From Time Series to Process Model Forecasting

Johannes De Smedt[1], Anton Yeshchenko[2], Artem Polyvyanyy[3], Jochen De Weerdt[1], and Jan Mendling[2]

[1] KU Leuven, Leuven, Belgium
{johannes.desmedt;jochen.deweerdt}@kuleuven.be
WU Vienna, Vienna, Austria
{anton.yeshchenko;jan.mendling}@wu.ac.at
[2] University of Melbourne, Melbourne, Australia
artem.polyvyanyy@unimelb.edu.au

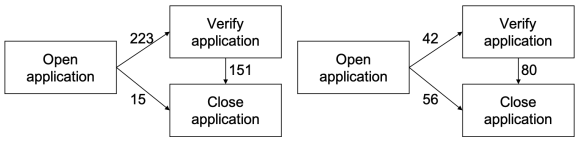
Abstract. The surge in event-based data recorded during the execution of business processes is ever-growing and has spurred an array of process analytics techniques to support and improve information systems. A major strand of process analytics encompasses forecasting the process's future development, mostly focusing on next-step, remaining time, or goal-oriented predictions. The granularity of such approaches lies with the events in the process. This work approaches process forecasting at the level of the entire process model. This allows process analysts to act on predicted global and longer-term changes such as impending drifts in the process model rather than fine-granular ones such as next-step prediction. To this purpose, event data is captured at various intervals and aggregated in directly-follows graphs. The relation of each activity pair in the graph is monitored over these intervals, and their future values forecasted using standard time-series techniques. Experiments confirm that this approach is already well-capable of informing process analysts about the future status of the process.

Keywords: Process model forecasting, predictive process modelling, process mining, time series analysis

1 Introduction

The growth in the use of information systems has fuelled a wide range of data analysis techniques which intend to describe and improve their inner workings. Process mining [1] is one of the strongest-growing fields in information systems analysis and encompasses the wide range of analysis which can be performed on event data generated by these systems including the visualisation, conformance checking, and improvement of process models generating these events. More recently, predictive process analysis techniques, often referred to as predictive process monitoring, have surfaced to support the prediction of the next event/activity in the process, the remaining time, or other goal-oriented outcomes [5]. Various predictive techniques exist, which make use of various analytics architectures including neural networks [20], stochastic Petri nets [17], or general classification techniques [21].

The approach presented in this paper entails a paradigm shift from these typical predictive applications as the focus is not on the level of single outcomes at the level of the trace and/or event but on the model. These predictive techniques construct a predictive model to infer the future development of a trace, while the proposed forecasting approach infers a model which can generate future traces itself. This is achieved by an aggregation at the model level both at the input as well as the output of the learning process. This is done by using directly-follows (DF) relations over all activity pairs as a summary of the process model at various time steps, which are further extrapolated into the future using time series techniques. Consider the example in Figure 1 where both an actual directly-follows graph (DFG) is presented constructed after less than half of the events has occurred in the system and a DFG containing the predicted relations towards a later phase of the process based on the same data at hand. Being able to construct such a predictions allows stakeholders to make estimates regarding how the overall system will evolve and allows to answer questions such as: How many more applications will be received? Will the backlog of verifications be reduced? Will all applications be closed? Will the ratio of immediately closeable applications stay the same? These offer a longer-term and more global view compared to the next- or near-next-step predictions, although can be enriched with remaining time predictions.



(a) DFG - actual (b) DFG - predicted

Fig. 1: Directly-follows graphs of the beginning of the process, as well as a forecast closer towards the end of the process.

An initial attempt to set up process model forecasting was presented in [4], however, the level of granularity of the forecasted horizons is coarser compared to the higher-frequency intervals used for time series underpinning the proposed approach. Where process mining focuses on learning the as-is model to inform the to-be model and suggest potential repairs and improvements, process model forecasting allows to already grasp the future outcomes of the current as-is process which allows to shortcut potentially wrong outcomes [15].

Next to the forecasting aspect, we also present how to visualize the forecasted models and show how they exist of adaptations to the current as-is model with a level of uncertainty through change graphs [9].

This paper makes three contributions:

1. Proposes the first process model forecasting technique, which fits the the broader area of predictive process monitoring;

2. Evaluates the quality of the forecasted process models using standard conformance checking techniques from process mining;
3. Confirms that the achieved quality of forecasts is useful for practical applications.

To this purpose, a variety of time series analysis techniques are applied to 3 real-life event logs with two event log aggregations that establish suitable time series. They predict the level of directly-follows occurrences for activity pairs over time. A variety of considerations regarding the use of time series analysis for event logs are discussed, as well as the suitability of other predictive algorithms. Results show that simple forecasting techniques often already perform adequately without extensive parameter tuning. More intricate models often fail to report consistent performance.

This paper is structured as follows...

2 Preliminaries

An event log L contains the recording of traces $\sigma \in L$ produced by an information system during its execution and contains a sequence of events. Events in these traces are part of the power set over the alphabet of activities Σ which exist in the information system $\sigma = \langle e_1, ..., e_{|\sigma|} \rangle \subseteq \Sigma^*$. Directly follows relations between activities in an event log can be expressed as a counting function over activity pairs $>_L$: $\Sigma \times \Sigma \to \mathbb{N}$ with $>_L (a_1, a_2) = |\{e_n = a_1, e_{n+1} = a_2, \forall e_i \in L\}|$. Directly follows relations can be calculated on traces and subtraces in a similar fashion. A Directly Follows Graph (DFG) of the process then exists as the weighted directed graph with the activities as nodes and their DF relations as weights $DFG = (\Sigma, >_L)$.

In order to obtain predictions regarding the evolution of the DFG we construct DFGs for subsets of the log. Many aggregations and bucketing techniques exist for next-step and goal-oriented outcome prediction [20, 21], e.g., predictions at a point in the process rely on prefixes of a certain length, or particular state aggregations [2]. In the proposed forecasting approach, however, not cross-sectional but time series data will be used. Hence, the evolution of the DFGs will be monitored over intervals of the log where multiple aggregations are possible:

- **Equitemporal aggregation:** each sublog $L_s \in L$ contains a part of the event log of equal time duration. This can lead to sparsely populated sublogs when the events' occurrences are not uniformly spread over time, however, is easy to apply (on new traces).
- **Equisized aggregation:** each sublog $L_s \in L$ contains a part of the event log where an equal amount of events (and hence DFs) occurred. This leads to well-populated sublogs, however, might be harder to apply when new data does not contain sufficient new DF occurrences.

Time series for the DFs $T_{a_1,a_2} = \langle >_{L_1} (a_1, a_2), \ldots, >_{L_s} (a_1, a_2) \rangle, \forall a_1, a_2 \in \Sigma \times \Sigma$ can be obtained for all activity pairs where $\bigcup_{L_1}^{L_s} = L$ by applying the aforementioned aggregations to obtain the sublogs. Tables 1 and 2 provide an example of both.

Case ID	Activity	Timestamp
1	A1	11:30
1	A2	11:45
1	A1	12:10
1	A2	12:15
2	A1	11:40
2	A1	11:55
3	A1	12:20
3	A2	12:40
3	A2	12:45

Table 1: Example event log with 3 traces and 2 activities.

DF	Equitemporal	Equisized
$<_{Ls}(A1, A1)$	(0,1,0)	(1,0,0)
$<_{Ls}(A1, A2)$	(1,1,1)	(1,1,1)
$<_{Ls}(A2, A1)$	(0,1,0)	(0,1,0)
$<_{Ls}(A2, A2)$	(0,0,1)	(0,0,1)

Table 2: An example of using an interval of 3 used for equitemporal aggregation (75 minutes in 3 intervals of 25 minutes) and equisized intervals of size 2 (6 DFs over 3 intervals)).

3 Methodology

This section outlines the forecasting techniques that will be used, as well as their connection with time series extracted from event logs.

3.1 Time series techniques

To model the time series of DFGs, various algorithms are used. In time series modelling, the main objective is to obtain a forecast or prediction $\hat{y}_{T+h|T}$ for a horizon $h \in \mathbb{N}$ based on previous T values in the series $(y_1, ..., y_T)$ [8]. A wide array of time series techniques exist, ranging from simple models such as naive forecasts over to more advanced approaches such as exponential smoothing and auto-regressive models. Many also exist in a seasonal variant due to their application in contexts such as sales forecasting. Below, the most-widely used techniques are formalised.

The naive forecast simply uses the last value of the time series T as its prediction:

$$\hat{y}_{T+h|T} = y_T$$

An alternative naive forecast uses the average value of the time series T as its prediction:

$$\hat{y}_{T+h|T} = \frac{1}{T}\Sigma_i^T y_i$$

A Simple Exponential Smoothing (SES) model uses a weighted average of past values where their importance exponentially decays as they are further into the past:

$$\hat{y}_{T+h|T} = \alpha y_T + (1-\alpha)\hat{y}_{T|T-1}$$

with $\alpha \in [0, 1]$ a smoothing parameter where lower values of α allow for focusing on the more recent values more. Holt's models introduce a trend in the forecast,

meaning the forecast is not flat:

$$\hat{y}_{T+h|T} = l_t + hb_t$$

with $l_t = \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1})$ modelling the overall level of the time series and $b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$ modelling the trend over the series where α is as before and β the smoothing parameter for the trend. Exponential smoothing models often perform very well despite their simple setup.

AutoRegressive Integrating Moving Average (ARIMA) models are based on auto-correlations within time series. They combine auto-regressions with a moving average over error terms.

An AutoRegressive (AR) model of order p uses the past p values in the time series and applies a regression over them. It can be written as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

where all ϕ_i, $i \in [1, p]$ can assign different weights to different time lags.

A Moving Average (MA) model of order q can be written as:

$$y_t = c + \epsilon_t + \phi_1 \epsilon_{t-1} + \cdots + \phi_q \epsilon_{t-q}$$

with $\phi > 0$ a smoothing parameter, and ϵ_t again a white noise series, hence the model creates a moving average of the past forecast errors. Given the necessity of using a white noise series for AR and MA models, data is often differenced to obtain such series.

ARIMA models then combine both AR and MA models where the integration is taking place after modelling as these models are fitted over differenced time series. An ARIMA(p, d, q) model can be written as:

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_t$$

with y'_t a differenced series of order d. Forecasting is possible by introducing $T+h$ to the equation and iteratively obtaining results by starting off with $T+1$. ARIMA models are considered to be one of the strongest time series modelling techniques.

An extension to ARIMA which is widely used in econometrics exists in (Generalized) AutoRegressive Conditional Heteroskedasticity ((G)ARCH) models [6]. They resolve the assumption that the variance of the error term has to be equal over time, but rather model this variance as a function of the previous error term. For AR-models, this leads to the use of ARCH-models, while for ARMA models GARCH-models are used as follows.

An ARCH(q) model captures the change in variance over time as follows:

$$\sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots + \alpha_q y_{t-q}^2$$

is the variance over time with $\alpha_0 > 0$ and $\alpha_i \geq 0$, $i \in [1, q]$ smoothing parameters, and $y_t = \sigma_t \epsilon_t$ where $\epsilon_t \overset{i.i.d}{\sim} (\mu = 0, \sigma^2 = 1)$. This can be used to capture

that the variance gradually increases over time, or has short bursts of increased variance.

A GARCH(p,q) model combines both the past values of observations as well as the past values of variance:

$$\sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots + \alpha_q y_{t-q}^2 + \cdots + \beta_1 \sigma_{t-1}^2 + \cdots + \beta_p \sigma_{t-p}^2$$

with $\alpha_0 > 0$ and $\alpha_i, \beta_i \geq 0$, $i > 0$.

(G)ARCH models often outperform ARIMA models in contexts such as the prediction of financial indicators of which the variance often changes over time [6].

3.2 Forecasting Directly Follows series

A trade-off exists between approaching DFGs as a multivariate collection of DF time series, or treating each DF separately. The aforementioned time series techniques all use univariate data in contrast with, e.g., Vector AutoRegression (VAR) models, or machine learning-based methods such as neural networks or random forest regressors. Despite their simple setup, it is debated whether traditional statistical approaches are necessarily outperformed by machine learning methods. [13] found that this is not the case on a large number of datasets and note that the machine learning algorithms require significantly more computational power, a result that was later reaffirmed although it is noted that hybrid solutions are effective [14]. Especially for longer horizons traditional time series approaches still outperform machine learning-based models. Given the potentially high number of DF pairs in a process' DFG, the proposed approach uses a time series algorithm for each DF series separately. VAR models would require a high number of intervals (at least as many as there are DFs times the lag coefficient) to be able to estimate all parameters of such a high number of time series despite their potentially strong performance [22]. Machine learning models could potentially leverage interrelations between the different DFs but again would require long training times to account for dimensionality issues due to the potentially high number of DFs. Therefore, in this paper, traditional time series approaches are chosen and applied to the univariate DF time series where these have at least 1 observation per sublog/time step present.

4 Experimental evaluation

In this section, an experimental evaluation over three real-life event logs is reported.

4.1 Re-sampling and test setup

In order to obtain training data, time series are obtained by specifying a number of intervals (i.e. time steps in the DF time series) using either equitemporal or

equisize aggregation a described in Section 2. Time series algorithms are parametric and sensitive to sample size requirements [7]. Depending on the number of parameters a model uses, a minimum size of at least 50 steps is not uncommon, although typically model performance should be monitored at a varying number of steps. In the experimental evaluation, the event logs are divided into 100 time steps with a varying share of training and test steps: $h = 25$ meaning all test sets contain 25 intervals, and the training sets are varied from $ts = 25$ to $ts = 75$ steps, meaning the forecasts progressively target the prediction of steps 25-50 (the second quarter of intervals) over to 75-100 (the last quarter of intervals). This allows to both inspect the difference in results when only few data points are used, and whether there is a difference forecasting data points in the middle or towards the end of the available event data. Resampling is applied based on a 10-fold cross-validation constructed following a rolling window approach for all horizon values $h \in [1, 25]$ where a recursive strategy is used to iteratively obtain $\hat{y}_{t+h|T_{t+h-1}}$ with $(y_1, \ldots, y_T, \ldots, \hat{y}_{t+h-1})$ [23]. 10 training sets are hence constructed for each training set length ts and exist from $(y_1, \ldots, y_{T-h-f})$ and the test sets from $(y_{T-h-f+1}, \ldots, y_{T-f})$ with $f \in [0, 9]$ the fold index [3]. While direct strategies with a separate model for every value of h can be used as well and avoid the accumulation of error, they do not take into account statistical dependencies for subsequent predictions.

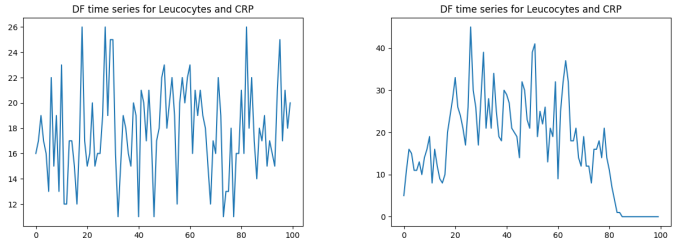
Three widely-used event logs are used: the 2012 BPI challenge log[3], the Sepsis cases event log[4], and the Road Traffic Fine Management Process log[5] (RTFMP) event log. Each of these logs has a diverse set of characteristics in terms of case and activity volume, as well as average trace length as can be seen in Table 3.

Event log	# cases	# activities	Average trace length
BPI 12	13,087	36	20.020
Sepsis	1,050	16	14.490
RTFMP	150,370	11	3.734

Table 3: Overview of the characteristics of the event logs used in the experimental evaluation.

An example of applying the equisize or equitemporal aggregation to the Sepsis event log with 100 intervals results in the DF time series of Figure 2 where the DF occurrences of the most frequently occurring activity pair is included. For the equisized aggregation the number of DFs is indeed relatively stable over the log's timeline where for the equitemporal aggregation a noticeable decline of DF pairs is visible towards the end of the series. This phenomenon is typical in event logs, as processes typically have particular endpoint activities, but can also

[3]https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
[4]https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460
[5]https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5



(a) Most common DF - equisize (b) Most common DF - equitemp

Fig. 2: Sepsis

be due to the unequal distribution of events over the event log's time line. If the level of occurrences of the DF pair is low and close to 0, the series might be too unsuitable for analysis with white noise series analysis techniques that assume stationarity. Ideally, every time series is tested using a stationarity test such as the Dickey-Fuller unit root test [12] and an appropriate lag order is established for differencing. Furthermore for each algorithm, especially ARIMA-based models, (partial) auto-correlation could establish the ideal p and q parameters. However, for the sake of simplicity and to avoid tedious solutions where each activity pair has to have different parameters, various values are used for p, d, and q and applied to all DF pairs where only the best-performing are reported below for comparison with the other time series techniques.

Given that we want to evaluate the capability of the approach to accurately predict the evolution of the process model, the combination of all DF predictions to obtain a global DFG prediction is considered. The following two criteria are used:

- **Cosine distance:** measures the distance between two vectors and is often used to compare graph distance. This metric is used to compare the DFG's edge weight matrices between the actual and predicted number of DF relations.
- **Entropic relevance:** a measure for stochastic conformance checking computed as the average number of bits required to compress each of the log's traces based on the structure and information about relative likelihoods provided by the model [16].

These criteria balance a predictive and structural evaluation of the algorithms and report on both the numeric performance common in a forecasting setting as well as their appropriateness in terms of reproducing a structurally usable process model which allows for the observed process behavior. In both cases a lower score is better. The entropic relevance also allows to compare the adequacy the forecasted DFGs with the actual DFGs.

4.2 Results

All pre-processing was done in Python with a combination of *pm4py*[6] and the *statsmodels* package [18]. The code is available

The results are displayed in Figures 3 to 8. - STILL NEED TO REDUCE THESE FIGURES. ANY INPUT WOULD BE HELPFUL (definitely no need to keep all ARIMAs I think).

NOTE THAT ALL FIGURES ARE AVAILABLE IN Results PMF 08-03.zip
Observations:

- Regardless of the aggregation type, only using 25-35 training points to forecast 25 test points creates very unstable results for all datasets and forecasting techniques both in terms of cosine distance and entropic relevance. Especially autoregressive models (both AR and ARIMA) from a higher order (2,4) suffer from very high cosine distances when few training points are used. This is potentially due to the inclusion of more distant observations in the prediction.
- Once a level of stability is present in the results, the difference between forecasting techniques is lower in terms of the cosine distance although GARCH and AR models still perform worse.
- In terms of entropic relevance, the actual DFGs do have better scores except for the BPI 12 log where GARCH even outperforms the actual DFG. In general, GARCH scores best in terms of entropic relevance while the other models perform similar to the naive model. ARIMA models perform worst overall.
- For the equitemporal aggregation this difference of the actual with the forecasts is less pronounced and the difference among the forecasts is minimal as well. The entropic relevance is also significantly higher for the equitemporal aggregation.

Implications:

- The entropic relevance learns us that the forecasting techniques are capable of obtaining similar results capturing as much of the process behavior as the actual DFGs generated by the process.
- The forecasting techniques used have little impact, meaning that even a naive forecast can produce good results, although GARCH models perform better, possibly due to their capturing a varying level of variance over time which seems especially suitable for DF relations which do not necessarily are generated by a white noise process. Note, however, that the confidence intervals have not yet been investigated.
- The results show that there is a need to have at least observed close to half of the process' events before reasonably stable predictions can be made.
- An equitemporal aggregation results in worse forecasts. This is potentially due to the inconsitent number of DF occurrences present, compared to the equisize aggregation which is stable and hence approaches a white noise process.

[6] https://pm4py.fit.fraunhofer.de

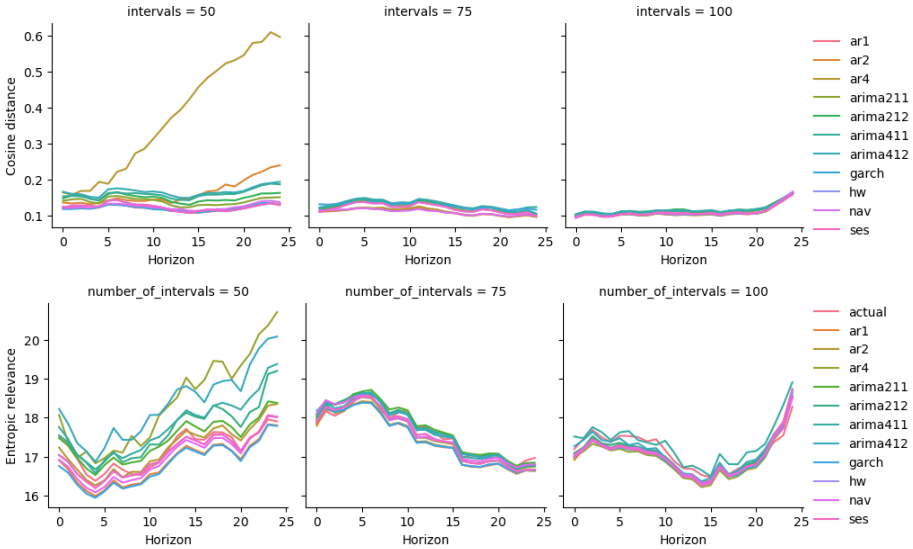


Fig. 3: Results for equi-size aggregation for the BPI12 event log.

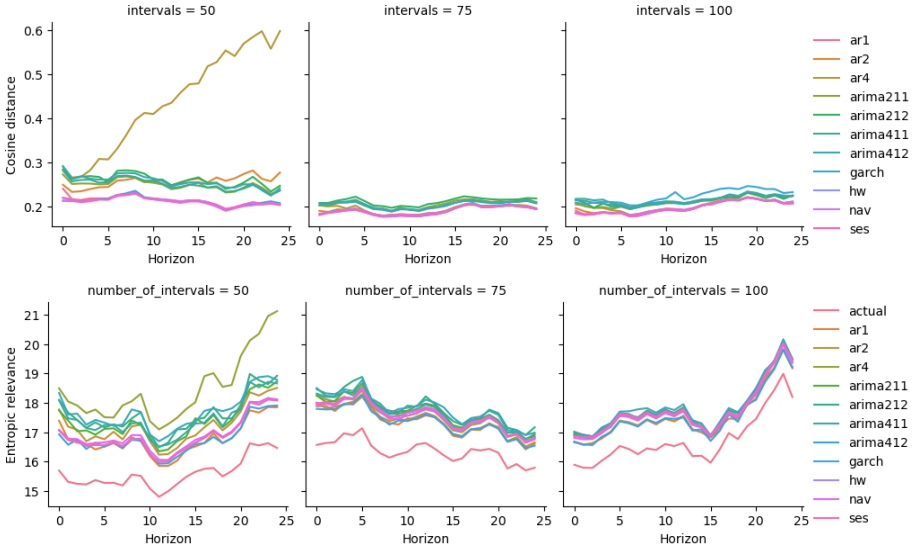


Fig. 4: Results for equi-size aggregation for the sepsis event log.

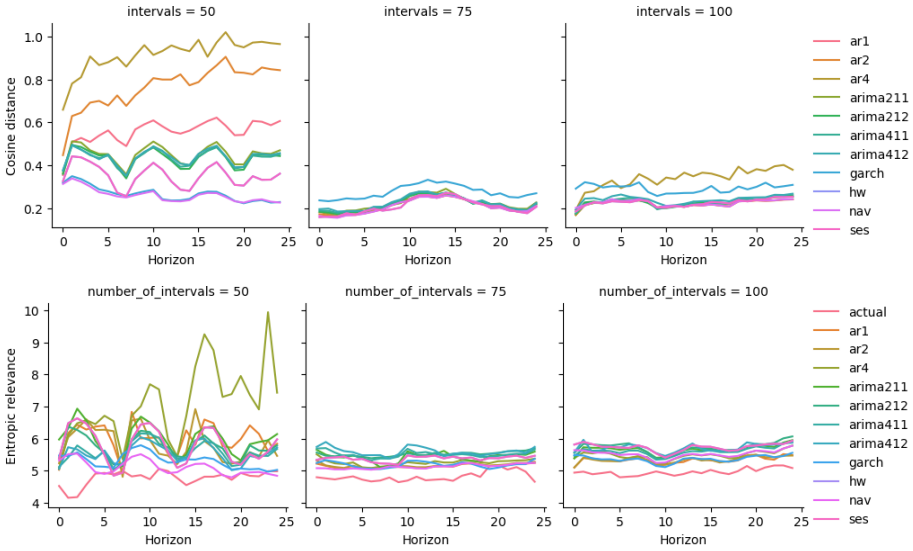


Fig. 5: Results for equi-size aggregation for the RTFMP event log.

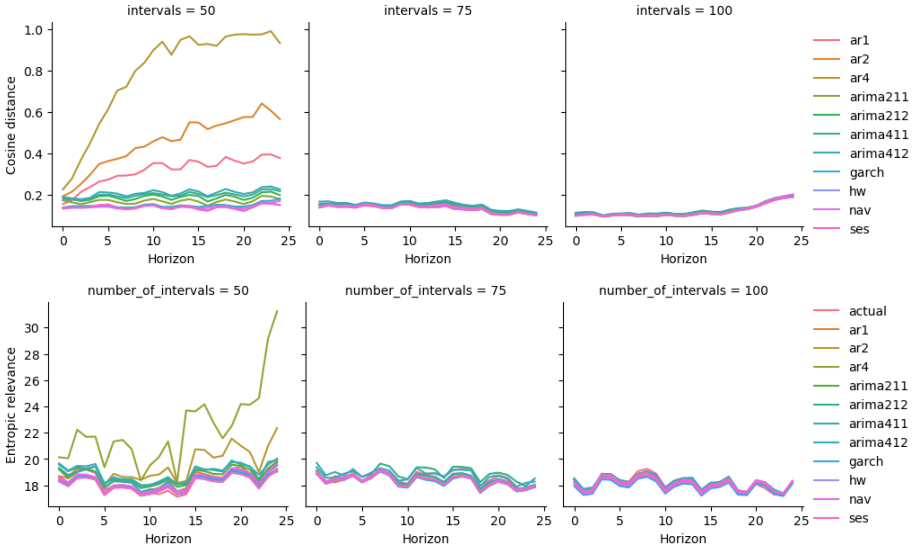


Fig. 6: Results for equi-temporal aggregation for the BPI12 event log.

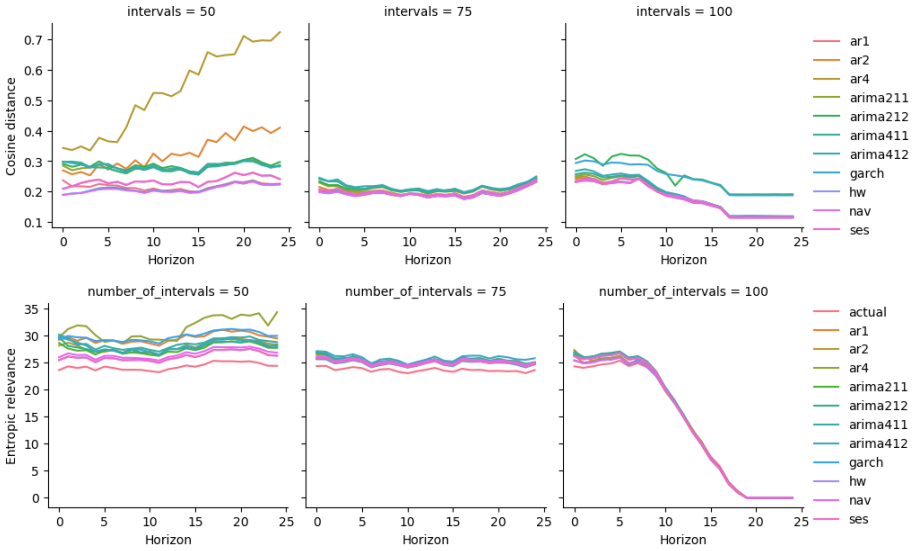


Fig. 7: Results for equi-temporal aggregation for the sepsis event log.

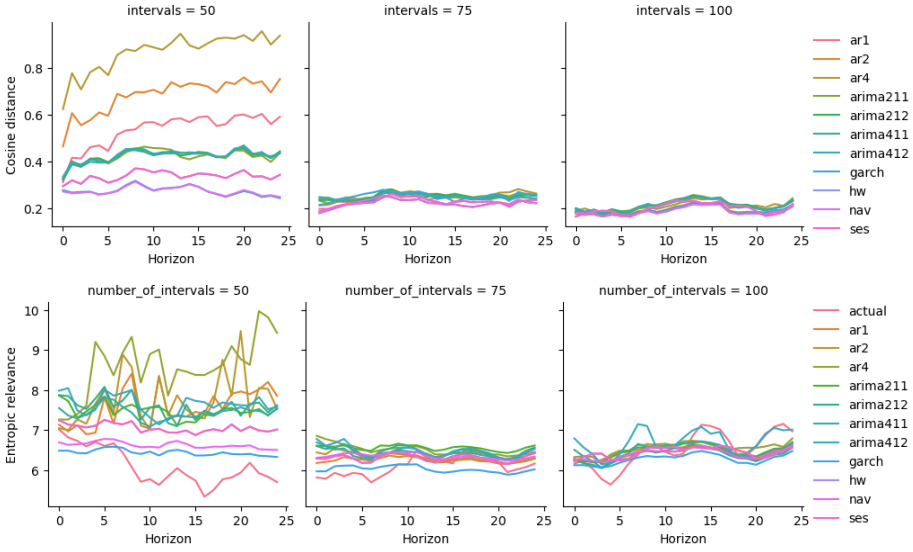


Fig. 8: Results for equi-temporal aggregation for the RTFMP event log.

5 Visualising Process Model Forecasts

In the Section 4 we evaluated forecasting results, ensuring the relevance of the predicted process models. To that end, gaining actual insights from such predicted data remains a difficult task for the analyst. This section sets off to present the design of a novel visualization system to aid analysts in exploration of the event logs.

We designed a Process Change Exploration (PCE) system to support the interpretation of the process model forecasts. In order to design the system we first established user tasks as a basis for the system design decisions. To derive the user tasks we focus on the requirements of process mining analysis with respect to process forecasting and visualization principles. The authors of [15] discuss the opportunities for process forecasting. They describe that the utility of process forecasting is an understanding of the incremental changes or adaptations that happen to the process model into the future. In designing an explorative visualization system, we also followed the "Visual Information-Seeking Mantra:" *overview first, zoom and filter, then details-on-demand* [19]. Thus, we expect the design of our system to assist in the following tasks:

R1. **Identify process adaptations:** The visualization system should assist the user in identifying the changes that happen in the process model of the future in respect to the past;

R2. **Allow for interactive exploration:** The user should be able to follow the visual information-seeking principles, including overview first, filtering, zooming, details-on-demand principles.;

The design of the PCE system is shown in Figure 9. It shows an interactive visualization system with several connected views. The view *(b)* shows the area graph for the number of activities executed per each period. The green area represents the actual data, and the grey region represents the predicted values. Users can brush one or more regions on this graph in order to filter the scope of the analysis *(b.1*, and *b.2)*. If a user brushes two regions then the earlier will be color-coded in red, and the later color-coded in green. The main DFG view *(a)* is showing the directly follows graph of the brushed region when none or one region is selected or the difference between two brushed regions when two regions are brushed. In case the difference is showed, the red transitions are used to show the decrease in executions, green to show the increase and grey in case of stable behavior. For coloring, we used ideas from [10]. Two additional views *(c.1)*, and *(c.2)* allow for the standard filtering [11] of activities and paths.

The process of designing and implementing the system started by designing several prototypes that undergone rounds of discussions to mature into the implemented visualization system. The system is implemented with D3.js JavaScript library and is available at [7].

[7] google.com

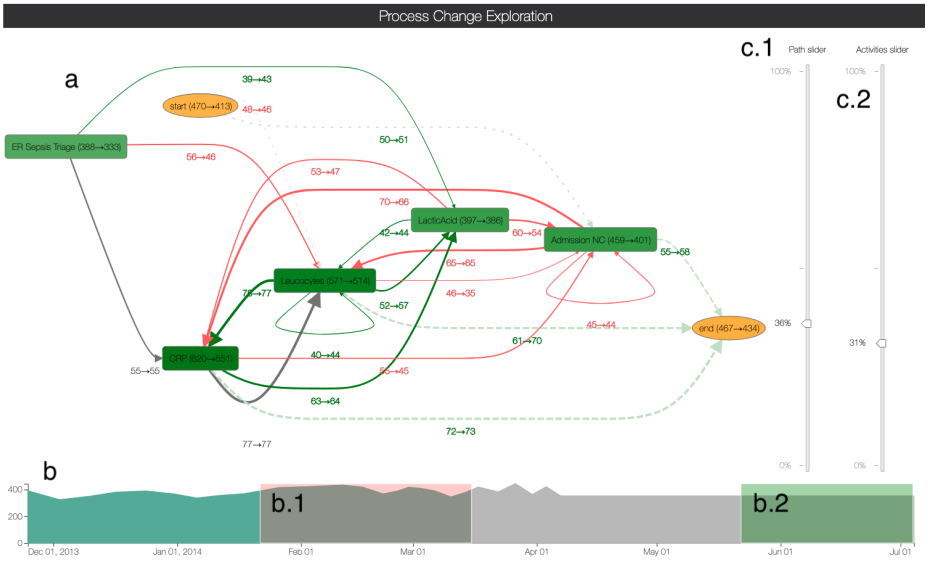


Fig. 9: Process Change Exploration (PCE) system. *(b)* is the activity timeline view, *(a)* main Directly-follows graph view, *(c.1), (c.2)* additional filtering views.

6 Conclusion

In this paper we investigated the potential of using time series forecasting for process model monitoring.

In future research we will cover more intricate forecasting techniques and compare with machine learning-based models. Furthermore, we will perform an extensive prediction confidence interval analysis.

References

1. van der Aalst, W.: Data Science in Action. In: Process Mining. Springer (2016)
2. van der Aalst, W.M.P., Rubin, V.A., Verbeek, H.M.W., van Dongen, B.F., Kindler, E., Günther, C.W.: Process mining: a two-step approach to balance between underfitting and overfitting. Softw. Syst. Model. 9(1), 87–111 (2010)
3. Bergmeir, C., Benítez, J.M.: On the use of cross-validation for time series predictor evaluation. Information Sciences 191, 192–213 (2012)
4. De Smedt, J., De Weerdt, J., Mori, J., Ochi, M.: Predictive process model monitoring using recurrent neural networks. arXiv preprint arXiv:2011.02819 (2020)
5. Francescomarino, C.D., Ghidini, C., Maggi, F.M., Milani, F.: Predictive process monitoring methods: Which one suits me best? In: BPM. Lecture Notes in Computer Science, vol. 11080, pp. 462–479. Springer (2018)
6. Francq, C., Zakoian, J.M.: GARCH models: structure, statistical inference and financial applications. John Wiley & Sons (2019)
7. Hanke, J.E., Reitsch, A.G., Wichern, D.W.: Business forecasting, vol. 9. Prentice Hall New Jersey (2001)

8. Hyndman, R.J., Athanasopoulos, G.: Forecasting: principles and practice. OTexts (2018)
9. Kabicher, S., Kriglstein, S., Rinderle-Ma, S.: Visual change tracking for business process models. In: ER. Lecture Notes in Computer Science, vol. 6998, pp. 504–513. Springer (2011)
10. Kriglstein, S., Rinderle-Ma, S.: Change visualizations in business processes - requirements analysis. In: GRAPP/IVAPP. pp. 584–593. SciTePress (2012)
11. Leemans, S., Poppe, E., Wynn, M.: Directly follows-based process mining: A tool. In: Proceedings of the ICPM Demo Track 2019 (CEUR Workshop Proceedings, Volume 2374). Vol. 2374. pp. 9–12. Sun SITE Central Europe (2019)
12. Leybourne, S.J., et al.: Testing for unit roots using forward and reverse dickey-fuller regressions. Oxford Bulletin of Economics and Statistics 57(4), 559–571 (1995)
13. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: Statistical and machine learning forecasting methods: Concerns and ways forward. PloS one 13(3), e0194889 (2018)
14. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The m4 competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting 36(1), 54–74 (2020)
15. Poll, R., Polyvyanyy, A., Rosemann, M., Röglinger, M., Rupprecht, L.: Process forecasting: Towards proactive business process management. In: BPM. Lecture Notes in Computer Science, vol. 11080, pp. 496–512. Springer (2018)
16. Polyvyanyy, A., Moffat, A., García-Bañuelos, L.: An entropic relevance measure for stochastic conformance checking in process mining. In: ICPM. pp. 97–104. IEEE (2020)
17. Rogge-Solti, A., Weske, M.: Prediction of Remaining Service Execution Time Using Stochastic Petri Nets with Arbitrary Firing Delays. In: ICSOC. Lecture Notes in Computer Science, vol. 8274, pp. 389–403. Springer (2013)
18. Seabold, S., Perktold, J.: Statsmodels: Econometric and statistical modeling with python. In: Proceedings of the 9th Python in Science Conference. vol. 57, p. 61. Austin, TX (2010)
19. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: VL. pp. 336–343. IEEE Computer Society (1996)
20. Tax, N., Verenich, I., Rosa, M.L., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: CAiSE. Lecture Notes in Computer Science, vol. 10253, pp. 477–492. Springer (2017)
21. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. ACM Trans. Knowl. Discov. Data 13(2), 17:1–17:57 (2019)
22. Thomakos, D.D., Guerard Jr, J.B.: Naive, arima, nonparametric, transfer function and var models: A comparison of forecasting performance. International Journal of Forecasting 20(1), 53–67 (2004)
23. Weigend, A.S.: Time series prediction: forecasting the future and understanding the past. Routledge (2018)