Kratsch, Wolfgang; Manderscheid, Jonas; Röglinger, Maximilian; Seyfried, Johannes

Machine Learning in Business Process Monitoring: A Comparison of Deep Learning and Classical Approaches Used for Outcome Prediction

Business & Information Systems Engineering

Bus Inf Syst Eng 63(3):261–276 (2021)
https://doi.org/10.1007/s12599-020-00645-0

RESEARCH PAPER

Machine Learning in Business Process Monitoring: A Comparison of Deep Learning and Classical Approaches Used for Outcome Prediction

Wolfgang Kratsch · Jonas Manderscheid · Maximilian Röglinger · Johannes Seyfried

Received: 11 December 2018 / Accepted: 6 March 2020 / Published online: 8 April 2020

Abstract Predictive process monitoring aims at forecasting the behavior, performance, and outcomes of business processes at runtime. It helps identify problems before they occur and re-allocate resources before they are wasted. Although deep learning (DL) has yielded breakthroughs, most existing approaches build on classical machine learning (ML) techniques, particularly when it comes to outcome-oriented predictive process monitoring. This circumstance reflects a lack of understanding about which event log properties facilitate the use of DL techniques. To address this gap, the authors compared the performance of DL (i.e., simple feedforward deep neural networks and long short term memory networks) and ML techniques (i.e., random forests and support vector machines) based on five publicly available event logs. It could be observed that DL generally outperforms classical ML techniques. Moreover, three specific propositions could be inferred from further observations: First, the outperformance of DL techniques is particularly strong for logs with a high variant-to-instance ratio (i.e., many non-standard cases). Second, DL techniques perform more stably in case of imbalanced target variables, especially for logs with a high event-to-activity ratio (i.e., many loops in the control flow). Third, logs with a high activity-to-instance payload ratio (i.e., input data is predominantly generated at runtime) call for the application of long short term memory networks. Due to the purposive sampling of event logs and techniques, these findings also hold for logs outside this study.

Keywords Predictive process monitoring · Business process management · Outcome prediction · Deep learning · Machine learning

1 Introduction

Gaining knowledge from data is an emergent topic in many disciplines (Hashem et al. 2015), high on many organizations' agendas, and a macro-economic game-changer (Lund et al. 2013). Many researchers use data-driven techniques such as machine learning (ML), currently at the top of Gartner's Hype Cycle (Gartner Inc. 2018), to mine information from large datasets (Shmueli and Koppius 2011). Over the past decade, sophisticated ML techniques commonly referred to as deep learning (DL) have yielded a breakthrough in diverse data-driven applications. The application of such techniques in fields as natural language processing or pattern recognition in images has shown that DL can solve increasingly complex problems (Goodfellow et al. 2016).

In business process management (BPM), lifecycle activities such as the identification, discovery, analysis, improvement, implementation, monitoring, and controlling of business processes rely on data, even though data had to be collected manually so far (Dumas et al. 2018). Today,

Accepted after three revisions by Jörg Becker.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s12599-020-00645-0) contains supplementary material, which is available to authorized users.

W. Kratsch · M. Röglinger (✉)
FIM Research Center, University of Bayreuth, Project Group Business and Information Systems Engineering of the Fraunhofer FIT, Wittelsbacherring 10, 95444 Bayreuth, Germany
e-mail: maximilian.roeglinger@fim-rc.de

J. Manderscheid · J. Seyfried
FIM Research Center, University of Augsburg, Universitätsstraße 12, 86159 Augsburg, Germany

these activities are supported by process-aware information systems that record events and additional attributes, e.g., resources or process outcomes (van der Aalst et al. 2011a). Process outcomes generally reflect the positive or negative result delivered to actors involved in a process (Dumas et al. 2018). While early data-driven approaches leveraged data for process discovery or analysis (van der Aalst et al. 2011a), interest has risen in using data-driven approaches in lifecycle phases such as monitoring to gain predictive insights (Grigori et al. 2004). Thus, there is a shift from design time-oriented phases (e.g., discovery, analysis, and improvement), where data is exploited in offline mode, to runtime-oriented phases (e.g., monitoring), where data is used in real-time to forecast process behavior, performance, and outcomes (van der Aalst 2013). As for runtime use cases, predictive process monitoring is growing in importance (Maggi et al. 2014). Predicting the behavior, performance, and outcomes of process instances – e.g., remaining cycle time (van der Aalst et al. 2011b), compliance (Ly et al. 2015), sequence of process activities (Polato et al. 2018), the final or partial outcome (Teinemaa et al. 2019), or the prioritization of processes (Kratsch et al. 2017) – helps organizations act proactively in fast-changing environments.

Various predictive process monitoring approaches use ML techniques as, in contrast to rule-based monitoring techniques, there is no need to rely on subjective expert-defined decision rules (Kang et al. 2012). Moreover, the increasing availability of data lowers the barriers of using ML. Although the popularity of DL has increased in predictive process monitoring, most works still use classical ML techniques such as decision trees, random forests (RF), or support vector machines (SVM) (Evermann et al. 2016). However, a drawback of such techniques is that their performance heavily depends on manual feature engineering in case of low-level feature representations (Goodfellow et al. 2016). From a BPM perspective, DL promises to leverage process data for predictive purposes.

Some predictive process monitoring approaches already use DL (Di Francescomarino et al. 2018). Most of them strive for insights that help predict the next events during process execution (Evermann et al. 2017a; Mehdiyev et al. 2018; Pasquadibisceglie et al. 2019; Tax et al. 2017). To the best of our knowledge, only one approach uses DL for outcome prediction (Hinkka et al. 2019), which is an important prediction task as early predictions may entail substantial savings related to cost, time, and corporate resources (Teinemaa et al. 2019). The rare use of DL, especially for outcome-oriented predictive process monitoring, reflects a lack of understanding about when the use of DL is sensible. While most papers propose new approaches, only two studies compare existing approaches, but without considering DL techniques (Metzger et al. 2015, Teinemaa et al. 2019). Moreover, these studies only use one or a few logs for evaluation, although logs are known to have different properties (van der Aalst et al. 2011a). Thus, we investigate the following research question: *Which event log properties facilitate the use of DL techniques for outcome-oriented predictive process monitoring?*

To address this question, we compare the performance of different ML and DL techniques for a diverse set of logs in terms of established evaluation metrics. To obtain transferable results and related propositions, we combined data-to-description (Level-1 inference) and description-to-theory (Level-2 inference) generalization, as included in Lee and Baskerville (2003) generalization framework for information systems research. This required to purposively sample both techniques and logs. As for the techniques, we selected long short term memory networks (LSTM) and simple feedforward deep neural networks (DNN), as representatives of DL, as well as RF and SVM as representatives of classical ML techniques. Regarding the event logs, we selected five publicly available logs that cover most conceivable log types, e.g., in terms of the number of process instances, number of events, or data attributes. These logs were the BPI Challenge 2011 (BPIC11), the BPI Challenge 2013 (BPIC13), the road traffic fine management process (RFTM), the production log (PL), and the review log (RL).

Our study is structured as follows: In Sect. 2, we outline the background on data-driven approaches in BPM and ML as a predictive process monitoring technique. Section 3 outlines our study design, followed by details related to data collection and analysis in Sect. 4. In Sect. 5, we present our results. In Sect. 6, we discuss the results, highlight limitations, and point to future research.

2 Theoretical Background

2.1 Data-Driven Approaches in Business Process Management

BPM is the science and practice of overseeing how work is performed to ensure consistent outcomes and to leverage opportunities for process improvement (Dumas et al. 2018). Interdisciplinary in nature, BPM combines knowledge from information technology and management sciences. BPM activities are commonly organized along lifecycle phases, such as identification, discovery, analysis, improvement, implementation, monitoring, and controlling (Dumas et al. 2018). In terms of the lifecycle phases analysis, monitoring, and controlling, one can distinguish between data-driven and model-based approaches (van der Aalst 2013). Data-driven approaches help analyze and

monitor running processes to determine how well they are performing with respect to performance metrics and objectives (Dumas et al. 2018). Additionally, data-driven approaches serve as input for model-based approaches, which in turn support process analysis during the redesign phase (van der Aalst 2013).

Some data-driven approaches use process data for descriptive purposes, such as process discovery or analysis (van der Aalst et al. 2011a), whereas others use process data to gain predictive insights during execution (Grigori et al. 2004). As a subset of such online data-driven approaches, predictive process monitoring exploits data related to past and current instances to predict the behavior, performance, and outcome of currently running instances (Breuker et al. 2016; Maggi et al. 2014; Conforti et al. 2016). The predictive process monitoring approaches proposed over the last years can be classified according to the underlying prediction task (Marquez-Chamorro et al. 2018): performance predictions such as the remaining cycle time of running instances (van der Aalst et al. 2011b; Polato et al. 2014; Rogge-Solti and Weske 2015; van Dongen et al. 2008), predictions regarding partial or final outcomes of process execution (Castellanos et al. 2005; Conforti et al. 2013; Kang et al. 2012), business rule violations (Metzger et al. 2015; Maggi et al. 2014; Leontjeva et al. 2015; Di Francescomarino et al. 2016), and predictions of the next event(s) (Evermann et al. 2016; Breuker et al. 2016; Lakshmanan et al. 2013; Ceci et al. 2014; Mehdiyev et al. 2017; Schönig et al. 2018), maybe including further information such as the performing resource (Evermann et al. 2017b).

Most predictive process monitoring approaches build on data from past process instances, also referred to as event logs. Event logs record series of events with each event referring to a distinct task in a process instance. Beyond standard XES event attributes[1] (e.g., event name, resource, timestamp, and duration), event logs may store attributes that provide additional information (so-called payload data) about events and their context (van der Aalst 2014; vom Brocke et al. 2016). As with standard attributes, additional attributes refer to a distinct event (e.g., the outcome of a performed activity) or – in case they describe context information – to an instance (e.g., the birth date of a patient) (Sindhgatta et al. 2016; Leontjeva et al. 2015). The characteristics of an event log highly depend on the underlying process (van der Aalst et al. 2011a; Russell et al. 2005). In general, business processes can be classified according to their complexity with regard to control flow, data flow, and resource involvement (Cardoso et al. 2006). By contrast, event logs are described based on technical characteristics such as the number of instances, the number of events, the number of distinct activities, or the number of process variants (van der Aalst et al. 2011b; Kratsch et al. 2017; Maggi et al. 2014; Leontjeva et al. 2015; Augusto et al. 2019). Merging both perspectives, event logs can be classified from a data perspective (including resource information) and a control flow perspective. In terms of the data perspective, one can distinguish between the availability of instance and activity payload data, and differentiate the type of payload data (i.e., numeric or categorical). Regarding the control flow perspective, the number of instances and variants, as well as the average number of events and activities per instance, are distinguishing criteria.

2.2 Machine Learning as a Predictive Process Monitoring Technique

ML is a subfield of artificial intelligence that uses real-world knowledge to make human-like decisions without defined rules (Goodfellow et al. 2016). ML uses statistical methods to learn structural patterns in typically large datasets in a (semi-) automated manner (Witten et al. 2017). Typical use cases of ML are classification, clustering, regression, and anomaly detection (Witten et al. 2017). Moreover, ML techniques can be divided into supervised and unsupervised learning (Bishop 2010; Witten et al. 2017). Supervised learning takes historical data that has already been classified by an external source and uses it for reproducing classifiers. By contrast, unsupervised learning algorithms process input data to gain insights by themselves (Bishop 2010). Whereas unsupervised learning is often used to group similar cases with an unclear definition of classes (e.g., for anomaly detection), supervised learning is appropriate when classifying cases according to predefined classes. With predictive process monitoring, possible outcomes of tasks and instances or predefined business goals can be used to specify relevant classes. Thus, predictive process monitoring belongs to the field of supervised learning.

Classical ML techniques in supervised learning are RF (Breiman 2001), logistic regressions (Hosmer et al. 2013), SVM (Cortes and Vapnik 1995), and shallow neural networks (i.e., single-layer perceptrons) (Haykin 2009). The popularity of these techniques reflects the fact that the underlying algorithms are easy for humans to understand. The use of RF, in particular, has become commonplace in recent studies on outcome-oriented predictive process monitoring (Teinemaa et al. 2019). RF is an ensemble learning technique that encompasses many decision trees, building a forest. To classify instances, each tree in a forest predicts a class, and the final classification is predicated on

[1] With XES (eXtensible Event Stream), the IEEE Task Force on Process Mining defined a standard for event logs; please refer to http://www.xes-standard.org.

the class with the most votes (Breiman 2001). In doing so, RF avoids overfitting when using single decision trees. SVM aim at finding a hyperplane that separates observations into two classes by maximizing the minimal distance. By using the "kernel trick" (i.e., solving the problem in higher dimensions), SVM can also separate non-linear observations (Witten et al. 2017).

The performance of classical ML approaches is highly dependent on the representation of input data (Goodfellow et al. 2016). For highly correlated features, in particular, RF produces substantially better results if the data is preprocessed in a way that aggregates basic features into features with higher information richness. This aspect of preprocessing is also known as feature engineering (Witten et al. 2017). Representation learning aims at automating manual feature engineering by discovering both the mapping of features to labels and the most important feature combinations, i.e., a high-level structure (Goodfellow et al. 2016). However, in cases where input features are highly interdependent, it is very difficult to find high-level representations (Goodfellow et al. 2016).

DL reduces manual feature engineering effort by applying the divide-and-conquer principle, which introduces representations that are themselves expressed in terms of simpler representations (Goodfellow et al. 2016). DL is a relatively new term for the application of deep neural networks (DNNs) (Witten et al. 2017). Until 2006, DNNs were generally thought to be too difficult to train (Goodfellow et al. 2016). Innovations in algorithms and hardware have enabled the application of DL in productive services, for example, in the recognition of a photo's location without geographical data in Google photos or video recommendations on YouTube (Schmidhuber 2015; Weyand et al. 2016; Covington et al. 2016). In contrast to feedforward DNNs that only allow a unidirectional information flow, RNNs enable a bidirectional information flow among network layers, facilitating the extraction of temporal changes on a feature level. RNNs have shown their potential to process sequential input data such as stored in event logs. In recent works, a specialized version of RNN, known as LSTM, has shown its potential for predictive process monitoring (Hinkka et al. 2019; Schönig et al. 2018). LSTM unfolds the input sequence in different time steps, each involving a fixed number of input features. By means of weighted forget gates, each LSTM cell can decide whether and to what extent information from time steps in the near or distant past is considered (Gers et al. 2000). Thus, the updating mechanism in the memory cell of an LSTM allows for information to be stored for longer time periods than in common RNN, a feature that helps overcome the gradient vanishing problem in recurrent network architectures (Bengio et al. 1994).

2.3 Performance Evaluation of Machine Learning Classifiers

Performance evaluation is crucial when building ML classifiers. The class predicted by the classifier must be compared to the actual class for each data point. If both classes (predicted and actual) match, it is referred to as a "true" prediction (T); otherwise, as a "false" prediction (F). In case of a binary prediction problem, one can develop a 2×2 confusion matrix denoting the true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). On this foundation, several metrics can be calculated. Table 1 gives an overview of common evaluation metrics, which we also used in our research, including a formal definition, value ranges, and best values. The most common metric is accuracy. By aggregating all predictions in a single metric, accuracy offers a convenient assessment. Using accuracy as a single evaluation metric, however, may produce misleading results in case of imbalanced data. For instance, if there are 95 normal and 5 problematic instances, a classifier can reach a very high accuracy of 95% by classifying all instances as normal. Nevertheless, we would not describe a classifier that never fires an alarm in the case of problematic instances as one that performs well.

Hence, class-wise metrics are needed. Specifically, in case of imbalanced target variables, minority classes must

Table 1 Overview of evaluation metrics

Metric	Formula	Value range	Best value
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	[0; 1]	1
Precision (p)	$\frac{TP}{TP+FP}$	[0; 1]	1
Recall (r)	$\frac{TP}{TP+FN}$	[0; 1]	1
F-beta score (F_β)	$\left(1+\beta^2\right) \cdot \frac{p \cdot r}{\left(\beta^2 \cdot p\right)+r}$	[0; 1]	1
Area under the receiver operating characteristic curve (ROC AUC)	$\frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)$	[0.5; 1]	1

be considered as they often represent instances that are important for predictive monitoring. To assess how many instances predicted to be problematic proved to be so, we use the precision metric. In contrast, the recall metric indicates how many actual problems were correctly predicted as problems. Higher precision means fewer "false alarms" (low alpha error), whereas higher recall indicates that for more "true problems" an alarm has been fired (low beta error). In fact, recall can be interpreted as class-wise accuracy (Witten et al. 2017). Recall and precision can also be aggregated to a single metric. One manifestation is the F-beta score F_β, where β allows the balance of recall and precision to be adjusted. Another metric specifically capturing a classifier's ability to avoid false classification is the area under the receiver operating characteristic curve (ROC AUC). Since no threshold value is required for weighting *FP* or *FN*, the *ROCAUC* is a rather objective metric for aggregated performance (Sokolova and Lapalme 2009).

To make accurate predictions, classifiers should achieve high levels of both precision and recall. However, increasing one of these may decrease the other. For each prediction, there is a trade-off between both metrics – e.g., in airport security, since passenger screening aims at preventing security risks, scanners can be set to trigger alarms on low-risk items such as seat belt buckles and keys (low precision) to increase the likelihood of identifying dangerous items and minimize the risk of missing items that pose a threat (high recall). This behavior can be visualized using the ROC AUC.

3 Study Design

To infer propositions about which log properties facilitate the use of DL techniques for outcome-oriented predictive process monitoring, we compared the performance of DL and classical ML techniques for multiple event logs. To do so, we followed the reference process for big data analysis in information systems research proposed by Müller et al. (2016), which comprises three phases: data collection, data analysis, and result interpretation.

In the data collection phase, we first compiled five event logs (i.e., BPIC11, BPIC13, RTFM, PL, and RL). Detailed information about these logs is shown in Sect. 4.1, including rationales why outcome-oriented predictive process monitoring makes sense. To purposively sample the event logs, we derived properties from diverse event logs and classified the selected ones accordingly as shown in Sect. 4.2. As discussed in Sect. 2.1, we classified the logs according to a data and a control flow perspective, which led to two pairs of log properties per perspective. Hence, we ensured to have covered the most conceivable log types. To conclude the data collection, we preprocessed the event data and defined input features as well as targets required for the application of supervised learning techniques.

In the data analysis phase, we first documented the preprocessing in Sect. 4.3 (Müller et al. 2016). We then built classifiers after each executed activity contained in the logs. In the sense of purposive sampling, we chose RF and SVM as representatives of classical ML techniques, as they are commonly used for predictive monitoring (Marquez-Chamorro et al. 2018). As for DL, we chose LSTM as they count among the most advanced techniques and a simple feedforward DNN as an entry-level technique (Tax et al. 2017). To analyze the classifiers, we evaluated their performance after each executed activity. We stopped at the tenth activity, as we were particularly interested in early predictions. From a business perspective, the rationale is that the earlier reliable outcome predictions can be made, the more valuable they are. For the RTFM log, we only built classifiers for the first six activities as approximately 91% of the contained instances terminate before this point. To prevent issues that result from an unfavorable configuration of classifiers, we performed a random search-based optimization of hyperparameters (Bergstra and Bengio 2012). When training and testing classifiers, we also employed tenfold cross-validation as recommended by Fushiki (2011).

In the result interpretation phase, we built on Lee and Baskerville (2003) generalization framework for information systems research. By combining an empirical (E) and a theoretical (T) layer, this framework proposes four generalization strategies: data-to-description (EE), theory-to-description (TE), description-to-theory (ET), and concepts-to-theory (TT). In our work, we used data-to-description and description-to-theory strategies. Also referred to as Level-1 inference (Yin 1994), data-to-description generalization takes empirical data as input, which is condensed into higher-level yet still empirical observations or descriptions. This strategy also covers the well-known statistical sample-to-population generalization. Description-to-theory generalization, which is also referred to as analytical generalization or Level-2 inference (Yin 1994), aims at inferring theoretical statements in the form of propositions, i.e., "variables and the relationships among them" (Lee and Baskerville 2003, p. 236), from empirical observations or descriptions. As for Level-1 inference, we analyzed the performance of the selected techniques per event log in terms of evaluation metrics and related statistical measures (i.e., mean and standard deviation) (Sect. 5.1). As for Level-2 inference, we identified relationships between the techniques' performance across the logs and related these cross-log observations to the log properties introduced in Sect. 2.1. Moreover, we analyzed

the distribution of class labels as the target variable of outcome-oriented predictive process monitoring. On this foundation, we inferred propositions about which log properties facilitate the use of DL techniques for outcome-oriented predictive process monitoring (Sect. 5.2). Due to the purposive sampling of logs and techniques, we can claim that these propositions also hold for event logs outside those we used in our research (Lee and Baskerville 2003).

4 Data Collection and Analysis

4.1 Description of the Used Event Logs

The IEEE Task Force on process mining keeps real-world event logs in a public data repository, making them available for academic purposes. Most process mining approaches use these logs to evaluate their work (van der Aalst et al. 2011a, b; Kratsch et al. 2017). We also used this data source and analyzed the included logs in terms of their properties (e.g., number of instances and process variants, and type of payload data). On this foundation, we selected a diverse sample of four real-world event logs and one synthetic event log. As for the real-world logs, we used the BPIC logs BPIC11 (van Dongen 2011) and BPIC13 (sub-log incidents) (Steeman 2013), the RTFM log (Mannhardt et al. 2016), and the PL (Levy 2014). We also examined the synthetic review log (RL) (van der Aalst 2010). Table 2 provides an overview of the descriptive statistics of these event logs, which we calculated with ProM (van Dongen et al. 2008).

BPI Challenge 2011 The BPIC 2011 log pertains to a healthcare process, containing the executions of a process related to the treatment of patients diagnosed with cancer in the Gynecology department of a large Dutch academic hospital. Each instance refers to the treatment of one patient. The event log contains instance and activity payload data. For example, age, diagnosis, and treatment codes are instance-level attributes, whereas activity code, number of executions, specialism code, and group are activity-level attributes. Some treatments are marked with the suffix "urgent" (Bose and van der Aalst 2011). The assignment of treatments may be facilitated by the early detection of urgent instances.

BPI Challenge 2013 The BPIC13 log refers to an incident management process with three support levels. If incident management processes are structured according to different support levels, a typical problem is a push-to-front mechanism (i.e., assignment of most instances to the first lane even though there is a lack of know-how) or back-and-forth movement between the support levels. Hence, it would be helpful to be able to predict whether a case will end up in third-level support. Such cases could be assigned in a more target-oriented manner and resolved faster.

Road Traffic Fine Management Process The RTFM log contains a process for handling traffic regulation infringements. Depending on the nature of the infringement, instances are either treated in a lean process (minor infringements) or are negotiated with the participation of a judge (serious infringements). A judge also intervenes if there is a disagreement regarding the sentence or if fines are not paid. If traffic fine regulation were based on early predictions as to the necessity of involving a judge, instances could be resolved in a more targeted manner and with shorter processing times.

Production Log The PL originates from an enterprise resource planning system recording a production process. This process is structured according to different departments and work centers and contains automated activities of production machines as well as manual work (e.g., quality checks). The log contains several additional attributes such as the number of goods rejected during each iteration of the process. An early prediction of an above-average rejection would help process managers to identify problems (e.g., waste in the making) and to trigger corrective actions earlier.

Review Log The RL covers a simulated paper review process of an academic journal. The editor must accept or reject a paper based on the recommendations of multiple reviewers. The editor may consult as many reviewers as necessary to decide. If the editor decides to request further

Table 2 Descriptive statistics of the event logs

Event log	Number of instances	Number of variants	Number of activity types	Number of events	Min/Max/Avg events per instance	Min/Max/Avg activities per instance
BPIC11	1143	981	624	150,291	1/1814/131	1/113/33
BPIC13	7554	1630	13	65,533	1/123/9	1/9/4
RTFM	150,370	231	11	561,470	2/20/4	2/10/4
PL	225	221	55	9086	2/350/40	2/28/12
RL	10,000	4118	14	236,360	11/86/24	11/15/15

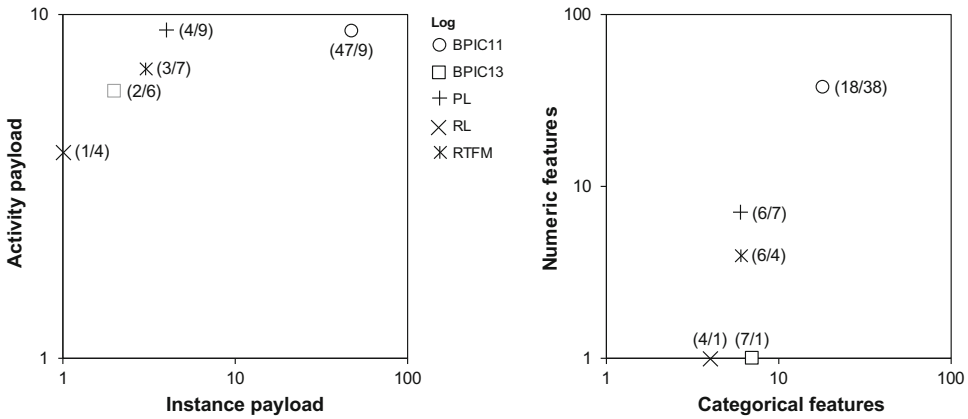


Fig. 1 Data perspective of log classification: activity-to-instance ratio (left, **a**) and numeric-to-categorical ratio (right, **b**) of payload data (logarithmic scales)

reviews, the process instance loops activities several times. Apart from the recommendation of the reviewers, no information is attached to the process. An early prediction about whether a paper will be rejected or accepted would help the editor prevent superfluous review rounds and streamline the review process.

4.2 Classification of the Used Event Logs

As discussed in Sect. 2.1, event logs can be classified according to their properties in terms of a data and a control flow perspective. Table 2 presents the descriptive statistics which we used for classifying the selected logs. Figures 1 and 2 visualize both perspectives. For better visualization, we used logarithmic scales.

Figure 1a shows the activity-to-instance payload ratio. The higher the number of instance-level attributes, the more information is available at early points of prediction to capture the internal context of a process instance. Conversely, in case of many activity-level attributes, the amount of information available for outcome predictions strongly increases during runtime. Most logs analyzed in our research contain more activity-related data (e.g., RTFM, RL, PL, and BPIC13). Only BPIC11 contains more instance- than activity-level attributes. Figure 1b covers the numeric-to-categorical payload ratio. In the context of predictive analytics, this characteristic is information, as some ML techniques perform poorly on categorical features and much better on numerical input. Regarding the logs used in our research, three contain more categorical than numeric features (e.g., BPIC13, RL, and RFTM), while the other two contained more numeric features.

Figure 2 classifies the used event logs from a control flow perspective. Figure 2a illustrates events-to-activities ratio. The more a log is situated above the bisector, the more the related process includes iterations and loops. For example, administration processes such as RTFM and BPIC13 perform a few standardized activities several times, whereas BPIC11 and PL include more but highly specific activities with few repetitions. The classification shows that a low average number of events per instance implies a low average number of activity types. Finally, Fig. 2b illustrates the variants-to-instances ratio. Logs containing few unique variants typically originate from highly standardized processes, e.g., the administration of road traffic fines (i.e., RTFM). In contrast, the treatment of patients in a hospital (i.e., BPIC11) tends to require individualized routing where almost every instance leads to a specific variant.

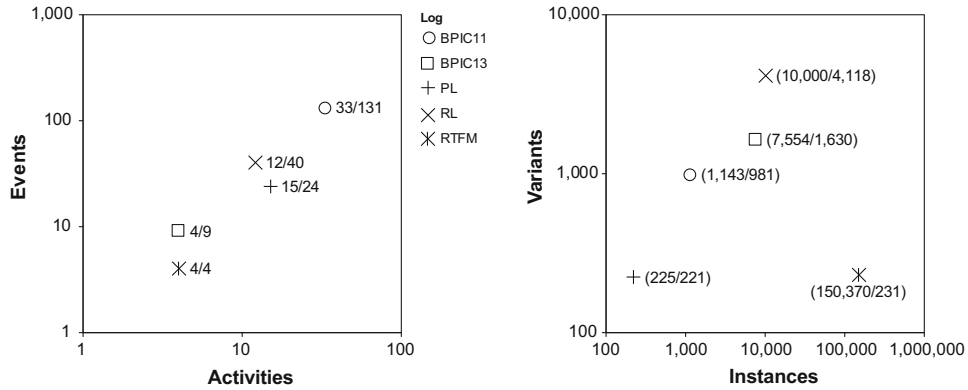


Fig. 2 Control flow perspective of log classification: events-to-activity ratio (left, **a**), and variants-to-instances ratio (right, **b**) (logarithmic scales)

Table 3 Overview of labeling rules

	BPIC11	BPIC13	RTFM	PL	RL
Class 0	Normal patients (875)	Resolved in support lane 1 (4546)	No judge involved (149,815)	Above average rejects (50)	Accept (4932)
Class 1	Urgent patients (268)	Resolved in support lanes 2 or 3 (3008)	Judge involved (555)	Up to average rejects (175)	Reject (5068)

4.3 Data Preprocessing

4.3.1 Labeling of Process Instances

As the applied techniques require labeled records, we built classes and labeled historic instances according to predefined labeling rules that we directly derived from attributes contained in the logs. In line with the idea of outcome prediction, we labeled all instances according to their outcomes or relevant partial outcomes. Table 3 provides an overview of all labeled event logs including the number of instances per class.

4.3.2 Sequence Encoding

As stated in Sect. 3, we built classifiers for each log after every executed activity. We refer to this as prediction time points. To ensure comparability, we used a similar log encoding for LSTM, DNN, SVM, and RF. We encoded every log for each prediction time point individually. Each encoded log only contained the information available at the prediction time point. In the encoded log, each process instance is represented as a feature vector. The feature vector includes all information about the instance and about all activities executed before the prediction time point, i.e., if the prediction time point is after the fourth activity, only the attributes for activities one to four are included in the encoded log. To prepare the encoded logs, we implemented a prototype[2] that follows a structured workflow and returns encoded logs for each prediction time point (Appendix B, available online via http://link.springer.com):

1. **Labeling and log-specific preprocessing** As supervised learning requires labeled records, we applied the predefined labeling (Sect. 4.3.1). Moreover, we performed log-specific preprocessing, e.g., standardizing date and time formats.
2. **Log cleaning** The number of activities included in the encoded log depends on the prediction time point for which the encoded log is built. Therefore, we removed instances with an insufficient number of activities. For instances with a higher number of activities than the prediction time point, we removed all activities that were executed after the prediction time point.
3. **Attribute cleaning** For each of the remaining instances, we collected all existing attributes. If an instance did not include all instance variables existing over all instances, the instance-level variable was added with *0* as replacement value. Instance-level attributes with an occurrence of below 99% were removed because too many replacement values would bias the classification. The same procedure was applied to event-level attributes.
4. **Creating feature vectors** In the final step, we assembled feature vectors for the instances included in the encoded log. LSTM requires feature vectors that have the same number of attributes in each time step. After the previous step, each event contains the same attributes. However, to include the instance-level attributes for prediction time points after the first event, they had to be added to each event. Doing so keeps the number of attributes constant in each time step while including both the event-level and instance-level attributes. Finally, we applied a one-hot-encoding[3] to attributes with categorical values as these cannot be handled by LSTM and DNN. The resulting log corresponds to an index-based encoding and includes all available information from instance- and event-level attributes (Leontjeva et al. 2015).
5. **Rearranging three-dimensional feature vectors** For LSTM, we rearranged the feature vectors in order to obtain the required three-dimensional data structure. The first dimension contains the events representing the time steps. In the second dimension, we fed all attributes associated with the respective event.

[2] The log encoder can be found on https://tinyurl.com/r39y4cu.

[3] http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html.

4.4 Implementing the Classifiers

To build LSTM and DNN classifiers, we implemented a two-stage learning scheme using Python. In the first stage, we optimized hyper-parameters using a simple case-based stratified train-validation-test split (i.e., 63.75%, 11.25%, 25%) by testing multiple hyper-parameter settings using a random search. We decided to randomize the parameter search, trying 20 different parameter settings instead of testing all existing combinations of possible values (i.e., grid search). This is reasonable as random search has been shown to lead to similar results more efficiently (Bergstra and Bengio 2012). Subsequently, we applied tenfold cross-validation to the best classifier to obtain stable out-sample results. To implement the hyper-parameter optimization for LSTM and DNN, we used the lightweight Python wrapper Hyperas,[4] which extends Keras with Hyperopt[5] functionalities. Appendix C shows the parameter ranges of the classifiers and provides a short description of the values we used. For RF and SVM classifiers, we used the function *RandomizedSearchCV* from Scikit-Learn,[6] which includes tenfold cross-validation (Zhang 1993). This is possible as the hyper-parameter optimization of RF and SVM requires much less computational effort and, hence, there is no need to separate the optimization step from the cross-validation step. The source files can be found on https://tinyurl.com/r39y4cu.

5 Result Interpretation

5.1 Observations for Individual Logs (Level-1 Inference)

As outlined in Sect. 3, we first analyzed each event log individually to provide a foundation for the identification of cross-log observations. Table 5 shows the results. All reported evaluation metrics are average scores compiled over all folds of the cross-validation. Each row represents one event log. The left-hand diagrams show the accuracy and the F-Score per classifier in relation to the prediction time points. By setting β to 1, we weigh recall and precision equally strong. The diagrams on the right illustrate the number of instances used for building the classifiers and the number of input features in the encoded log depending on the prediction time point. Moreover, the tables embedded below the diagrams show the mean and standard deviation of the evaluation metrics over all prediction time points. More details are included in Appendix A. Finally, Table 4 contains specific observations per event log.

5.2 Observations Across Logs and Inference of Propositions (Level-2 Inference)

Based on the individual log analysis, we made observations regarding the classifiers' performance across logs. We made one general observation (O1) and three specific observations (O2 to O4), which are shown in Table 4. Thereby, O2 and O3 relate to the classes of ML techniques, i.e., DL and classical ML, while O4 refers to LSTM, the most sophisticated DL technique investigated. By tracing the specific observations back to the log properties introduced in Sect. 2.1, we inferred propositions that answer our research question. Moreover, we looked for patterns regarding the distribution of class labels representing the target variables of outcome-oriented predictive process monitoring. As we purposively sampled the event logs and techniques, we can claim that the propositions also hold for logs outside our study (Lee and Baskerville 2003). As we observed a general outperformance of DL (O1), we only formulate propositions if the presence of distinct log properties causes a substantial outperformance of DL.

O1: DL classifiers generally outperform classical ML classifiers regarding accuracy and F-Score In terms of accuracy and F-Score, we observed a general outperformance of DL classifiers across all selected logs. On average, the DL classifiers lead to an 8.4 pp higher accuracy as well as to a 4.8 pp higher F-Score compared to classical ML classifiers.

O2: DL classifiers substantially outperform classical ML classifiers regarding accuracy and F-Score for logs with a high variant-to-instance ratio In addition to the general outperformance of DL, we observed substantial outperformance for PL and BPIC11. Averaging the results for both logs, DL classifiers lead to a 9.5 pp higher accuracy and to a 6.4 pp higher F-Score. Both PL and BPIC11 feature a high variant-to-instance ratio. That is, almost every instance needs to be treated as a distinct variant, and there are no standard variants. The outperformance for logs with a high variant-to-instance ratio is rooted in the circumstance that DL can extract sub-variants (i.e., sequences of activities that occur in many variants). In line with the literature, we also observed that high variability of training samples specifically impairs the performance of RF, whereas DL benefits from the possibility to generate high-level features automatically (Goodfellow et al. 2016). Overall, this observation leads to proposition P1.

[4] https://github.com/maxpumperla/hyperas.

[5] https://github.com/hyperopt/hyperopt.

[6] https://github.com/scikit-learn/scikit-learn/blob/master/sklearn/model_selection/_search.py.

Table 4 Observations across logs

Observation	Underlying event logs	Underlying evaluation metrics
O1	All event logs	Accuracy (mean: +8.4 pp), F-Score (mean: +4.8 pp)
O2	BPIC 11, PL	Accuracy (mean: +9.5), F-Score (mean: +6.4 pp)
O3	BPIC11, BPIC13, PL	ROC AUC (mean: +13.7 pp), Class frequency
O4	BPIC13, PL	Accuracy (mean: +7.3 pp), F-Score (mean: +5.5 pp)

Proposition P1 *Logs with a high variant-to-instance ratio (i.e., many non-standard cases) facilitate the use of DL.*

O3: DL classifiers substantially outperform classical ML classifiers regarding ROC AUC for logs with a high event-to-activity ratio and imbalanced class labels For PL, BPIC11, and BPIC13, we found that DL leads to a considerably higher ROC AUC score, which points to a more balanced classification in terms of less alpha and beta errors. For these logs, the ROC AUC score of DL classifiers is on average 13.7 pp higher as compared to RF and SVM. BPIC11, BPIC13, and PL contain a high event-to-activity ratio per instance. That is, there are many loops in the control flow. Additionally, two logs feature imbalanced labeling of the target variable (BPIC11: 268/875; PL: 50/175) (Table 2). However, RTFM (555/149,815), the only log for which ML classifiers deliver the best average ROC AUC (i.e., DL on average -6.6 pp compared to classical ML), shows that the outperformance of DL disappears in case of too few training samples of the minority class. Based on this observation, we infer proposition P2 (Table 5).

Proposition P2 *DL techniques perform more stably in case of imbalanced target variables, especially for logs with a high event-to-activity ratio (i.e., many loops in the control flow).*

O4: LSTM substantially outperforms DNN regarding accuracy and F-Score for logs featuring a high activity-to-instance payload ratio Finally, we observed that, in PL and BPIC13, LSTM substantially outperform simple DNN. For these logs, LSTM shows on average a 7.3 pp higher accuracy and 5.5 pp better F-Score. PL and BPIC13 show a high activity-to-instance payload ratio. This log property differentiates PL and BPIC13 from BPIC11, where also a high amount of instance-level payload is available. In case of BPIC11, much data is already available prior to execution, whereas for PL and BPIC13, most data available for outcome prediction is generated during runtime. In such situations, sequential models such as LSTM decide on their own whether most recent or older (up to instance-level) features are more important. In contrast, non-sequential DL techniques struggle with an increasing number of features with low predictive power. Such techniques are overloaded by the growing amount of data during execution and can no longer differentiate between important and unimportant features (Goodfellow et al. 2016). On this foundation, we infer proposition P3.

Proposition P3 *Logs with a high activity-to-instance payload ratio (i.e., input data is predominantly generated at runtime) facilitate the use of LSTM.*

6 Conclusion

6.1 Summary

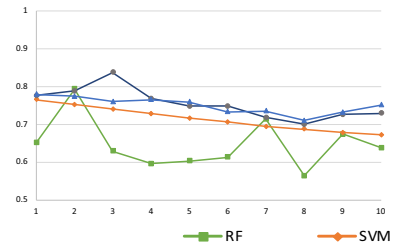
Although DL has experienced great progress over the last years, its potential for outcome-oriented predictive process monitoring yet needs to be explored. As things stand, there is a lack of knowledge related to which log properties facilitate the use of DL techniques in this domain. To address this gap, we performed a structured comparison of the performance of two DL techniques (i.e., LSTM and DNN) and two classical ML techniques (i.e., RF and SVM) for five publicly available logs (i.e., BPIC11, BPIC13, RTFM, PL, and RL). We purposively sampled not only the techniques, but also the used logs in line with properties from a data perspective (e.g., number of instance payload attributes and number of activity payload attributes) and a control flow perspective (e.g., number of variants, number of instances, and number of events per instance). We also combined the data-to-description (Level-1 inference) and the description-to-theory (Level-2 inference) generalization strategy as included in Lee and Baskerville (2003) generalization framework for information systems research. While Level-1 inference yielded insights into the classifiers' performance per log in terms of established evaluation metrics (i.e., accuracy, F-Score, and ROC AUC), Level-2 inference resulted in propositions about which log properties facilitate the use of DL techniques for outcome-oriented predictive process monitoring.

In a nutshell, our observations led us to conclude that DL generally outperforms classical ML approaches when it comes to outcome-oriented predictive process monitoring. Specifically, we observed that a high variant-to-instance ratio (i.e., many non-standard cases) and a high activity-to-instance payload ratio (i.e., input data is predominantly

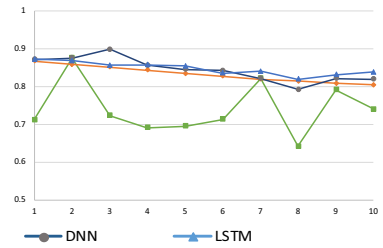
Table 5 Observations for individual logs

BPIC11

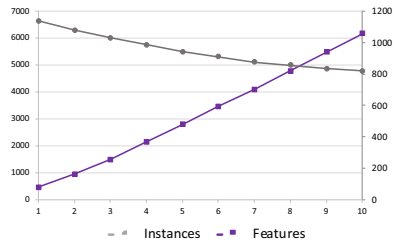
Accuracy



F-Score



Number of Instances vs. Number of Features



Log-specific evaluation metrics:

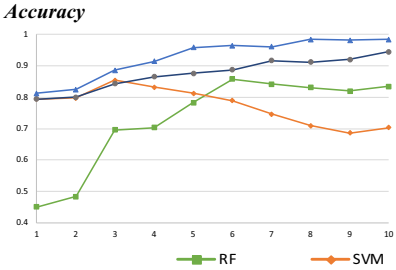
	Accuracy		*F-Score*		*ROC AUC*	
	Mean	***Std. Dev.***	***Mean***	***Std. Dev.***	***Mean***	***Std. Dev.***
RF	0.6473	0.0627	0.7402	0.0656	0.6128	0.0581
SVM	0.7137	0.0318	0.8326	0.0216	0.5632	0.0213
DNN	0.7539	0.0400	0.8440	0.0318	0.7539	0.0400
LSTM	0.7498	0.0206	0.8470	0.0162	0.7293	0.0264

Log-specific observations:

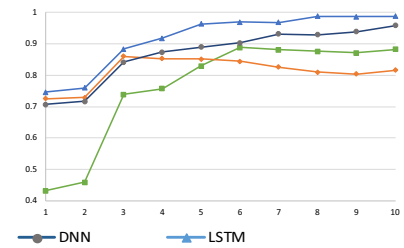
- Accuracy and F-Score: The DL classifiers outperform the classical ML classifiers for every prediction time point. DNN and LSTM perform similarly, SVM substantially outperforms RF for most prediction time points. In prediction points two and seven, RF delivers higher accuracy than SVM.
- ROC AUC: DNN shows on average the highest AUC, LSTM performs second best. RF and SVM deliver similarly low AUC values. RNN and RF yield the most unstable AUC over time, as indicated by a high standard deviation.
- Temporal stability: LSTM, and SVM show high temporal stability regarding accuracy and F-Score.
- Number of instances and features: The number of input features grows strongly between the first and the tenth activity. This can be explained by the high number of categorical features and the high activity payload. The number of process instances which terminate between the first and the tenth event is rather limited. Therefore, the number of instances shows high temporal stability.

BPIC13

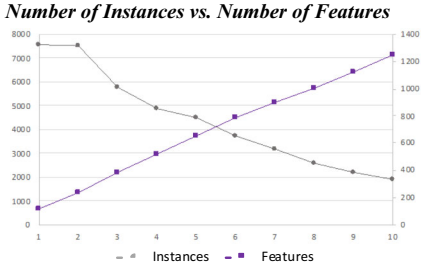
Accuracy



F-Score



Number of Instances vs. Number of Features



Log-specific evaluation metrics:

	Accuracy		*F-Score*		*ROC AUC*	
	Mean	***Std. Dev.***	***Mean***	***Std. Dev.***	***Mean***	***Std. Dev.***
RF	0.7299	0.1419	0.7616	0.1658	0.7807	0.1644
SVM	0.7725	0.0580	0.8118	0.0485	0.8482	0.0423
DNN	0.8761	0.0511	0.8688	0.0894	0.8697	0.0600
LSTM	0.9269	0.0620	0.9166	0.0881	0.9226	0.0688

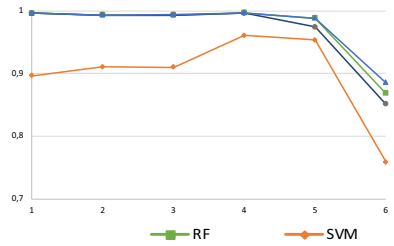
Log-specific observations:

- Accuracy and F-Score: The DL techniques show higher overall accuracy and a lower standard deviation. Compared to DNN, LSTM shows a substantial dominance, especially in later prediction time points. Concerning the classical techniques, SVM shows advantages in earlier prediction time points, whereas RF yields better results after the sixth activity.
- ROC AUC: All classifiers deliver good results regarding the ROC AUC. The DL classifiers outperform the classical ML classifiers. However, DNN only slightly outperforms SVM, while RF falls behind.
- Temporal stability: DL techniques show higher temporal stability than RF and SVM. The performance advantage regarding the accuracy and the F-Score is especially high for earlier prediction time points.
- Number of instances and features: The number of instances reduces substantially over time, while the number of features increases.

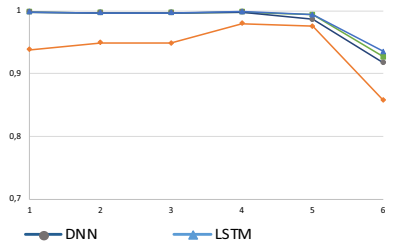
Table 5 continued

RTFM

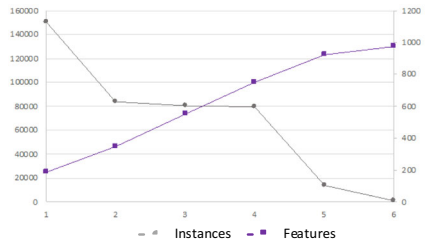
Accuracy



F-Score



Number of Instances vs. Number of Features



Log-specific evaluation metrics:

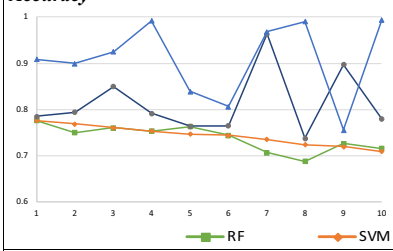
	Accuracy		*F-Score*		*ROC AUC*	
	Mean	***Std. Dev.***	***Mean***	***Std. Dev.***	***Mean***	***Std. Dev.***
RF	0.9613	0.0608	0.9792	0.0334	0.7794	0.1145
SVM	0.8983	0.0731	0.9412	0.0444	0.6001	0.1446
DNN	0.9673	0.0577	0.9822	0.0322	0.6071	0.1439
LSTM	0.9757	0.0405	0.9866	0.0230	0.6410	0.1355

Log-specific observations:

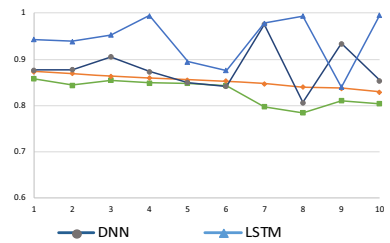
- Accuracy and F-Score: All techniques deliver high accuracy scores but the accuracy drops after the fifth prediction time point. While LSTM and DNN perform quite similarly, RF has advantages over SVM.
- ROC AUC: As opposed to accuracy and F-Score, no classifier delivers very high values. This is due to the fact that the classes are especially imbalanced in this log. No class of classifiers outperforms the other. DNN delivers rather poor results and RF delivers the best score over all classifiers.
- Temporal stability: The performance regarding the accuracy and F-Score drops for all classifiers after the fifth prediction time point.
- Number of instances and features: The number of instances included drops after the first and again after the fourth activity. This may explain why all performance metrics drop after the fourth activity.

PL

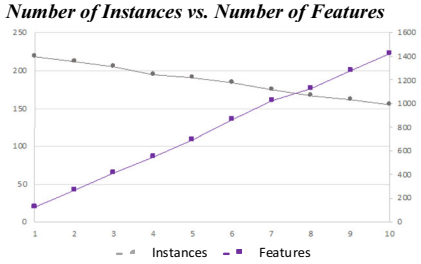
Accuracy



F-Score



Number of Instances vs. Number of Features



Log-specific evaluation metrics:

	Accuracy		*F-Score*		*ROC AUC*	
	Mean	***Std. Dev.***	***Mean***	***Std. Dev.***	***Mean***	***Std. Dev.***
RF	0.7387	0.0266	0.8295	0.0258	0.7039	0.0221
SVM	0.7442	0.0217	0.8531	0.0143	0.5001	0.0127
DNN	0.8130	0.0700	0.8796	0.0485	0.7017	0.1163
LSTM	0.9082	0.0796	0.9410	0.0517	0.8514	0.1204

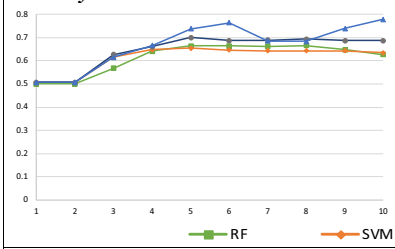
Log-specific observations:

- Accuracy and F-Score: The DL techniques show substantially better results than the classical ML techniques. LSTM outperforms DNN with varying intensity. RF and SVM perform very similarly.
- ROC AUC: The performance of the classifiers diverges substantially, and no class of classifiers outperforms the other. SVM delivers poor results and is considerably outperformed by RF, DNN, and LSTM. LSTM delivers by far the best score.
- Temporal stability: The classical ML techniques yield to more time stable predictors. In contrast, the metrics for the DL techniques fluctuate strongly over time. For some prediction time points, LSTM clearly exceeds the DNN.
- Number of instances and features: The log shows a relatively small number of instances, which decreases moderately over time. Meanwhile, the number of features substantially increases.

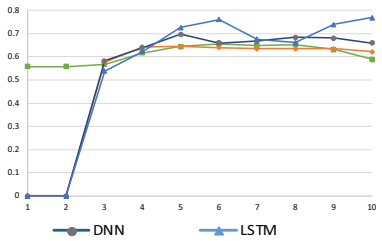
Table 5 continued

RL

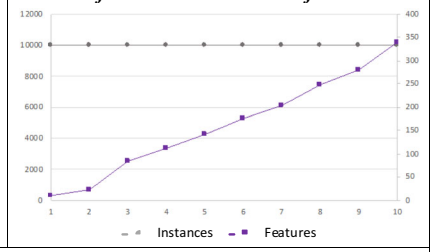
Accuracy



F-Score



Number of Instances vs. Number of Features



Log-specific evaluation metrics:

	Accuracy		*F-Score*		*ROC AUC*	
	Mean	***Std. Dev.***	***Mean***	***Std. Dev.***	***Mean***	***Std. Dev.***
RF	0.6141	0.0635	0.6123	0.0385	0.6599	0.0861
SVM	0.6142	0.0576	0.5026	0.2656	0.6084	0.0584
DNN	0.6445	0.0755	0.5269	0.2795	0.6427	0.0782
LSTM	0.6682	0.0932	0.5492	0.2824	0.6664	0.0956

Log-specific observations:

- Accuracy and F-Score: The DL techniques show a slightly better performance, except for the first two prediction time points. RF and SVM perform quite similarly, except for the first two prediction time points. Only RF is able to perform correct predictions after the first and second activity. Especially at prediction time point five, six, and ten, LSTM can show its advantages over DNN
- ROC AUC: All classifiers perform rather similar and no clear outperformance is notable. LSTM delivers the best result closely followed by RF and DNN. SVM falls a little behind
- Temporal stability: SVM and DNN are slightly more stable in time.
- Number of instances and features: The number of instances stays the same over all prediction time points. Thus, no instances ended prematurely. The number of features increases strongly, but the maximum is still relatively low.

generated at runtime) cause substantial outperformance. Moreover, we inferred that DL techniques perform more stably in case of imbalanced target variables, especially for logs with a high event-to-activity ratio (i.e., many loops in the control flow). Due to the purposive sampling of logs and techniques, these propositions also hold for logs outside our study.

6.2 Implications

By inferring propositions about which log properties facilitate the use of DL for outcome-oriented predictive process monitoring, our work contributes to the knowledge on process mining in general and on predictive process monitoring in particular. Our analysis showed a general outperformance of DL over classical ML techniques, which is particularly high if certain log properties are present. We specifically found that the outperformance of DL is not rooted in the values of individual log properties, but in the relationship between certain properties (e.g., variant-to-instance ratio). According to our findings, it is reasonable to conduct further research on DL and no longer on classical ML approaches to outcome-oriented predictive process monitoring. On the one hand, our results support the findings of studies that compared DL and classical ML techniques in other domains (Shickel et al. 2018; Menger et al. 2018). On the other hand, our results operationalize these findings with respect to outcome-oriented predictive process monitoring. Overall, our study is the first to systematically compare the performance of DL and ML techniques for outcome-oriented predictive process monitoring in a multi-log setting.

From a managerial perspective, our findings generally justify investments in the adoption and use of DL techniques for outcome-oriented predictive process monitoring in practice, specifically in the presence of certain log properties. However, we also observed log properties for which DL only slightly outperforms classical ML techniques. Related logs feature rather homogeneous instances and little information gain during execution. If organizations plan to use outcome-oriented predictive process monitoring only in such cases, it may be sensible to rely on classical ML techniques as the slight outperformance may not justify the higher investment required for DL techniques. On the one hand, the preprocessing effort is still higher for DL techniques (e.g., LSTM requires more complex feature encoding since the required feature vector is three-dimensional). On the other hand, novel frameworks such as Keras provide ready-to-use classifiers and reduce the complexity of the underlying libraries (e.g., TensorFlow), which makes the implementation almost as easy as for classical ML techniques. The higher hardware

requirements of DL can also be handled by using scalable graphics-processing-unit-enabled cloud computing infrastructures (e.g., Microsoft Azure or Amazon AWS). In the end, the decision whether to use DL for outcome-oriented predictive process monitoring (or for predictive process monitoring in general) depends not only on the log properties captured through our propositions but also on other organizational and economic factors. That is why we propose in Sect. 6.3 to develop decision models that account, among others, for the extent of outperformance and the costs associated with wrong predictions.

6.3 Limitations and Future Research

When comparing DL and classical ML techniques for outcome-oriented predictive process monitoring, we identified limitations that should be addressed in the future as well as directions in which our study should be extended.

- First, as we purposively sampled logs and techniques, we can claim that our propositions also hold for event logs outside our study. Nevertheless, as is typical, we only covered a specific sample of logs and techniques, not the respective populations. Hence, future research should analyze more logs and techniques (e.g., with more complex network topologies) in line with the introduced log properties to challenge and refine our propositions. Future research may also incorporate new trends in log structures (e.g., NoSQL) and log sources (e.g., IoT devices or logs from robotic process automation).
- Second, our propositions refer to log properties instead of process characteristics, because no literature-backed mapping is available. While such a mapping can be established for some control flow-related properties (e.g., the degree of standardization can be translated into the number of variants), there is a lack of knowledge regarding data-related log properties. Hence, a mapping between log properties and process characteristics should be developed in future research so as to empower process managers to make informed decisions about the adoption and use of outcome-oriented predictive process monitoring in practice.
- Third, since we have observed a varying outperformance of DL for different log properties, the question arises how the business value of DL techniques can be assessed for specific organizational settings. This is important as, in essence, the adoption and use of DL techniques is an organization- and process-specific decision. Thus, future research should develop decision models that not only account for our log properties, but also for the investment and training effort related to such techniques as well as for costs of wrong predictions.
- Finally, it would be interesting to investigate whether our propositions also hold for other prediction tasks (e.g., the prediction of the next action(s) or performance-related predictions) and for other types of datasets (e.g., sequential data not originating from process event logs). Our propositions could serve as a starting point for such efforts. The results for different prediction tasks may eventually be aggregated through a meta-study including propositions for all prediction tasks.

Acknowledgements Open Access funding provided by Projekt DEAL.

References

Augusto A, Conforti R, Dumas M, La Rosa M, Maggi FM, Marrella A, Mecella M, Soo A (2019) Automated discovery of process models from event logs: review and benchmark. IEEE Trans Knowl Data Eng 31(4):686–705

Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neur Netw 5(2):157–166

Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13(1):281–305

Bishop CM (2010) Pattern recognition and machine learning. Springer, New York

Bose JCB, van der Aalst WMP (2011) Analysis of patient treatment procedures: the BPI Challenge case study. In: BPM Workshops 2011 Proceedings. Clermont-Ferrand

Breiman L (2001) Random forests. Mach Learn 45(1):5–32

Breuker D, Matzner M, Delfmann P, Becker J (2016) Comprehensible predictive models for business processes. Manag Inf Syst Q 40(4):1009–1034

Castellanos M, Salazar N, Casati F, Dayal U, Shan M-C (2005) Predictive business operations management. In: DNIS 2005 Proceedings. Aizu-Wakamatsu, pp 1–14

Cardoso J, Mendling J, Neumann G, Reijers HA (2006) A discourse on complexity of process models. In: Eder J, Dustdar S (eds) BPM workshops 2006 proceedings, Vienna, pp 117–128

Ceci M, Lanotte PF, Fumarola F, Cavallo DP, Malerba D (2014) Completion time and next activity prediction of processes using

sequential pattern mining. In: Džeroski S, Panov P, Kocev D, Todorovski L (eds) DS 2014 proceedings, Bled, pp 49–61
Conforti R, Leoni M de, La Rosa M, van der Aalst WMP (2013) Supporting risk-informed decisions during business process execution. In: CAiSE 2013 Proceedings. Valencia, pp 116–132
Conforti R, Fink S, Manderscheid J, Röglinger M (2016) PRISM: a predictive risk monitoring approach for business processes. In: La Rosa M, Loos P, Pastor O (eds) BPM 2016 proceedings, Rio de Janeiro, pp 383–400
Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
Covington P, Adams J, Sargin E (2016) Deep neural networks for youtube recommendations. In: ACM RECSYS 2016 proceedings, Boston
Di Francescomarino C, Dumas M, Federici M, Ghidini C, Maggi FM, Rizzi W (2016) Predictive business process monitoring framework with hyperparameter optimization. In: Nurcan S, Soffer P, Bajec M, Eder J (eds) CAiSE 2016 proceedings, Ljubljana, pp 361–376
Di Francescomarino C, Ghidini C, Maggi FM, Milani F (2018) Predictive process monitoring methods: which one suits me best? In: Weske M, Montali M, Weber I, vom Brocke J (eds) BPM 2018 proceedings, Sydney, pp 462–479
Dumas M, La Rosa M, Mendling J, Reijers HA (2018) Fundamentals of business process management. Springer, Heidelberg
Evermann J, Rehse J-R, Fettke P (2016) A deep learning approach for predicting process behaviour at runtime. In: Dumas M, Fantinato M (eds) PARISE 2016 proceedings, Rio de Janeiro
Evermann J, Rehse J-R, Fettke P (2017a) Predicting process behaviour using deep learning. Decis Support Syst 100:129–140
Evermann J, Rehse J-R, Fettke P (2017b) XES TensorFlow: process prediction using the tensorflow deep-learning framework. In: Franch X, Ralyté J, Matulevičius R, Salinesi C, Wieringa R (eds) CEUR workshop 2017 proceedings 2017, Essen
Fushiki T (2011) Estimation of prediction error by using K-fold cross-validation. Stat Comput 21(2):137–146
Gartner Inc. (2018) Gartner identifies five emerging technology trends that will blur the lines between human and machine. https://www.gartner.com/en/newsroom/press-releases/2018-08-20-gartner-identifies-five-emerging-technology-trends-that-will-blur-the-lines-between-human-and-machine. Accessed 19 Nov 2018
Gers FA, Schmidhuber JA, Cummins FA (2000) Learning to forget: continual prediction with LSTM. Neural Comput 12(10):2451–2471
Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge
Grigori D, Casati F, Castellanos M, Dayal U, Sayal M, Shan M-C (2004) Business process intelligence. Comput Ind 53(3):321–343
Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A, Ullah Khan S (2015) The rise of "big data" on cloud computing: review and open research issues. Inf Syst 47:98–115
Haykin SS (2009) Neural networks and learning machines: A comprehensive foundation. Pearson, New York
Hinkka M, Lehto T, Heljanko K, Jung A (2019) Classifying process instances using recurrent neural networks. In: Daniel F, Sheng Q, Motahari H (eds) BPM workshops proceedings, Vienna, pp 313–324
Hosmer DW, Lemeshow S, Sturdivant RX (2013) Applied logistic regression. Wiley, Hoboken
Kang B, Kim D, Kang S-H (2012) Real-time business process monitoring method for prediction of abnormal termination using KNNI-based LOF prediction. Expert Syst Appl 39(5):6061–6068
Kratsch W, Manderscheid J, Reißner D, Röglinger M (2017) Data-driven process prioritization in process networks. Decis Support Syst 100:27–40
Lakshmanan GT, Shamsi D, Doganata YN, Unuvar M, Khalaf R (2013) A Markov prediction model for data-driven semi-structured business processes. Knowl Inf Syst 42(1):97–126
Lee AS, Baskerville RL (2003) Generalizing generalizability in information systems research. Inf Syst Res 14(3):221–243
Leontjeva A, Conforti R, Di Francescomarino C, Dumas M, Maggi FM (2015) Complex symbolic sequence encodings for predictive monitoring of business processes. In: Motahari-Nezhad H, Recker J, Weidlich M (eds) BPM 2015 proceedings, Innsbruck, pp 297–313
Levy D (2014) Production Analysis with process mining technology. Dataset. https://doi.org/10.4121/uuid:68726926-5ac5-4fab-b873-ee76ea412399
Lund S, Manyika J, Nyquist S, Mendonca L, Ramaswamy S (2013) Game changers: five opportunities for US growth and renewal. https://www.mckinsey.com/~/media/McKinsey/Featured%20Insights/Americas/US%20game%20changers/MGI_US_game_changers_Executive_Summary_July_2013.ashx. Accessed 19 Nov 2019
Ly LT, Maggi FM, Montali M, Rinderle-Ma S, van der Aalst Wil MP (2015) Compliance monitoring in business processes: functionalities, application, and tool-support. Inf Syst 54:209–234
Maggi FM, Di Francescomarino C, Dumas M, Ghidini C (2014) Predictive monitoring of business processes. In: Jarke M et al (eds) CAiSe 2014 proceedings, Thessaloniki, pp 457–472
Mannhardt F, de Leoni M, Reijers HA, van der Aalst WMP (2016) Balanced multi-perspective checking of process conformance. Computing 98(4):407–437
Marquez-Chamorro AE, Resinas M, Ruiz-Cortes A (2018) Predictive monitoring of business processes: a survey. IEEE Trans Services Comput 11(6):962–977
Mehdiyev N, Evermann J, Fettke P (2017) A multi-stage deep learning approach for business process event prediction. In: IEEE 19th CBI proceedings, Thessaloniki, pp 119–128
Mehdiyev N, Evermann J, Fettke P (2018) A novel business process prediction model using a deep learning method. Bus Inf Syst Eng 62(2):143–157
Menger V, Scheepers F, Spruit M (2018) Comparing deep learning and classical machine learning approaches for predicting inpatient violence incidents from clinical text. Appl Sci (Switzerland) 8(6):981
Metzger A, Leitner P, Ivanovic D, Schmieders E, Franklin R, Carro M, Dustdar S, Pohl K (2015) Comparing and combining predictive business process monitoring techniques. IEEE Trans Syst Man Cybern Syst 45(2):276–290
Müller O, Junglas I, vom Brocke J, Debortoli S (2016) Utilizing big data analytics for information systems research: challenges, promises and guidelines. Eur J Inf Syst 25(4):289–302
Pasquadibisceglie V, Appice A, Castellano G, Malerba D (2019) Using convolutional neural networks for predictive process analytics. In: IEEE ICPM 2019 proceedings, Aachen, pp 129–136
Polato M, Sperduti A, Burattin A, Leoni M de (2014) Data-aware remaining time prediction of business process instances. In: IEEE IJCNN 2014 proceedings, Beijing, pp 816–823
Polato M, Sperduti A, Burattin A, de Leoni M (2018) Time and activity sequence prediction of business process instances. Computing 100(9):1005–1031
Rogge-Solti A, Weske M (2015) Prediction of business process durations using non-Markovian stochastic Petri nets. Inf Syst 54:1–14
Russell N, ter Hofstede AHM, Edmond D, van der Aalst WMP (2005) Workflow data patterns: Identification, representation and tool support. In: Delcambre L, Kop C, Mayr HC, Mylopoulos J, Pastor O (eds) ER 2005 proceedings, Klagenfurt, pp 353–368

Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117

Schönig S, Jasinski R, Ackermann L, Jablonski S (2018) Deep learning process prediction with discrete and continuous data features. In: Damiani E, Spanoudakis G (eds) ENASE 2018 proceedings, Funchal, pp 314–319

Shickel B, Tighe PJ, Bihorac A, Rashidi P (2018) Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. IEEE J Biomed Health Inf 22(5):1589–1604

Shmueli G, Koppius OR (2011) Predictive analytics in information systems research. Manag Inf Syst Q 35(3):553–572

Sindhgatta R, Ghose A, Dam HK (2016) Context-aware analysis of past process executions to aid resource allocation decisions. In: Nurcan S, Soffer P, Bajec M, Eder J (eds) CAiSE 2016 proceedings, Ljubljana, pp 575–589

Sokolova M, Lapalme G (2009) A systematic analysis of performance measures for classification tasks. Inf Process Manag 45(4):427–437. https://doi.org/10.1016/j.ipm.2009.03.002

Steeman W (2013) BPI Challenge 2013. Dataset. https://doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07

Tax N, Verenich I, La Rosa M, Dumas M (2017) Predictive business process monitoring with LSTM neural networks. In: Dubois E, Pohl K (eds) CAiSE 2017 proceedings, Essen, pp 477–492

Teinemaa I, Dumas M, La Rosa M, Maggi FM (2019) Outcome-oriented predictive process monitoring: review and benchmark. ACM Trans Knowl Discov Data 13(2):17

van der Aalst WMP (2010) Synthetic event logs: review example. Dataset. https://doi.org/10.4121/uuid:da6aafef-5a86-4769-acf3-04e8ae5ab4fe

van der Aalst WMP et al (2011a) Process mining manifesto. In: Daniel F, Barkaoui K, Dustdar S (eds) BPM international workshops 2011 proceedings, Clermont-Ferrand, pp 169–194

van der Aalst WMP (2013) Business process management: a comprehensive survey. ISRN Softw Eng 2013(1):1–37

van der Aalst WMP (2014) Data scientist: the engineer of the future. In: Mertins K, Bénaben F, Poler R, Bourrières J-P (eds) Enterprise interoperability VI. Interoperability for agility, resilience and plasticity of collaborations. Springer, Cham, pp 13–26

van der Aalst WMP, Schonenberg MH, Song M (2011) Time prediction based on process mining. Inf Syst 36(2):450–475

van Dongen BF (2011) BPI Challenge 2011. Dataset. https://doi.org/10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54

van Dongen BF, Crooy RA, van der Aalst WMP (2008) Cycle time prediction: when will this case finally be finished? In: Meersman R, Tari Z (eds) OTM 2008 proceedings, Monterrey, pp 319–336

vom Brocke J, Zelt S, Schmiedel T (2016) On the role of context in business process management. Int J Inf Manag 36(3):486–495

van Dongen BF, de Medeiros AKA, Verbeek HMW, Weijters, AJMM, van der Aalst WMP (2005) The ProM framework: a new era in process mining tool support. In: Ciardo G, Darondeau P (eds) ICATPN 2005 proceedings, Miami, pp 444–454

Weyand T, Kostrikov I, Philbin J (2016) PlaNet - photo geolocation with convolutional neural networks. In: Leibe B, Matas J, Sebe N, Welling M (eds) ECCV 2016 proceedings, Amsterdam, pp 37–55

Witten IH, Frank E, Hall MA, Pal CJ (2017) Data mining: Practical machine learning tools and techniques. Morgan Kaufmann/Elsevier, Amsterdam

Yin RK (1994) Case study research: design and methods, 2nd edn. Sage, Thousand Oaks

Zhang P (1993) Model selection via multifold cross validation. Ann Stat 21(1):299–313