EasyChair Preprint

№ 5808

An Initial Study on Temporal Loss Functions for Remaining Time Models

Mike Riess

June 15, 2021

An initial study on temporal loss functions for remaining time models

Mike Riess

School of Business and Economics
Norwegian University of Life Sciences
Universitetstunet 3 1433 Ås
Mike.riess@nmbu.no

Abstract. Guided by two business goals (earliness and accuracy), this initial study investigate the performance of five different loss functions, across event-log data from four different domains (healthcare, public administration and IT services). Three different temporal losses are proposed for improvement of earliness-performance. The results show that MAE is either outperformed or tied with the temporal losses in terms of both earliness and accuracy. Based on the results from the experiments, the optimal weighting of the temporal penalty vary based on the characteristics of the event log. However, the proposed MAE_{MtD} loss proved to perform well in most cases, in terms of both accuracy and earliness.

Keywords: Event-log data · Predictive process monitoring · LSTM · Earliness · Remaining time models.

1 Introduction

Acting on predictions of process-related KPIs [7] like throughput time, before actual performance is influenced, is the playbook of modern predictive analytics. Using event-log data exclusively, this is one of the main goals of predictive process monitoring [1, 30]. Successful development of such a predictive monitoring system, lies in its fit with the goals of the organization wherein it is implemented. This again depend on the alignment between business understanding, model formulation, and method of evaluation [5, 24].

Much research have been done on remaining time prediction of ongoing cases [28, 8, 21, 31, 20, 3, 30], but solutions are seldom evaluated with respect to the assumed business goals of the organizations from which the event-logs originate. The works of [6] propose a hyper-parameter optimization framework with respect to different business goals, however, this approach does not alter the learning of the models themselves. Instead, this act as a hyper-parameter optimization (HPO) and model selection framework [9].

At the core of every supervised machine learning model is a loss function, which essentially determine which patterns in the data to learn from, and which ones to ignore [11]. Aligning the loss function with a given business goal [5] will

thus ensure that the model learn to produce the most valuable predictions for the organization.

At the time of writing, this aspect of model development have not yet been studied for remaining time models based on event-log data. In this study, 5 different loss functions are tested across 4 different real world event-logs and evaluated with respect to two slightly different business goals. The results show that temporal penalties can improve performance in both aspects.

1.1 Business goals

A remaining time model can be measured with respect to multiple types of errors [30], which largely depends on the context of the use-case. In the following, two different business goals are presented: Accuracy and Earliness.

These are both fundamental properties of the model fit, but previous literature discuss a *trade-off* between the two [31, 30]. The *importance* or *priority* of each dimension might also differ based on an organizations needs. This paper therefore treat each as a separate business goal, with lower or higher priority based on the business case (see section 3.1).

Goal A: Overall accuracy For remaining time predictions to be useful in *any* area, the overall accuracy of the model need to be at an acceptable level. The average accuracy of a model is thus the main objective across use-cases. In business processes where traces may have many events (like the Sepsis [19] case data), and remaining time is needed in the full life-cycle of the trace, overall accuracy might be more important than a *good early estimate*.

Goal B: Earliness In other cases, the timing of the accuracy is important for end users to act in time. This is especially important in scenarios where a predictive monitoring system is used to aid a *prescriptive* component. In healthcare and service industries, one such component is often a dynamic work shift planning system [16, 27, 32, 4]. Needless to say, the more time an organization has to reorganize its resources based on expected demand, the more likely it will be able to adapt in time. The earliness goal is thus to have the best possible estimate of the total case duration, as early as possible.

1.2 Research question

To help understand whether a *temporal loss function* can adapt a predictive process monitoring system improve on one or more of the goals above, this study will answer the following research question: *Which loss functions perform best with respect to each of the business goals?*

To answer this question, a set of loss functions are proposed as alternatives to the most commonly used loss functions in the literature on remaining time models [30, 14]. LSTM remaining time models [20] are trained and evaluated on four different real world event logs from different business domains.

The evaluation consist in comparing the resulting models across two different dimensions: Accuracy (goal A), Earliness (goal B). Further details on the evaluation procedure can be found in section 3.2.

1.3 Related work

The initial work in [28], demonstrated the performance of non-parametric regression for predicting the total throughput time (remaining time) of an ongoing trace, using event-log data. This motivated a series of studies into different approaches to predict remaining time and other aspects of event-log data, using supervised learning. In recent years, recurrent neural networks, and more specifically Long Short-Term Memory (LSTM) Neural networks [13] have proved to be superior to decision trees, transition systems etc. as first presented in [8], further improved in[21] and finally in [20]. The recent work in [22] study the various approaches to architecture and loss functions, but find that studies with modified loss functions, do this for the purpose of multi-task learning as in [3]. In addition, the authors of [22] found that the approach in [20] had the best performance across 11 real world event-logs. This is consistent with the result in [30]. In [23] the authors studied the effect of intra and inter-case features for predictive process monitoring, where proposed intra-case features have been adopted in [20] as well as in this study. The concept of earliness was first used in [10] for evaluation of classification-tasks, and have later also been used for evaluation of remaining time models [30]. Some studies use accuracy alone to evaluate remaining time models [20, 22], which to some degree is problematic, as discussed with the examples in section 1.1.

2 Key concepts

In the following, some of the most important concepts used in this paper will be described briefly.

2.1 Predictive Process Monitoring

Predictive process monitoring, is as mentioned in [30]: *"multi-disciplinary area that draws concepts from process mining on one side, and machine learning on the other"*. Process mining [1] is a sub-field of Business Process Management (BPM)[7] and is focused on the analysis of (often business) processes through event data stored from management information systems, while the process is running. The techniques in process mining mainly span from Process Discovery, Conformance checking, Process reenginering, and Operational support [2].

Predictive Process Monitoring [26] mainly relate to operational support. Here, the main goal is to use process data to train Machine learning algorithms [11] to predict currently unknown characteristics about the outcome of a process (duration, activities, conformance, etc.), before they are realised. In other words, being *proactive* instead of *reactive* [1].

2.2 Event log data

Event log data are time-stamped pieces of information related to a single *case* or *instance* in a (business) process. Event-log data is most often found in process-aware information systems [1] such as Enterprise Resource Planning system (ERP) and Customer Relationship Management (CRM) systems. An example event-log consisting of events most often found in CRM-systems is found in table 1 below. Each row is an event which relate to a specific case.

Table 1. *Example event-log in a customer service unit.*

Case ID	Case type	Activity	Timestamp	Resource
1001	Complaint	Email interaction	01-01-2019 15:01	System
1001	Complaint	Phone interaction	01-01-2019 16:04	Employee 2
1001	Complaint	Subscription changes	01-01-2019 16:58	Employee 1
1002	Service termination	Email interaction	01-01-2019 12:01	System
1002	Service termination	Phone interaction	01-01-2019 13:10	Employee 2
1002	Service termination	Subscription terminated	01-01-2019 14:15	Employee 5
1002	Service termination	Sent invoice	02-01-2019 09:35	System

The sequence of events generated by a given case forms a *trace*. A trace contains events related to a single case only, and contains a case identifier, an event identifier, timestamps, and associated attributes.

2.3 Long Short-Term Memory Neural Networks

Recurrent Neural Networks are generally known for having problems with exploding or vanishing gradients [12]. This is due to the fact that they often have so many parameters that the gradient decays exponentially for every added layer. The Long Short-Term Memory RNN [13] has modified RNN units, called LSTM-cells. The main idea is that of forcing the gradient to be within 0 and 1, and adaptively change the amount of information that is learnt, by "forgetting" unimportant updates. A LSTM-cell have 5 different components, an input gate i_t, forget gate f_t, cell state c_t, output gate o_t, and the final output of the cell itself h_t:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc} + W_{hc}h_{t-1} + b_c) \tag{3}$$

$$o_t = \sigma(W_{x0}x_t + W_{h0}h_{t-1} + W_{c0}c_t + b_0) \tag{4}$$

$$h_t = o_t \tanh(c_t) \tag{5}$$

The main difference between a vanilla-type RNN and a LSTM is that hidden units are replaced with hidden cells which have multiple functions. To counter vanishing/exploding gradient problem [12]. Each of the units above i_t, f_t, c_t, o_t and h_t are j-dimensional vectors, where j denote the number of cells per layer. A LSTM is trained using backpropagation through time (BPTT) [13] and gradient descent.As can be seen by eq. 1, the input gate i_t takes in both the signal from the input data X_t as well as the previous hidden state h_{t-1}, and the previous cell state c_{t-1}. This enables each cell to learn both from the signal from the previous state, as well as signals multiple steps back in time (controlled by the forget f_t and output o_t gates).

This does in other words give the model a long-term memory, which can be beneficial for problems where the beginning of a sequence can have an importance to the prediction in the end of the sequence. This is one of the main motivations of adapting the LSTM network for predictive process monitoring [21], since the beginning of a trace have might in some cases be of importance for the rest of the progress. Some traces might also be very long, due to rework or a generally complex process, and in these scenarios it is beneficial to have a selective memory and e.g. only remember important signals from the beginning of a trace.

2.4 Loss functions

At the core of every machine learning algorithm is a *loss function* which calculate some error based on how well the model is doing at predicting the target. In other words, the model parameters are updated using using stochastic optimization, based on a loss function that is minimized, subject to the observations in a training data set X^{TRN}.

$$\hat{y_i} = g(X_i^{TRN}) \tag{6}$$

$$\min_z z = loss(y_i, \hat{y_i}) \tag{7}$$

Here, $g()$ is an abstraction of an arbitrary model which produce a prediction $\hat{y_i}$, given some input data X_i^{TRN}, and y_i is the ground truth of the *i'th* sample. The parameters of a model is most commonly optimized using a variant of stochastic gradient descent (SGD). The most basic form of the method can be seen in algorithm 1 below. Here λ is the learning rate, which control how much a given model parameter Θ_j is updated. An update is done with respect to the gradient of the loss of the prediction from sample i. What the model learns from the data is highly dependent on the loss function, as it either penalize or reward the parameter changes in the stochastic optimization process, based on the loss (or reward) function. One of the earliest steps in developing a machine learning model, is thus to align the form of the loss function, with the goal of the model [5, 24] (or business goal).

Algorithm 1 Stochastic gradient descent

Initialize parameter vector $\Theta \approx U(-1, 1)$
for iteration $1, ..., n$ **do**
  $i \leftarrow SelectAtRandom[1, n]$
  $\hat{y_i} = g(X_i^{TRN})$
  for Each parameter j in Θ **do**
    $\Theta_j = \Theta_j - \lambda \frac{\delta loss(y_i, \hat{y_i})}{\delta \Theta_j}$
  end for
end for
Return Θ

In the following, two of the most well-known loss functions are introduced, following 3 different variants of the MAE with different temporal penalties. Each of the proposed metrics penalize the residual at timestep t, based on its size, as well as the number of (maximal) remaining steps. Figure 1 illustrate the difference between the baseline MAE, and the proposed temporal variants given a fixed prediction error of 50 at each time step.

MSE The mean squared error (MSE) is a well-known loss function for regression problems. The MSE uses the $L2$ norm over the difference between the prediction and the target, which effectively makes it sensitive towards outliers. The MSE is time-invariant and does thus not penalize errors based on their order in a sequence. Its major *drawback* in terms of modelling event-log data, is that outliers are not uncommon between two consecutive events. For this reason, it has not been widely used in the field of predictive process monitoring [30].

$$MSE = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{T} \sum_{t=1}^{T_i} (y_t^i - \hat{y}_t^i)^2 \tag{8}$$

MAE The mean absolute error uses the $L1$ norm over the difference between the prediction and the target, making it robust towards outliers. MAE is time-invariant but ensure optimal *accuracy* om event-log data with large time differences between events [30]. This metric does not account for order of the errors, but due to the format of the prefix-log (discussed in [30]), the first prefixes will have the highest support and thus the lowest error. This loss should thus yield the best earliness, as well as overall accuracy. MAE is the most commonly used loss function for training RNN-based remaining time models [21, 20, 30].

$$MAE = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{T} \sum_{t=1}^{T_i} \mid y_t^i - \hat{y}_t^i \mid \tag{9}$$

MAE$_{EtD}$ Mean absolute error over all prefixes in the test set, with a temporal penalty in the form of a exponential decay factor depending on t alone. This loss

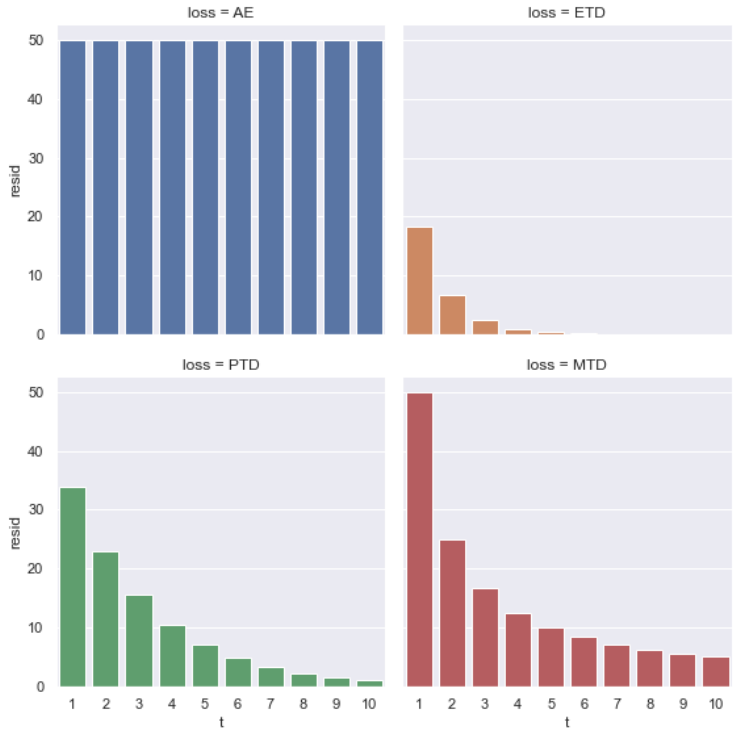

Fig. 1. Loss functions and their individual weighting of a constant error.

thus have a rapidly decreasing error as t becomes larger, meaning that early errors are weighted relatively higher throughout a trace, as compared to MAE_{PtD}. The mean absolute error with ***exponential temporal decay*** is formally defined as:

$$MAE_{EtD} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T}\sum_{t=1}^{T_i}\frac{\mid y_t^i - \hat{y}_t^i \mid}{e^{(t)}} \tag{10}$$

MAE_{EtD} weight the residuals with respect to *earliness* at an exponential rate, meaning it might have little use other than in cases where *earliness* is of out-most importance compared to accuracy.

MAE$_{PtD}$ Mean absolute error over all prefixes in the test set, with a penalty factor based on the power of the ratio between the maximal trace length T minus the current timestep t, and the maximal trace length T. As t gets larger, the ratio go towards 0, effectively weighing the residual at $t = T$ to 0. The mean absolute error with ***progressive temporal decay*** is formally defined as:

$$MAE_{PtD} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T}\sum_{t=1}^{T_i} \mid y_t^i - \hat{y}_t^i \mid^{\frac{T_i - t}{T_i}} \tag{11}$$

For this loss function, *earliness* is weighted higher than overall accuracy, due to the decreasing weight of the errors.

MAE$_{MtD}$ Mean absolute error over all prefixes in the test set, with a temporal decay factor depending on division with t alone. As t becomes larger, the weight of the error goes towards 0 at a *moderate* pace compared to MAE_{PtD} and MAE_{EtD} due to the decay factor $\frac{1}{t}$. MAE_{MtD} thus prioritize *earliness*, but weigh errors in later timesteps relatively higher than MAE_{PtD} and MAE_{EtD}. The mean absolute error with ***moderate temporal decay*** is formally defined as:

$$MAE_{MtD} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T}\sum_{t=1}^{T_i}\frac{\mid y_t^i - \hat{y}_t^i \mid}{t} \tag{12}$$

3 Methodology

To evaluate the performance of each of the loss functions compared to each other, a series of experiments have been performed. The experimental design is a block-design with *event-log* as the blocking variable, in addition, two factors are included in the experiments: the loss functions and the number of LSTM-cells. An overview can be seen in table 2. The experiments were performed with 8 replications per experiment, since training procedure itself is stochastic in nature. This resulted in a total of 320 runs.

Table 2. Experimental factors and their associated levels.

Block: Dataset	F: Loss function	F: LSTM cells
Hospital billing	MAE	200
Traffic fines	MSE	400
Helpdesk	MAE$_{PtD}$	
Sepsis	MAE$_{EtD}$	
	MAE$_{MtD}$	

In previous studies, LSTM models have been found to perform well on the data used in this study, when using between 200 and 400 LSTM-cells [30, 20, 21]. However, only 1 recurrent layer have been used in this study for computational reasons. The networks are regularized with recurrent dropout of 0.20, and optimized using *ADAM* [15] with an initial learning rate of 0.01 and early stopping at 5 epochs to prevent *over-fitting*. Each experiment was performed over 200 epochs with a batch size of 2048. These settings were found to perform the best

in an initial grid-search experiment (batch sizes: 256, 512, 1024, 2048, learning rates: 0.001, 0.005, 0.01, 0.02) with accuracy (MAE) as main goal.

3.1 Data

Four different event-logs from different domains are used in this study to test the performance of the proposed loss functions. The event-logs are publicly available, and well-known in the process-mining community, as they have previously been used in multiple studies [19, 20, 17, 18]. An overview of the differences between the event-logs, can be seen from tables 3 and 4. The *traffic fines* and *hospital billing* data has the largest amount of cases, and the longest average durations. The *Sepsis* and *Helpdesk* data both have a low number of cases, as well as a lower average duration.

Table 3. *Overview of the event-log data.*

Dataset name	Area	Period	Business goal
Sepsis[17]	Healthcare	07/11/2013 - 05/06/2015	Accuracy > Earliness
Helpdesk[29, 20]	IT Services	13/01/2010 - 03/01/2014	Earliness > Accuracy
Traffic fines[18]	Public admin.	01/01/2000 - 18/06/2013	Accuracy > Earliness
Hospital billing[19]	Healthcare	13/12/2012 - 19/01/2016	Earliness > Accuracy

The event-logs differ the most in terms of their trace lengths, where the Sepsis data has traces as long as 185. Compared to the trace distribution of the rest of the event-logs, this stands out as $> 80\%$ of the traces are longer than 10 events, where $< 5\%$ of the rest of the event-logs have more than 10 events.

For the largest event-logs, this is to some degree due to truncation, which have been performed to reduce the computational requirements. However, truncation have been done at values where the majority of the cases still have their full traces preserved. For the *Hospital billing* and *Traffic fines* event-logs, the same truncation values as in [25] have been used (see table 3). For the *Helpdesk* and *Sepsis* event-logs, no truncation was done.

Table 4. Dataset statistics (full event-log). Parenthesis denote dropped cases due to censoring.

Dataset name	Num. cases	Max trace length	Truncation	Avg. trace length
Sepsis	966 (83)	185	None	**18.51**
Helpdesk	4362 (218)	15	None	5.07
Traffic fines	**125815** (3800)	20	10	4.25
Hospital billing	63645 (13880)	**217**	8	5.73

3.2 Evaluation

Each of the event-logs are partitioned into a train and test period. The date that separate the two subsets, is the date that split the first event of 60% of the first cases into in the train period, and the remaining 40% is then then test period. Cases that overlap the two periods are censored (deleted), if they do not finish within their beginning period. This is similar to the approach in [30, 25], and help validate that the model can generalize outside the period in which it was trained.

Accuracy To evaluate the accuracy of the models, the most commonly used metric in literature is the mean absolute error over all traces in the test set [30], which can be seen in equation 13 below.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T}\sum_{t=1}^{T_i} \mid y_t^i - \hat{y}_t^i \mid \tag{13}$$

Where t is the prefix or event number in each trace, and i is the trace in the test period. To test that the differences between the accuracy across the experiments are not due to random chance, standard F-test is performed.

Earliness Earliness is commonly described in predictive process monitoring literature [30, 26, 6] as a models ability to predict a given target value early in a sequence, where information is minimal. To evaluate earliness in a single metric, the MAE over the first event/prefix $t = 1$ is calculated for each case as seen in equation 14.

$$MAE_t = \frac{1}{N}\sum_{i=1}^{N} \mid y_t^i - \hat{y}_t^i \mid \tag{14}$$

4 Results

In the following, the results will be presented with respect to each of the business goals. Goal A (accuracy) is evaluated in section 4.1 and goal B (earliness) in 4.2.

4.1 Overall accuracy across all traces

An overview of the average accuracy (MAE) across the loss functions, along with their confidence intervals can be seen in table 5. The results show very similar results for the Sepsis and Helpdesk data. In most cases (except from Heldesk data), the MSE loss achieve the worst performance. Across all experiments, the MAE baseline achieve accuracy very close to the best candidates, except from the Traffic fines data. In general, the MAE achieves the second or third best accuracy on average across all datasets.

Table 5. Average MAE of different loss functions across datasets, measured in days. **** = 0.01, ** = 0.05, * = 0.1.*

Loss function	Sepsis***	helpdesk	hospital_billing***	traffic_fines***
MAE	13.34 ±0.02	10.71 ±0.10	50.81 ±0.07	82.52 ±2.82
MAE_{EtD}	13.34 ±0.01	10.71 ±0.11	51.38 ±1.27	84.97 ±2.88
MAE_{PtD}	13.35 ±0.04	12.04 ±2.54	52.06 ±1.52	**77.58** ±0.73
MAE_{MtD}	**13.32** ±0.01	**10.64** ±0.12	**50.80** ±0.09	87.26 ±2.50
MSE	19.17 ±0.09	10.79 ±0.16	53.53 ±0.21	95.87 ±2.15

By a small margin, the MAE_{MtD} perform the best in 3 of 4 datasets. In the case of the traffic fines data, the MAE_{PtD} has considerably lower errors than any of the other loss functions (4.9 days lower on average). Differences across the loss functions were found to be insignificant on the helpdesk data.

4.2 Earliness performance

In line with the results on overall accuracy, the differences between the loss functions for the sepsis data is very small, except from the MSE which perform much worse than the rest of the candidates on this data. The differences are again insignificant for the Helpdesk data. Compared with the accuracy results, the MAE_{PtD} is the *worst* candidate on all but the Sepsis data, where the MSE is the best in the traffic fines data. The MAE_{PtD} does also seem to have very high variation for the Helpdesk and Hospital billing data.

For the two smallest event-logs (sepsis and helpdesk), the loss functions with best earliness performance are losses with exponential and moderate *temporal decay* (MAE_{EtD}, MAE_{MtD}) where squared error with no temporal penalty (MSE) perform the best on the largest event-log (traffic fines).

Table 6. Average MAE of different loss functions across event-logs at $t = 1$, measured in days. **** = 0.01, ** = 0.05, * = 0.1.*

Loss function	Sepsis***	helpdesk	hospital_billing***	traffic_fines***
MAE	12.29 ±0.01	8.86 ±0.73	61.36 ±0.26	98.39 ±0.92
MAE_{EtD}	**12.28** ±0.01	8.67 ±0.74	62.73 ±3.16	98.42 ±1.13
MAE_{PtD}	12.28 ±0.02	11.31 ±4.04	64.53 ±4.34	100.19 ±0.92
MAE_{MtD}	12.30 ±0.02	**8.34** ±0.67	**61.13** ±0.23	99.47 ±0.54
MSE	19.08 ±0.08	8.34 ±0.89	61.39 ±0.10	**97.53** ±1.69

5 Discussion and further research

The standard MAE loss was outperformed by temporal losses in all 4 event-logs in terms of accuracy, all but the traffic fines data in terms of earliness. The

MAE_{MtD} proved to be most universal, by performing the best in 3/4 event-logs. It was, however, largely ineffective for the traffic fines data. In this case the MAE_{PtD} had the best accuracy, and MSE the best earliness.

The effectiveness (with regards to accuracy and earliness) of the steepness of the temporal curvature of the loss appear to be related to the trace distribution. In particular, for event logs with many long traces like the Sepsis data, extreme weight like MAE_{EtD} result in the best earliness, but not accuracy at the same time. Current literature [30] suggest the existence of a *trade-off* between accuracy and earliness, which the findings in this study seem to support. The work in this study is initial, and further modifications might be made to the losses in future studies, such as parameterized versions with more flexibility. For event logs with a majority of short traces, moderate temporal penalty MAE_{MtD} seem to result in the best earliness. However, in order to systematically examine this relationship, a separate simulation study is suggested.

6 Reproducibility

The code used for the experiments is freely available on github, and can be found at[1]. The event-log data used in the experiments can be found at Eindhoven University of Technology website[2].

7 Conclusion

The aim of this study was to investigate the effect of modifying the loss functions with different temporal penalties, to see the impact on model performance with respect to two different business perspectives: Accuracy and earliness. Through a series of experiments across 4 real world event logs from different domains, 5 different loss functions were evaluated on their relative performance in terms of average accuracy and earliness (measured as performance at $t = 1$).

The results show that the temporal losses in most cases outperform MAE (the most commonly used loss in literature). The MAE_{MtD} perform the best in 5/8 cases. The results also indicate that the optimal degree of temporal penalty might rely on the properties of the event log. It is hypothesized that event logs with long traces benefit more from large temporal penalties, where event logs with short traces benefit more from moderate temporal penalties. It is therefore recommended that more research is done on the relation between temporal penalties and event log characteristics through e.g. simulation.

References

1. Van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, Heidelberg, 2 edn. (2016). https://doi.org/10.1007/978-3-662-49851-4

[1] *https://github.com/mikeriess/Temp_loss_RT*
[2] *https://research.tue.nl/en/datasets/*

2. Van der Aalst, W.M.: Process discovery from event data: Relating models and logs through abstractions. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8**(3), e1244. https://doi.org/10.1002/widm.1244
3. Camargo, M., Dumas, M., González-Rojas, O.: Learning Accurate LSTM Models of Business Processes, pp. 286–302 (07 2019). https://doi.org/10.1007/978-3-030-26619-6_19
4. Cheng, M.Y., Huang, K.Y., Hutomo, M.: Multiobjective dynamic-guiding pso for optimizing work shift schedules. Journal of Construction Engineering and Management **144** (09 2018). https://doi.org/10.1061/(ASCE)CO.1943-7862.0001548
5. Chollet, F.: Deep Learning with Python. Manning (2017)
6. Di Francescomarino, C., Dumas, M., Federici, M., Ghidini, C., Maggi, F.M., Rizzi, W., Simonetto, L.: Genetic algorithms for hyperparameter optimization in predictive business process monitoring **74**(P1) (2018)
7. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer Publishing Company, Incorporated, 2nd edn. (2018)
8. Evermann, J., Rehse, J.R., Fettke, P.: A deep learning approach for predicting process behaviour at runtime. International Conference on Business Process Management **1**, 490 (2016). https://doi.org/10.1007/978-3-319-58457-7, http://b-ok.xyz/book/2942192/1d94cd
9. Feurer, M., Hutter, F.: Hyperparameter Optimization, pp. 3–33. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-05318-$5_1, https://doi.org/10.1007/978-3-030-05318-5_1$
10. Francescomarino, C.D., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring (2015)
11. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016)
12. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: Kremer, S.C., Kolen, J.F. (eds.) A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press (2001)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**, 1735–80 (12 1997). https://doi.org/10.1162/neco.1997.9.8.1735
14. Jan, S.T., Ishakian, V., Muthusamy, V.: Ai trust in business processes: The need for process-aware explanations. Proceedings of the AAAI Conference on Artificial Intelligence **34**(08), 13403–13404 (Apr 2020). https://doi.org/10.1609/aaai.v34i08.7056, https://ojs.aaai.org/index.php/AAAI/article/view/7056
15. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. International Conference on Learning Representations (12 2014)
16. Lujak, M., Billhardt, H.: A distributed algorithm for dynamic break scheduling in emergency service fleets. In: An, B., Bazzan, A., Leite, J., Villata, S., van der Torre, L. (eds.) PRIMA 2017: Principles and Practice of Multi-Agent Systems. pp. 477–485. Springer International Publishing, Cham (2017)
17. Mannhardt, F., Blinde, D.: Analyzing the trajectories of patients with sepsis using process mining (06 2017)
18. Mannhardt, F., de Leoni, M., Reijers, H., Aalst, W.: Balanced multi-perspective checking of process conformance. Computing (02 2015). https://doi.org/10.1007/s00607-015-0441-1
19. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Data-driven process discovery - revealing conditional infrequent behavior from event logs. In: Dubois, E., Pohl, K. (eds.) Advanced Information Systems Engineering. pp. 545–560. Springer International Publishing, Cham (2017)

20. Navarin, N., Vincenzi, B., Polato, M., Sperduti, A.: LSTM networks for data-aware remaining time prediction of business process instances. 2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017 - Proceedings **2018-Janua**, 1–7 (2018). https://doi.org/10.1109/SSCI.2017.8285184
21. Niek Tax, Marlon dumas, Ilya veenich, Marcello la rosa: Predictive Business Process Monitoring with LSTM Neural Networks. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **10253 LNCS**, V–VI (2017). https://doi.org/10.1007/978-3-319-59536-8
22. Rama-Maneiro, E., Vidal, J., Lama, M.: Deep learning for predictive business process monitoring: Review and benchmark. ArXiv **abs/2009.13251** (2020)
23. Senderovich, A., Di Francescomarino, C., Ghidini, C., Jorbina, K., Maggi, F.: Intra and inter-case features in predictive process monitoring: A tale of two dimensions. pp. 306–323 (08 2017). https://doi.org/10.1007/978-3-319-65000-5_18
24. SPSS, Teradata, D.A.N..O.: Crisp-dm 1.0 - step-by-step data mining guide, https://www.the-modeling-agency.com/crisp-dm.pdf
25. Teinemaa, I., Dumas, M., Leontjeva, A., Maggi, F.M.: Temporal stability in predictive process monitoring. Data Mining and Knowledge Discovery **32**(5), 1306–1338 (2018). https://doi.org/10.1007/s10618-018-0575-9
26. Teinemaa, I., Dumas, M., Maggi, F., Di Francescomarino, C.: Predictive business process monitoring with structured and unstructured data. pp. 401–417 (09 2016). https://doi.org/10.1007/978-3-319-45348-4_23
27. Valouxis, C., Housos, E.: Hybrid optimization techniques for the workshift and rest assignment of nursing personnel. Artificial intelligence in medicine **20**, 155–75 (11 2000). https://doi.org/10.1016/S0933-3657(00)00062-2
28. Van Dongen, B.F., Crooy, R.A., Van Der Aalst, W.M.P.: Cycle time prediction: When will this case finally be finished? Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **5331 LNCS**(PART 1), 319–336 (2008). https://doi.org/10.1007/978-3-540-88871-0_22
29. Verenich, I.: Helpdesk. Mendeley data (Dec 2016). https://doi.org/10.17632/39bp3vv62t.1
30. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Transactions on Intelligent Systems and Technology **10**(4), 1–34 (2019). https://doi.org/10.1145/3331449
31. Verenich, I., Nguyen, H., Rosa, M.L., Dumas, M.: White-box prediction of process performance indicators via flow analysis. In: ACM International Conference Proceeding Series. vol. Part F128767 (2017). https://doi.org/10.1145/3084100.3084110
32. Wang, T.C., Liu, C.C.: Optimal work shift scheduling with fatigue minimization and day off preferences. Mathematical Problems in Engineering **2014** (04 2014). https://doi.org/10.1155/2014/751563