evaluate how the sampled process cases' cycle times align with the actual cycle times by using two common proper scoring rules, which "assess the quality of probabilistic forecasts, by assigning a numerical score based on the predictive distribution and on the event or value that materializes" [9]. Traditional BPS evaluation metrics compare only a single sampled outcome of a process case with its true outcome [4]. As we sample multiple scenarios for a single process case, we apply different metrics.

The first metric we use is the average log-likelihood, where higher average log-likelihood values are desirable. We obtain the average log-likelihood by averaging the log sum of the probabilities of the true cycle times on the sampled cycle times. To obtain the probability of the true cycle time x_a on the sampled cycle times $X = (x_1, ..., x_n)$, we conduct kernel density estimation on X. We use Gaussian kernels and Silverman's rule to estimate the bandwidth parameter h.

$$\hat{f}(X, x_a) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(x_i - x_a, h) \tag{1}$$

The second metric we used is the average continuous ranked probability score (CRPS), where a lower CRPS is desirable. Unlike the average log-likelihood metric, the CRPS is sensitive to the distance of the predicted case cycle times to the true cycle time. We adapt the notation of [10] and define the CRPS as a distance between the empirical cumulative density function $F_X(x)$ of our sampled cycle times, where $X = (x_1, ..., x_n)$ denotes our samples, and F_{x_a} as the shifted Heaviside function, shifted by the true cycle time x_a.

$$\begin{aligned} CRPS(X, x_a) &:= \int_{-\infty}^{\infty} [F_X(u) - F_{x_a}(u)]^2 du \\ F_X(x) &:= \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(X_i \leq x) \\ F_{x_a}(x) &:= \begin{cases} 0 & \text{if } x_a > x \\ 1 & \text{if } x_a \leq x \end{cases} \end{aligned} \tag{2}$$

5 Results

The evaluation results on the two metrics across the data sets and models can be seen in Table 4. The results show that the performance of simulation models that sample activity processing and waiting times from i) our herein presented DR-BART approach with different metrics; ii) the resource differentiation approach from [15]; and iii) an approach were probability distributions were fit only to activity names (without differentiating for resources) by using the PIX framework.

Not surprisingly, the performance of the DR-BART approach depends on the selected attributes: Selecting only a few attributes, e.g., only the activity label, leads to poor results for all data sets. However, selecting too many attributes also leads to degrading performances across all data sets.

AR On the AR data set, DR-BART could achieve the best results when the right attributes were selected. While it could leverage the *seconds in day* and *resource count* information to yield better results than the *differentiated resources* approach, its performance decreases when trained for more iterations, indicating an overfitting of the models.

PCR On the PCR data set, which does not come with resources, DR-BART could clearly outperform the approaches where only a single-peaked probability distribution was fit for each activity. Training DR-BART with the *activity count* attribute leads to degrading performances, while the *seconds in day* attribute seems to have an impact on the performances. As in the AR data set, training with more iterations decreases the performance for most DR-BART models.

BPIC 2017 The BPIC 2017 data set is the only one in which the *differentiated resources* approach consistently outperforms DR-BART. However, the difference between the *differentiated resources* and the simple activity-based probability

Table 4. Results DRB = DR-BART, PIX = PIX-Framework from [15], Opt. = Actual probabilistic model, a = activity, r = Resource, ac = activity count, rc = resource count, s = seconds in day, d = day of week, Training: (burn in iterations / kept iterations / thinning interval, Samples = Number of samples for a process case, underscored scores: best DR-BART scores, bold scores: overall best scores, - = training error due to numerical instabilities, * = training computationally infeasible

AR		Training: 100,000/100/100 Samples: 10,000				Training: 1,000,000/100/100 Samples: 10,000			
		Train		Test		Train		Test	
	a r ac rc s d	LL	CRPS	LL	CRPS	LL	CRPS	LL	CRPS
DRB	x - - - - -	-1.83	1.02E+04	0.15	9.96E+03	-0.03	1.45E+04	-0.11	1.45E+04
DRB	- x - - - -	0.06	1.43E+04	0.05	1.37E+04	-0.15	1.79E+04	-0.14	1.78E+04
DRB	x x - - - -	0.07	1.04E+04	0.30	1.00E+04	-0.00	1.30E+04	0.04	1.30E+04
DRB	x x - - - x -	0.16	1.01E+04	0.33	9.65E+03	0.21	1.21E+04	0.21	1.21E+04
DRB	x x x - - x -	0.09	1.34E+04	0.14	1.26E+04	-1.58	1.01E+04	-1.83	1.01E+04
DRB	x x - x - x -	0.26	**8.25E+03**	**0.40**	**8.05E+03**	0.33	1.04E+04	**0.32**	1.04E+04
DRB	x x x x - x -	**0.36**	9.69E+03	0.37	9.35E+03	0.29	1.14E+04	0.29	1.13E+04
DRB	x x - - x x	0.20	1.21E+04	0.23	1.15E+04	-1.21	1.03E+04	-1.28	1.03E+04
DRB	x x x x x x	0.32	9.58E+03	0.39	9.21E+03	0.28	1.16E+04	0.27	1.16E+04
PIX	x - - - - -	0.28	9.88E+03	-1.75	1.01E+04	0.28	9.88E+03	-1.75	1.01E+04
PIX	x x - - - -	0.34	9.62E+03	-0.41	9.80E+03	**0.34**	**9.62E+03**	-0.41	**9.80E+03**
Opt.	x x - x x -	***0.50***	***7.97E+03***	***0.54***	***7.97E+03***	***0.50***	***7.97E+03***	***0.54***	***7.97E+03***

PCR		Training: 10,000/100/100 Samples: 1000				Iter.: 100,000/100/100 Samples: 1000			
		Train		Test		Train		Test	
	a r ac rc s d	LL	CRPS	LL	CRPS	LL	CRPS	LL	CRPS
DRB	x - - - - -	0.66	1.25E+04	0.69	1.18E+04	0.13	1.55E+04	0.17	1.52E+04
DRB	x - - - x -	**0.80**	1.15E+04	0.70	1.10E+04	**0.61**	1.25E+04	**0.63**	1.20E+04
DRB	x - x - x -	0.22	1.57E+04	0.32	1.53E+04	0.54	1.42E+04	0.51	1.37E+04
DRB	x - - - x x	-3.96	**9.67E+03**	**1.20**	**8.83E+03**	-2.47	**1.03E+04**	-0.36	**9.79E+03**
PIX	x - - - - -	-8.62	1.15E+04	0.14	1.08E+04	-8.62	1.15E+04	0.14	1.08E+04

BPIC 2017		Training: 750/5/5 Samples: 1000				Training: 7500/50/50 Samples: 1000			
		Train		Test		Train		Test	
	a r ac rc s d	LL	CRPS	LL	CRPS	LL	CRPS	LL	CRPS
DRB	x - - - - -	-70.71	2.46E+64	-70.67	2.41E+64	*	*	*	*
DRB	- x - - - -	-	-	-	-	*	*	*	*
DRB	x x - - - -	-	-	-	-	-	-	-	-
DRB	x x - - x -	-20.44	5.98E+21	-20.44	5.81E+21	-	-	-	-
DRB	x x x - x -	-9.70	4.23E+13	-9.74	8.98E+13	-7.58	1.29E+13	-7.60	1.17E+11
DRB	x x - x x -	-5.43	4.33E+06	-5.45	4.32E+06	-7.07	2.54E+20	-7.06	1.45E+13
DRB	x x x x x -	n.a.	3.00E+17	n.a.	6.14E+16	-11.17	3.49E+12	-11.18	3.32E+12
DRB	x x - - x x	-16.11	2.09E+19	-16.17	1.89E+19	-4.29	8.56E+05	**-4.26**	8.55E+05
DRB	x x x x x x	-9.77	2.85E+17	-9.86	3.80E+16	-10.94	2.18E+12	-11.00	1.11E+12
PIX	x - - - - -	-4.30	**8.33E+05**	**-4.45**	**8.31E+05**	-4.30	**8.33E+05**	-4.45	**8.31E+05**
PIX	x x - - - -	**-4.22**	8.34E+05	-4.58	8.33E+05	**-4.22**	8.34E+05	-4.58	8.33E+05

distribution approach is only marginal, indicating that the processing and waiting times in this data set depend only little on the tested attributes. Moreover, when trained for only 750 iterations, most DR-BART models show distinctively worse results than when trained for 7500 iterations. While training for many more iterations proved computationally intractable with the current DR-BART implementation, it remains unclear whether extending training to more iterations would have yielded better results.

Overall, the results show that DR-BART is able to outperform the other approaches in two of the three tested data sets when meaningful feature attributes are selected. The longer-trained models on the AR and PCR data sets show a decreasing performance. This could be due to overfitting of the DR-BART models to the individual processing and waiting times.

6 Discussion & Conclusion

In this work, we examined the applicability of a probabilistic learner, DR-BART, for learning probabilistic models that represent activity processing times and waiting times in business processes. We used feature encoding and engineering techniques to encode sequential data into fixed-sized input data required by DR-BART. We then compared the performance of sampling processing and waiting times using DR-BART models with sampling from traditional probabilistic models. Our results show that DR-BART models can contribute to better BPS models than the currently used probabilistic models.

DR-BART models were able to outperform the performance of existing approaches in two of three data sets, when meaningful feature attributes were selected. The selection of irrelevant features for DR-BART has been shown to degrade the model's predictive performance. Similarly, when only a few features were selected for training DR-BART models, the models' performance decreased in most cases when training for many iterations (e.g., for one million iterations on the AR data set). This indicates that our trained DR-BART models tend to overfit. Selecting only a few feature attributes has been shown to cause problems training DR-BART models on the BPIC 2017 data set. This issue is likely due to the model performing excessive splits on its latent variable, resulting in probability distributions that consist of many Gaussian components for each data sample. These complex distributions might then increase the computational cost for calculating the likelihood of the training samples and, at the same time, decrease the model's predictive performance due to overfitting.

Future work should address the computational issues and overfitting problems of DR-BART, which might be achievable by different means: First, the training process of DR-BART itself could be enhanced by applying hyperparameter search techniques, or implementing early stopping or restarting techniques. Second, it might be meaningful to add additional regularization to DR-BART models. For example, the number of splits on the latent variable could be regularized to avoid overly complex and overfitting models. Third, instead of training a single DR-BART model for processing and waiting times of all known activ-

ities, it might be meaningful to apply bucketing techniques [22] and, e.g., train individual DR-BART models for each activity, or one DR-BART model for waiting times and one for processing times. Fourth, the DR-BART implementation could be heavily parallelized. Currently, the DR-BART implementation uses only a single CPU core. Calculating the likelihood of the individual training samples, on which most of the time during training is spent, could be parallelized, such that, e.g., multiple CPU cores are utilized.

A future use case for DR-BART could involve testing for undesired influences of different process attributes and contextual data on processing and waiting times. For example, an organization might be interested in whether daily working hours or the time of day affect the processing times of activities. Our approach could help answer this question and assist the organization in mitigating undesired influences, such as by limiting working hours.

While we have focused on training DR-BART models in this work, future work should test the applicability of other probabilistic models. Neural network-based models, such as Bayesian Neural Networks (BNNs), or BNN approximation techniques, e.g., Monte Carlo (MC) dropout as Bayesian approximation [8], appear promising because, on the one hand, they can learn complex non-linear relationships and, on the other hand, they can approximate any probability distribution, when the neural network has sufficient capacity.

The presented results are limited to the use of a simplified business process simulator. Our simulator simulated the individual cases independently from each other, ignoring that the performances can depend on other running cases, e.g., because resources work on multiple cases simultaneously [16].

In this work, we examined the applicability of probabilistic learning for processing and waiting times in business processes using DR-BART. Our results show that DR-BART models can contribute to better BPS models than the currently used probabilistic models when meaningful feature attributes are selected.

References

1. van der Aalst, W.M.P.: Business process simulation revisited. In: Barjis, J. (ed.) Enterprise and Organizational Modeling and Simulation - 6th International Workshop, EOMAS 2010, held at CAiSE 2010, Hammamet, Tunisia, June 7-8, 2010. Selected Papers, Lecture Notes in Business Information Processing, vol. 63, pp. 1–14 (2010), https://doi.org/10.1007/978-3-642-15723-3_1
2. van der Aalst, W.M.P.: Business process simulation survival guide. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 1: Introduction, Methods, and Information Systems, pp. 337–370, Springer Berlin Heidelberg, Berlin, Heidelberg (2015), ISBN 978-3-642-45100-3, https://doi.org/10.1007/978-3-642-45100-3_15, URL https://doi.org/10.1007/978-3-642-45100-3_15
3. van der Aalst, W.M.P., Adriansyah, A., de Medeiros, A.K.A., Arcieri, F., Baier, T., Blickle, T., Bose, R.P.J.C., van den Brand, P., Brandtjen, R., Buijs, J.C.A.M., Burattin, A., Carmona, J., Castellanos, M., Claes, J., Cook, J.E., Costantini,

N., Curbera, F., Damiani, E., de Leoni, M., Delias, P., van Dongen, B.F., Dumas, M., Dustdar, S., Fahland, D., Ferreira, D.R., Gaaloul, W., van Geffen, F., Goel, S., Günther, C.W., Guzzo, A., Harmon, P., ter Hofstede, A.H.M., Hoogland, J., Ingvaldsen, J.E., Kato, K., Kuhn, R., Kumar, A., Rosa, M.L., Maggi, F.M., Malerba, D., Mans, R.S., Manuel, A., McCreesh, M., Mello, P., Mendling, J., Montali, M., Nezhad, H.R.M., zur Muehlen, M., Munoz-Gama, J., Pontieri, L., Ribeiro, J., Rozinat, A., Pérez, H.S., Pérez, R.S., Sepúlveda, M., Sinur, J., Soffer, P., Song, M., Sperduti, A., Stilo, G., Stoel, C., Swenson, K.D., Talamo, M., Tan, W., Turner, C., Vanthienen, J., Varvaressos, G., Verbeek, E., Verdonk, M., Vigo, R., Wang, J., Weber, B., Weidlich, M., Weijters, T., Wen, L., Westergaard, M., Wynn, M.T.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part I, Lecture Notes in Business Information Processing, vol. 99, pp. 169–194 (2011), https://doi.org/10.1007/978-3-642-28108-2_19

4. Chapela-Campa, D., Benchekroun, I., Baron, O., Dumas, M., Krass, D., Senderovich, A.: A framework for measuring the quality of business process simulation models. Information Systems **127**, 102447 (2025), https://doi.org/10.1016/j.is.2024.102447
5. Chipman, H.A., George, E.I., and, R.E.M.: Bayesian cart model search. Journal of the American Statistical Association **93**(443), 935–948 (1998), https://doi.org/10.1080/01621459.1998.10473750
6. Chipman, H.A., George, E.I., McCulloch, R.E.: BART: Bayesian additive regression trees. The Annals of Applied Statistics **4**(1), 266 – 298 (2010), https://doi.org/10.1214/09-AOAS285
7. Destercke, S., Dubois, D., Chojnacki, E.: Unifying practical uncertainty representations – i: Generalized p-boxes. International Journal of Approximate Reasoning **49**(3), 649–663 (2008), https://doi.org/10.1016/j.ijar.2008.07.003
8. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: Balcan, M.F., Weinberger, K.Q. (eds.) Proceedings of The 33rd International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 48, pp. 1050–1059 (20–22 Jun 2016)
9. Gneiting, T., Raftery, A.E.: Strictly Proper Scoring Rules, Prediction, and Estimation. Journal of the American Statistical Association **102**(477), 359–378 (2007), https://doi.org/10.1198/016214506000001437
10. Hersbach, H.: Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems. Weather and Forecasting **15**(5), 559–570 (2000), https://doi.org/10.1175/1520-0434(2000)015<0559:DOTCRP>2.0.CO;2
11. Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. Mach. Learn. **110**(3), 457–506 (2021), https://doi.org/10.1007/S10994-021-05946-3
12. Klein, N.: Distributional Regression for Data Analysis. Annual Review of Statistics and Its Application **11**(Volume 11, 2024), 321–346 (2024), https://doi.org/10.1146/annurev-statistics-040722-053607
13. Kneib, T., Silbersdorff, A., Säfken, B.: Rage Against the Mean – A Review of Distributional Regression Approaches. Econometrics and Statistics **26**, 99–123 (Apr 2023), https://doi.org/10.1016/j.ecosta.2021.07.006
14. Kunkler, M., Rinderle-Ma, S.: Online resource allocation to process tasks under uncertain resource availabilities. In: 2024 6th International Conference on Process Mining (ICPM), pp. 137–144 (2024), https://doi.org/10.1109/ICPM63005.2024.10723280

15. López-Pintado, O., Dumas, M., Berx, J.: Discovery, simulation, and optimization of business processes with differentiated resources. Information Systems **120**, 102289 (2024), https://doi.org/10.1016/j.is.2023.102289
16. Martin, N., Depaire, B., Caris, A.: The use of process mining in a business process simulation context: Overview and challenges. In: 2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 381–388 (2014), https://doi.org/10.1109/CIDM.2014.7008693
17. Meneghello, F., Middelhuis, J., Genga, L., Bukhsh, Z., Ronzani, M., Francescomarino, C.D., Ghidini, C., Dijkman, R.M.: Optimizing resource allocation policies in real-world business processes using hybrid process simulation and deep reinforcement learning. In: Marrella, A., Resinas, M., Jans, M., Rosemann, M. (eds.) Business Process Management - 22nd International Conference, BPM 2024, Krakow, Poland, September 1-6, 2024, Proceedings, Lecture Notes in Computer Science, vol. 14940, pp. 167–184 (2024), https://doi.org/10.1007/978-3-031-70396-6_10
18. zur Muehlen, M., Swenson, K.D.: BPAF: A standard for the interchange of process analytics data. In: zur Muehlen, M., Su, J. (eds.) Business Process Management Workshops - BPM 2010 International Workshops and Education Track, Hoboken, NJ, USA, September 13-15, 2010, Revised Selected Papers, Lecture Notes in Business Information Processing, vol. 66, pp. 170–181 (2010), https://doi.org/10.1007/978-3-642-20511-8_15
19. Orlandi, V., Murray, J., Linero, A., Volfovsky, A.: Density Regression with Bayesian Additive Regression Trees (2021), https://doi.org/10.48550/arXiv.2112.12259, arXiv:2112.12259 [stat]
20. Rozinat, A., Mans, R., Song, M., van der Aalst, W.: Discovering simulation models. Information Systems **34**(3), 305–327 (2009), https://doi.org/10.1016/j.is.2008.09.002
21. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Workflow exception patterns. In: Dubois, E., Pohl, K. (eds.) Advanced Information Systems Engineering, 18th International Conference, CAiSE 2006, Luxembourg, Luxembourg, June 5-9, 2006, Proceedings, Lecture Notes in Computer Science, vol. 4001, pp. 288–302 (2006), https://doi.org/10.1007/11767138_20
22. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Trans. Intell. Syst. Technol. **10**(4), 34:1–34:34 (2019), https://doi.org/10.1145/3331449
23. Wombacher, A., Iacob, M.: Start time and duration distribution estimation in semi-structured processes. In: Shin, S.Y., Maldonado, J.C. (eds.) Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, Coimbra, Portugal, March 18-22, 2013, pp. 1403–1409 (2013), https://doi.org/10.1145/2480362.2480626