# scientific reports

OPEN

# Identifying cause-and-effect relationships of manufacturing errors using sequence-to-sequence learning

Jeff Reimer[1,4]✉, Yandong Wang[1,4], Sofiane Laridi[1], Juergen Urdich[3], Sören Wilmsmeier[2] & Gregory Palmer[1]

In car-body production the pre-formed sheet metal parts of the body are assembled on fully-automated production lines. The body passes through multiple stations in succession, and is processed according to the order requirements. The timely completion of orders depends on the individual station-based operations concluding within their scheduled cycle times. If an error occurs in one station, it can have a knock-on effect, resulting in delays on the downstream stations. To the best of our knowledge, there exist no methods for automatically distinguishing between *source* and *knock-on* errors in this setting, as well as establishing a causal relation between them. Utilizing real-time information about conditions collected by a production data acquisition system, we propose a novel *vehicle manufacturing analysis system*, which uses deep learning to establish a link between source and knock-on errors. We benchmark three sequence-to-sequence models, and introduce a novel composite time-weighted action metric for evaluating models in this context. We evaluate our framework on a real-world car production dataset recorded by Volkswagen Commercial Vehicles. Surprisingly we find that 71.68% of sequences contain either a source or knock-on error. With respect to seq2seq model training, we find that the Transformer demonstrates a better performance compared to LSTM and GRU in this domain, in particular when the prediction range with respect to the durations of future actions is increased.

Time-series forecasting is increasingly being used for predicting future events within business and industry to enable informed decision making[1,2]. In this paper we evaluate its potential to revolutionize automated vehicle manufacturing, where real-time information is collected by a *production data acquisition* (PDA) system. The effects of errors in interlinked manufacturing systems have dire consequences, such as production delays and even production system failure. In industrial manufacturing, downtimes are associated with high costs. To counteract downtimes, research and development has so far focused on predictive maintenance of equipment[3] and the use of alternative manufacturing routes through the production process[4]. However, these approaches do not explicitly focus on delays (micro-disturbances) in individual process steps, which are propagated throughout the process chain and amplified in the process.

The optimal utilisation of a fully automated car body production line depends on the individual station-based work steps completing within their scheduled cycle times. However, various disturbances with statistical significance are often detected. In particular, *source errors* (typically logged by the PDA system, e.g., "No components available."), may not only impact the current station, but also have a detrimental effect on the downstream workstations (hereinafter referred to as *stations*), resulting in *knock-on errors* and delays. Even minimal delays that are barely noticeable by humans can result in high additional costs.

While time-series forecasting for car-body production is challenging (due to discontinuities, spikes and segments[5]), deviations in the manufacturing process can be identified through comprehensive production data acquisition and the structured evaluation of these data . However, currently process delays and anomalies are identified through rule based classifiers that are manually programmed and maintained using extensive domain

[1]L3S Research Center, Leibniz University of Hanover, 30167 Hanover, Germany. [2]Institut für Fertigungstechnik und Werkzeugmaschinen, Leibniz University of Hanover, 30167 Hanover, Germany. [3]Volkswagen Group, Post Office Box 21 05 80, 30405 Hanover, Germany. [4]These authors contributed equally: Jeff Reimer and Yandong Wang. ✉email: reimer@l3s.de
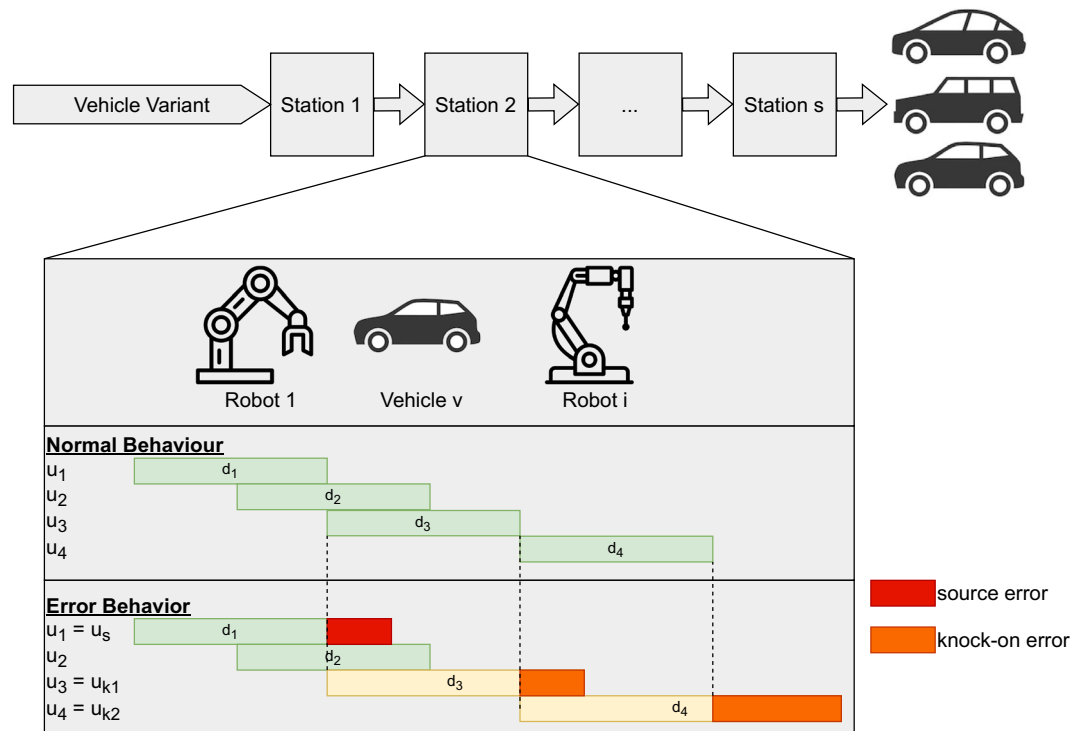
**Figure 1.** Production line for assembling vehicle variants, illustrating normal behaviour and error behavior in a Gantt-chart, with respect to durations $d$. Several actions can happen in parallel. In the error behavior scenario a source error for an action duration tuple $u_s$ (marked red) can lead to multiple knock-on error $u_{k1}$ and $u_{k2}$ (marked orange).

knowledge. In addition, further efforts are incurred in the interpretation of the processed data. This prevents production staff from rapidly deploying targeted countermeasures.

To the best of our knowledge no approach currently exists that automatically: (i.) learns to classify both source and knock-on errors; (ii.) establish a link between errors; and (iii.) measures the knock-on effect of source errors. In this work we take steps towards solving these challenges using machine learning (ML).

Our contributions can be summarized as follows:

(i.)   We introduce an ML-based *vehicle manufacturing analysis system* (VMAS) for process monitoring and cycle time optimization. The system is designed to detect delays and malfunctions in the production process early and automatically without manual effort. Furthermore, it identifies cause-effect relationships and predicts critical errors using sequence-to-sequence (seq2seq) models.

(ii.)  To enable a fair comparison between different seq2seq architectures for predicting errors in this context, we introduce a novel *Composite Time-weighted Action* (CTA) metric. Our metric allows stakeholders to weight the sequences of predictions output by our model, and choose to what extent immediate action duration predictions are prioritized over distant ones.

(iii.) Our VMAS is evaluated on PDA system data from the car body production of Volkswagen Commercial Vehicles. This includes the benchmarking of a number of popular seq2seq models for learning cause-effect relationships, including LSTM, GRU and Transformer. Surprisingly our evaluation shows the prevalence of source and knock-on errors, which occur in 71.68% of action sequences. The evaluation of prediction component meanwhile shows that the Transformer outperforms LSTM and GRU models, capable of accurately predicting the durations of up to seven actions into the future.

## Problem definition

The objective of our work is to analyse and better understand the performance of a car manufacturing system in terms of efficiency and productivity. Modeling and analyzing systems at different levels of abstraction (e.g., via a discrete event based simulation) is frequently used to gain insight and improve the design and operation of a manufacturing system process, such as logistic networks or the shop-floor material flow[6]. In this paper we introduce a pure data-driven approach towards solving this problem. First, we shall formally define our problem setting in this section.

In our car manufacturing system the vehicle body is processed through visiting a sequence of fully-automated stations. Each station comprises an ensemble of manufacturing robots (see Fig. 1). The production line is synchronous, each station has the same cycle time with no buffers. The stations are clocked out to measure the timeliness of the vehicles to-be assembled until they exit the production line. At each station actions are

performed, which we define as a triple $a = (s, v, i)$, consisting of a station $s$, vehicle code $v$, and action ID $i$. The production of vehicles also includes variants (left or right-hand drive vehicles for example) and therefore the nominal action is variable dependent on the vehicle variant, which is information included in the vehicle code.

Each action describes a specific and accomplished production step (for example transportation or manufacturing step). We are interested in the duration $d$ required to complete each executed action $a$, which can be viewed as an action duration tuple $u = (s, v, i, d)$. For notational convenience we shall refer to $d^a$ as the duration taken by an action $a$. In a clocked out vehicle production system, for each action $a$ there exists an expected maximum allowed duration $d^a_{max}$. The duration of an action $a$ must therefore be less than, or equal to, this expected allowed maximum time: $d^a \leq d^a_{max}$. In this work, we focus on sequences of actions and their durations, i.e., chains of action duration tuples, defined as $x = (u_1, u_2, \ldots, u_n)$. It is worth noting however, that actions can overlap, e.g., be executed in parallel. Therefore, it is not the case that one particular action has to have completed its task before another action can start. The sequence of actions is also dependent on the vehicle variant.

Malfunctions are a recurring problem in production. In the rare instance that a malfunction causes a long period of downtime, usually a situation analysis is conducted and possible fix is performed by staff engineers in the factory. However, our focus is on the small, seemingly insignificant and common delays, that not only have an effect on a station itself, but where subsequent perturbation propagate to downstream stations, causing further delays. Here we consider executed actions with two types of errors resulting from delays, where the duration $d^a > d^a_{max}$: (i.) *source errors*, $u_s$ where an abnormal action duration is accompanied by an error message; (ii.) *knock-on errors*, where an action $u_k$ with an abnormally long action duration is not accompanied by an error message. In this work we are interested in knock-on errors that occur after a source error (i. e., a logged error) within the sequence of actions: $(\ldots, u_s, \ldots, u_k, \ldots)$.

An individual source error may appear inconspicuous, since source errors do not have to deviate significantly from the normal time. However, the knock-on errors, which also do not have to deviate much individually, can result in a significant accumulated time-delay. From the PDA system it is not possible to understand the scope of downstream actions and the knock-on effects of a source error. It is only possible to assert that downstream actions can accumulate time-delays without reported fault messages. Consequentially, this leads to a significant loss of effective production time overall.

The analysis of the relationship between source and knock-on errors is challenging due to the latent entanglement of the individual processes of actions. An argument can be made that a rule-based model can determine the relationship of a source and knock-on errors. However, this approach requires extensive domain knowledge and the resulting model would not be transferable across stations. We hypothesize that deep learning-based seq2seq models are able to learn the nominal sequence of actions and, more importantly for the producer, the recurring source and knock-on errors in them as well. If the errors can be predicted with a satisfying accuracy, then it means inherent causal-effect rules are learned from the abundance of data.

## Related work

The design and operation of manufacturing systems can be improved by modelling them at different levels of abstraction. Material flow within a manufacturing plant, as well as logistic chains from original equipment manufacturers (OEMs), requires strategic foresight and ruling for a just-in-time as well as just-in-sequence delivery. Advanced modeling approaches have the potential to enable system designers to analyze phenomena that frequently lead to delays (e.g., sequence scrambling) and take steps towards a stabilised production[6]. As result, flexible manufacturing system have received significant attention from researchers from various fields, where approaches such as the bottleneck-based dispatching heuristic aim to improve the throughput of manufacturing shop-floors. However, bottleneck shifting can occur as a result of unexpected anomalies appearing within the lanes, e.g., sequence scrambling or machine failure. To address this, Huang et al.[7] propose a method that combines a deep neural network (DNN) and time series analysis for predicting and resolving future bottlenecks in an Internet of Things enabled production environment. In contrast, our work focuses on a singular lane where sequence scrambling is not possible. Instead our focus is on the modeling of the small, subliminal as well as common delays and measure their error propagation significance.

Within the context of intelligent industrial production a significant amount of data-driven research has been dedicated towards forecasting, failure prediction and anomaly detection using time series data[8,9]. The literature in this area provides an overview of the suitability of approaches designed to solve these problems when applied to various production contexts, often featuring a comparison between traditional machine learning approaches and that of advanced deep neural networks. Failure prediction for instance has often been limited to standard key performance indicators. Moura et al.[10] evaluate the effectiveness of support vector machines in forecasting time-to-failure and reliability of engineered components based on time series data. Yadav et al.[11] present a procedure to forecast time-between-failure of software during its testing phase by employing fuzzy time series approach. Others use artificial neural networks or statistical approaches to model machine tool failure durations continuously and cause-specific[12,13].

*Recurrent neural network* (RNN) models meanwhile are capable of identifying long-term dependencies from time-series data directly[14]. Successes here include: multi-step time-series forecasting of future system load with the goal of performing anomaly detection and system resource management, enabling the automated scaling in anticipation of changes to the load[15]; and using stacked LSTM networks to detect deviations from normal behaviour without any pre-specified context window or pre-processing[16]. However, the performance of encoder-decoder architectures relying on memory cells alone typically suffers, as the encoding step must learn a representation for an (potentially lengthy) input sequence. Here *attention based encoder-decoder architectures* provide a solution, where the hidden states from all encoder nodes are made available at every time step. In-fact,

pioneering work by[17] demonstrated that one can dispense with recurrent units and rely solely on the attention, introducing the Transformer. Further improvements can be obtained via Transformers implemented with GRUs[18].

Not surprisingly attention based approaches are increasingly being applied to industry problems[19]. Li et al.[20] present a novel approach to extracting dynamic time-delays to reconstruct multivariate data for an improved attention-based LSTM prediction model and apply it in the context of industrial distillation and methanol production processes. But they do not explicitly consider failure propagation in concatenated manufacturing systems to evaluate failure criticality and to generate a reliable failure impact prediction. Attention-based models have also been applied to failure prediction and rated as favorable. LI et al.[21] propose an attention-based deep survival model to convert a sequence of signals to a sequence of survival probabilities in the context of real-time monitoring. While Jiang et al.[9] use time series multiple channel convolutional neural network integrated with the attention-based LSTM network for remaining useful life prediction of bearings. Near real-time disturbance detection becomes possible with the attention-based LSTM encoder-decoder network by Yuan et al.[22], which allows to align an input time series with the output time series and to dynamically choose the most relevant contextual information while forecasting. In contrast to previous work, we propose a workflow and evaluate seq2seq approaches for failure impact prediction in concatenated manufacturing systems.

## Vehicle manufacturing analysis system

In this section we introduce our vehicle manufacturing analyses system (VMAS), which we developed according to the cross-industry standard process for data mining (CRISP-DM)[23]. Our use-case has two separate databases that store *cycle times* and *error reports* data respectively. The PDA system in our use-case registers and stores action duration tuples $u$ in the cycle times database. The data are processed by our VMAS, which consists of two main components: 1.) an error classification module for identifying source and knock-on errors within our dataset; and 2.) a duration prediction module, trained to predict the time required for $n$ future actions. We describe each component in detail below and a flowchart can be found in Fig. 3.

**Module 1: error classification.** We begin with an actions dataset $\mathcal{D}_a$ and an error reports database that stores timestamped error logs as well as the duration of the logged errors. Each sample $x \in \mathcal{D}_a$, is a sequence of action duration tuples $x = (u_0, u_1, u_2, \dots, u_n)$, where $n$ is the number of actions executed during a *complete sequence*. The error classification module of our workflow allows us to identify the most significant errors within our dataset, and distinguishes source from knock-on errors. More specifically, this module allows us to split samples from our dataset into four subsets: normal $\mathcal{D}_n$, source errors $\mathcal{D}_s$, knock-on errors $\mathcal{D}_k$ and misc $\mathcal{D}_m$. This splitting of the dataset into sub-sets serves two purposes: i.) The classification in $\mathcal{D}_s$ and $\mathcal{D}_k$ helps the stakeholder to conduct an automated analysis of all actions and it eliminates the need for manual and often time consuming inspection of actions; ii.) During preliminary trials we found that samples from $\mathcal{D}_m$ are exceedingly rare and disturb the training of the seq2seq models. Therefore, the error classification module also provides a valuable preprocessing step prior to training our seq2seq models to predict future delays. Below we first discuss our approach for labelling our samples, and then formally define the conditions for a sequence $x$ to belong to one of the four subsets. We note that for our VMAS there is an assumption that *all* source errors are *logged errors*.

*Labelling* We use the maximum likelihood estimation (MLE) method for the labelling of anomalous behavior. For each action $a$, a normal (Gaussian) distribution is sought that fits the existing data distribution with respect to the frequency of each duration (for an example see Fig. 2).

The density function of the normal distribution contains two parameters: the expected value μ and standard deviation σ, which determine the shape of the density function and the probability corresponding to a point in the distribution. The MLE method is a parametric estimation procedure that finds μ and σ that seem most plausible for the distribution of the observation $z$ [24]:

$$f(z \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right). \tag{1}$$

The density function describes the magnitude of the probability of $z$ coming from a distribution with μ and σ. The joint density function can be factorised as follows:

$$f(z_1, z_2, \dots, z_n \mid \vartheta) = \Pi_{i=1}^n f(z_i \mid \vartheta) \tag{2}$$

For a fixed observed variable, the joint density function of $z$ can be interpreted. This leads to likelihood function:

$$L(\vartheta) = \Pi_{i=1}^n f_\vartheta(z_i) \tag{3}$$

The value of $\vartheta$ is sought for which the sample values $z_1, z_2, \dots, z_n$ have the largest density function. Therefore, the higher the likelihood, the more plausible a parameter value $\vartheta$ is. As long as the likelihood function is differentiable, the maximum of the function can be determined. Thus, the parameters μ and σ can be obtained.

Next, we seek to identify *high frequency peaks* with respect to the durations $d^a$ for an action $a$, that exceed the nominal duration $d_{norm}^a$. We are interested in significant errors, where we use the MLE threshold to determine if an error is significant or not. We denote significant errors as $d_{sig}^a$. These abnormal and distinct duration are indicating a recurring behaviour. We formally define the criteria for each sub-set below:

- *Source errors* are samples where for each complete sequence $x$, we have at least one action duration that is considered critical, of statistical significance, and is accompanied by an error message. More formally: a com-
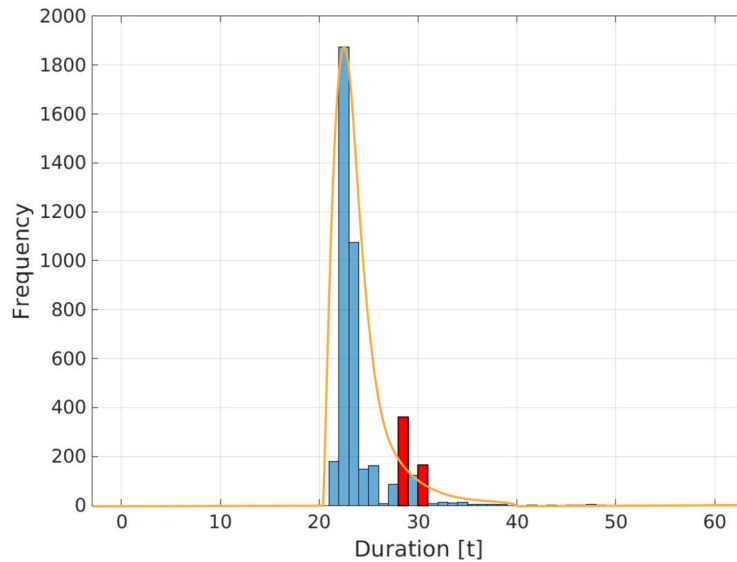
**Figure 2.** Peak detection in a histogram of durations for a specific action utilizing MLE as a threshold, based on data from a real-world car production dataset recorded by Volkswagen Commercial Vehicles. Peaks above the MLE threshold (red) are considered as significant errors.
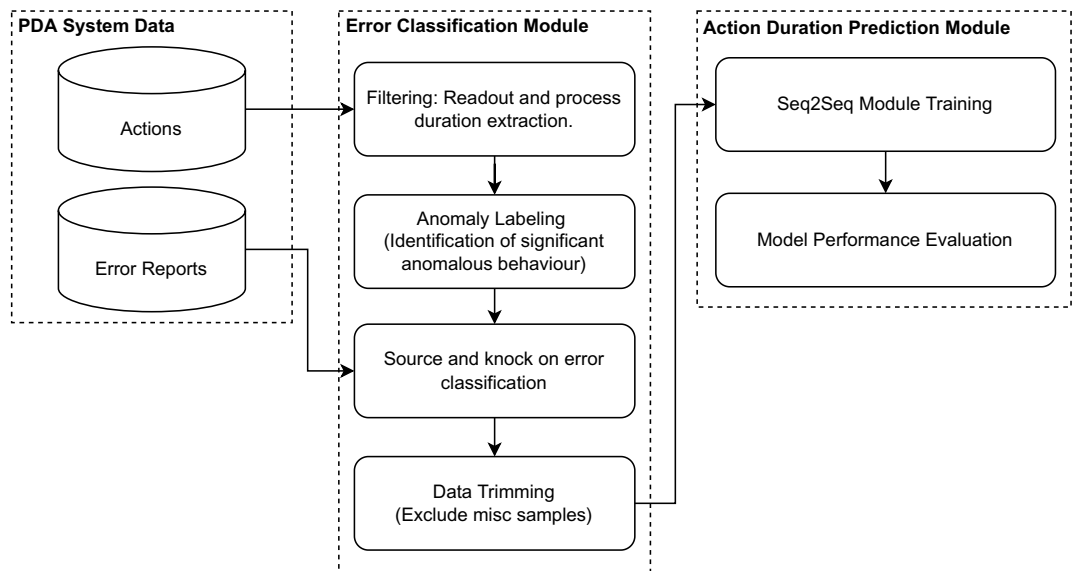


**Figure 3.** Flowchart of our vehicle manufacturing analyses system (VMAS). First the PDA system data is processed by our Error Classification Module, resulting in four sub-sets: source errors, knock-on errors, normal and misc. The resulting source and knock-on error sets can then be used by our stakeholders for obtaining valuable insights w. r. t. causes of delays. Next, upon excluding misc samples, we use our data for training sequence-to-sequence models for predicting future delays.

plete action sequence $x$ is considered a *source error sequence* $x \in \mathcal{D}_s$ iff there exists an action duration tuple $u \in x$, where the duration is $d_{sig}^a$ and there is a corresponding error message in the error reports database.
- *Knock on errors* meet the same criteria as source errors, but lack an accompanying error message for $d_{sig}^a$. Therefore, a complete action sequence $x$ is considered a *knock-on error sequence* $x \in \mathcal{D}_s$ iff there exists an action duration tuple $u \in x$, where the duration is $d_{sig}^a$ and there is not a corresponding error message in in the error reports database.
- *Normal* samples don't include $d_{sig}^a$. Therefore, a complete sequence $x$ is considered a *normal sequence* $x \in \mathcal{D}_n$ iff for all $u \in x$ there does not exist a duration $d_{sig}^a$.
- *Misc* contains two types of complete action sequences: i.) where for an action $u$ there is a duration $d_{sig}^a$ that is above a defined global threshold $d_{globalmax}^a$, meaning the duration is either intended (e. g., the production line

is paused), or staff are handling them; and ii.) where $x$ consists only of duration $d$ that exceed the nominal duration, but each of low significance, i. e., not exceeding the corresponding MLE threshold.

It is worth noting that $\mathscr{D}_n \cup \mathscr{D}_s \cup \mathscr{D}_k$ may contain individual $d^a$ above the nominal duration, but below the threshold determined by the MLE, and therefore are errors of low significance. There can also exist an intersection between source and knock-on errors. Furthermore, the labelling of knock-on errors is deliberately modular, as different methods can be applied here based on the stakeholder's requirements. Naturally this will impact the subsequent training of our seq2seq models, and therefore their predictions.

**Module 2: action duration prediction.**  While our error classification module assigns labels to past errors, our second module focuses on the prediction of future errors. Upon removing misc samples, we utilize our dataset to train seq2seq models to predict knock-on errors. Given a sequence of action duration tuples our objective is to predict the time required by each of the next $n$ steps. We therfore convert the data received from the error classification module into a dataset containing pairs $(x, y) \in \mathscr{D}$, where each $x$ is a sequence of action duration tuples $x = (u_{t-n}, u_{t-n+1}, u_{t-n+2}, \ldots, u_t)$, and $y$ is the duration of the $n$ actions that follow $y = (d_t^a, d_{t+1}^a, d_{t+2}^a, \ldots, d_{t+n}^a)$. Using these data, we train and evaluate popular seq2seq models, including LSTM[25], GRU[14] and the Transformer[17]. The later is of particular interest, as it represents the current state-of-the-art for a number of seq2seq tasks. Vaswani et al.[17] presented the Transformer architecture for the Natural Language Processing (NLP) or Transductor task domain. Previous RNN/CNN architectures pose a natural obstacle to the parallelization of sequences. The Transformer architecture replaces the recurrent architecture by its attention mechanism and encodes the symbolic position in the sequence. This relates two distant sequences of input and output, which in turn can take place in parallel. The time for training is thereby significantly shortened. At the same time, the sequential computation is reduced and the complexity $O(1)$ of dependencies between two symbols, regardless of their distance from each other in the sequence, remains the same[17]. Next we consider a novel metric for fairly evaluating models of different architectures–in particular regarding the number of steps $n$–using a single scalar (Fig. 3).

## Composite Time-weighted Actions Metric

A sequence of actions can consist either of nominal behaviour or error behaviour by having at least one source or knock-on error included. To predict a distinct behavior we pass a partial sequence of actions to a seq2seq model to predict $n$ actions into the future. However, in production there are a number of scenarios (including our current one), where a greater weighting needs to be placed on the performance of the classifier with respect to short term predictions in order to enable a quick intervention. Therefore, to evaluate our model in this setting a metric is required that: i.) assigns a higher importance on the immediate predictions versus later predictions in the sequence of actions; ii.) allows a prediction of quality invariant of the number of predicted future steps $n$, in order to cross compare various setups; iii.) has high precision when predicting the duration of an action. For the evaluation of any seq2seq model we introduce the *Composite Time-weighted Action* (CTA) metric. The CTA is a convex combination of a Time-weighted Action RMSE (which we introduce below) and an F1 score that uses a threshold $b$:

$$\text{CTA} = \tau(\text{TARMSE}) + (1 - \tau)(\text{F1}). \tag{4}$$

In the above equation stakeholders can use the weighting $\tau$ to either emphasize the TARMSE or precision when evaluating and comparing models. In the following we will discuss the two components.

*Time-weighted Action RMSE (TARMSE)* To measure the performance of a model globally, we introduce a *Time-weighted RMSE* that returns a single scalar metric for the $n$ model outputs. The model performance should not diminish if the starting point of predictions varies within the sequence of actions. For our current problem setting immediate predictions should also have a higher importance than later ones. In order to compensate for the increase of uncertainty we introduce a weighting factor $\beta_i = e^{-i}$ with $i$ being the action index. The following formula is considering only predictions which are below the expected allowed maximum time $d_{globalmax}^a$:

$$\text{TARMSE}(n, k) = \frac{1}{k \times S(n)} \sum_{i=1}^{n} \beta_i (k - R_i) \tag{5}$$

with

$$S(n) = \frac{e^{-n} - 1}{1 - e} = \sum_{i=1}^{n} \beta_i \tag{6}$$

and

$$\beta_i = e^{-i}. \tag{7}$$

In Equation (5) $R_i$ is the RMSE for action $i$ and the $k$ value is oriented to the mean standard deviation of all the times of actions in this station within the max tolerance. The standard deviation has the property of fitting a Gaussian distribution. Therefore, it can be considered as the amount of error that naturally occurs in the estimates of the target variable.

*F1 Score* By introducing a threshold value $b$, it is possible to gauge how many of the action predictions are considered correct and thereby obtain an evaluation of the binary classifier. The threshold $b$ is selected using

| LSTM or GRU Model configurations | |
|---|---|
| Nodes per layer | 100 |
| Layers | 4 |
| Dropout | 0.2 |
| **Transformer Model Parameters** | |
| Number of heads | 2 |
| Head Size | 256 |
| Feed Forward Dimension | 1024 |
| Number of Transformer Blocks | 4 |
| MLP Units | 1024 |
| Dropout | 0.1 |
| **General** | |
| Epochs | 50 |
| Batch Size | 128 |
| Optimizer | Adam |
| Learning rate | 0.001 |

**Table 1.** Hyperparameters.

domain knowledge. With the knowledge where the expected value for either the nominal or pattern error behaviour is, we can compare our predictions with the ground truth. Our reason for including the F1 score in our composition metric is that it will be used to evaluate models within a real-world production environment. Within our target domain, a low false positive warning rate is required, as otherwise workers will consider warnings as unreliable and not trustworthy. Given that alerts require investigation, false positives will result in a superfluous waste of time.

## Empirical evaluation

**Experiment setup.** For the empirical evaluation we first discuss the result of applying our error classification module to the dataset provided by Volkswagen Commercial Vehicles . This dataset contains hierarchical actions. However, to enhance our sequence-to-sequence model training we remove the hierarchy of actions to lessen the noise in the data. Therefore, in the last data preprocessing step we remove the hierarchy of actions, as superordinate actions document the total times of subactions. We focus on a single station to test the hypothesis that pattern errors can be learnt from the completion time of actions within an action sequence. We consider an exemplary station that has 22 actions. This workstation is of particular interest for Volkswagen Commercial Vehicles, as delays are frequently observed. For our error classification module we set the global threshold as ten times $d_{globalmax}^a = 10 \times d_{max}^a$. For the scalar for obtaining $d_{globalmax}^a$ we ran preliminary trials with 3, 5, 10, but found the former two removed a large proportion of data points, impacting the accuracy of the predictions of the seq2seq models. We therefore chose a scalar of 10, allowing us to retain 94.8% of the data points. The parameters chosen for our seq2seq models can be found in Table 1. Four different seq2seq architectures $n$-$m$ are compared with respect to length of the input sequence $n$ and the number of outputs $m$: 5–2, 5–5, 5–7, 7–7. We conducted 10 training runs per model architecture, and the results in Table 2 are the averages from applying the models to our test data, using a 80% sequences for training (30,744 sequences), 20% of sequences for the test (7,686 sequences) split. From the application point of view, it is important to choose an F1 threshold value $b$ that generalizes across vehicle variant dependent actions, which can have very different lead times. For actions with very short lead times (1-2 seconds) the sensor noise of the PDA system is larger than 5%, therefore a suitably large threshold value needs to be selected. In collaboration with VW Commercial Vehicles we found in preliminary sensitivity analyses conducted with 5%, 10% and 20% that the F1 threshold $b = 10\%$ is a suitable operating value. After considering only actions below $d_{max}^a$ and then calculate the RMSE from all of them we get $k = 5.14$.

**Error classification results.** Upon applying the error classification module to our dataset we first remove 2,106 out of 40,536 sequences of actions (corresponding to 40,536 vehicles processed on the station) that contain outliers (5.2%). Next, we apply our MLE based approach, finding that 3.94% of samples sequences containing at least one source error (without knock-on errors), 61.20% containing knock-on errors and 6.54% containing both. With respect to normal and misc samples, we have 0.068% only normal, 0.0902% only misc, and 18.62% *only* misc and normal. An analysis of the dataset following preprocessing reveals that 71.68% of sequences contain at least one error. Therefore, surprisingly the majority of the sequences contain either a source or knock-on errors. As mentioned, during preliminary trials we also find that the small percentage of misc sequences can negatively impact the performance of the seq2seq models. We discuss this in more detail in the evaluation of our seq2seq model results below.

**Sequence-to-sequence model results.** In this section we shall first compare the results for the four different seq2seq architecture types based on length of the input sequences and predictions. Then we shall take a closer look at the impact of the choice for the TARMSE weighting factor $\tau$ for evaluating our models. An over-

| Metric | Outputs $m = 2$ | | | Outputs $m = 5$ | | | | | | Outputs $m = 7$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GRU 5-2 | LSTM 5-2 | TF 5-2 | GRU 5-5 | LSTM 5-5 | TF 5-5 | GRU 7-5 | LSTM 7-5 | TF 7-5 | GRU 7-7 | LSTM 7-7 | TF 7-7 |
| $RMSE_1$ | 4.14 ± 0.24 | 4.01 ± 0.39 | 2.95 ± 0.02 | 4.07 ± 0.18 | 4.03 ± 0.21 | 3.04 ± 0.02 | 3.99 ± 0.22 | 4.08 ± 0.32 | 2.68 ± 0.04 | 4.10 ± 0.32 | 4.05 ± 0.40 | 2.72 ± 0.03 |
| $RMSE_2$ | 4.12 ± 0.63 | 4.06 ± 0.49 | 3.29 ± 0.05 | 3.72 ± 0.29 | 3.88 ± 0.21 | 2.70 ± 0.02 | 3.97 ± 0.19 | 4.15 ± 0.35 | 2.55 ± 0.02 | 4.00 ± 0.29 | 4.07 ± 0.26 | 2.56 ± 0.02 |
| $RMSE_3$ | NA | NA | NA | 3.50 ± 0.18 | 3.84 ± 0.30 | 2.43 ± 0.02 | 4.09 ± 0.41 | 4.21 ± 0.24 | 2.59 ± 0.02 | 4.24 ± 0.41 | 4.30 ± 0.16 | 2.57 ± 0.03 |
| $RMSE_4$ | NA | NA | NA | 3.51 ± 0.23 | 3.68 ± 0.34 | 2.35 ± 0.02 | 3.77 ± 0.35 | 3.92 ± 0.34 | 2.57 ± 0.04 | 4.06 ± 0.28 | 3.82 ± 0.32 | 2.57 ± 0.02 |
| $RMSE_5$ | NA | NA | NA | 4.29 ± 0.41 | 4.09 ± 0.53 | 2.77 ± 0.06 | 4.23 ± 0.52 | 4.29 ± 0.84 | 2.78 ± 0.10 | 4.15 ± 0.27 | 4.15 ± 0.29 | 2.66 ± 0.05 |
| $RMSE_6$ | NA | NA | NA | NA | NA | NA | NA | NA | NA | 3.49 ± 0.31 | 3.51 ± 0.98 | 2.86 ± 0.06 |
| $RMSE_7$ | NA | NA | NA | NA | NA | NA | NA | NA | NA | 3.28 ± 0.24 | 3.68 ± 0.56 | 2.70 ± 0.07 |
| $F1_1$ | 0.94 ± 0.02 | 0.95 ± 0.01 | 0.80 ± 0.02 | 0.94 ± 0.02 | 0.94 ± 0.02 | 0.79 ± 0.03 | 0.89 ± 0.03 | 0.91 ± 0.04 | 0.81 ± 0.01 | 0.87 ± 0.04 | 0.92 ± 0.02 | 0.80 ± 0.01 |
| $F1_2$ | 0.95 ± 0.01 | 0.95 ± 0.01 | 0.82 ± 0.03 | 0.95 ± 0.02 | 0.94 ± 0.01 | 0.81 ± 0.02 | 0.89 ± 0.01 | 0.90 ± 0.02 | 0.81 ± 0.02 | 0.89 ± 0.03 | 0.90 ± 0.01 | 0.80 ± 0.02 |
| $F1_3$ | NA | NA | NA | 0.89 ± 0.03 | 0.89 ± 0.02 | 0.79 ± 0.03 | 0.94 ± 0.01 | 0.94 ± 0.01 | 0.81 ± 0.01 | 0.91 ± 0.02 | 0.93 ± 0.01 | 0.79 ± 0.01 |
| $F1_4$ | NA | NA | NA | 0.89 ± 0.01 | 0.90 ± 0.01 | 0.80 ± 0.01 | 0.94 ± 0.01 | 0.94 ± 0.01 | 0.81 ± 0.01 | 0.93 ± 0.02 | 0.94 ± 0.01 | 0.79 ± 0.01 |
| $F1_5$ | NA | NA | NA | 0.92 ± 0.01 | 0.93 ± 0.01 | 0.78 ± 0.01 | 0.92 ± 0.01 | 0.92 ± 0.01 | 0.75 ± 0.02 | 0.85 ± 0.04 | 0.89 ± 0.02 | 0.75 ± 0.02 |
| $F1_6$ | NA | NA | NA | NA | NA | NA | NA | NA | NA | 0.85 ± 0.03 | 0.88 ± 0.04 | 0.72 ± 0.02 |
| $F1_7$ | NA | NA | NA | NA | NA | NA | NA | NA | NA | 0.85 ± 0.04 | 0.88 ± 0.03 | 0.76 ± 0.03 |
| TARMSE | 0.20 ± 0.05 | 0.22 ± 0.08 | 0.41 ± 0.01 | 0.24 ± 0.02 | 0.23 ± 0.03 | 0.44 ± 0.00 | 0.22 ± 0.03 | 0.20 ± 0.04 | 0.49 ± 0.01 | 0.21 ± 0.05 | 0.21 ± 0.05 | 0.48 ± 0.00 |
| F1 | 0.94 ± 0.01 | 0.95 ± 0.01 | 0.80 ± 0.01 | 0.94 ± 0.02 | 0.93 ± 0.01 | 0.80 ± 0.02 | 0.89 ± 0.01 | 0.91 ± 0.03 | 0.81 ± 0.01 | 0.88 ± 0.02 | 0.92 ± 0.01 | 0.80 ± 0.01 |
| CTA (%) | 59.89 ± 3.02 | 58.49 ± 4.08 | 60.55 ± 0.76 | 58.67 ± 1.33 | 58.11 ± 1.53 | 61.69 ± 0.85 | 55.83 ± 2.22 | 55.80 ± 2.19 | 64.75 ± 0.59 | 54.54 ± 3.58 | 56.22 ± 2.55 | 63.88 ± 0.70 |

**Table 2.** Results table comparing each of our LSTM, GRU and Transformer (TF) architectures *n-m*, where *n* represents the number of inputs, and *m* the number of model outputs. The table provides RMSE and F1 scores for each outputs, as well as TARMSE, averge F1 and composite time-weighted actions (CTA) scores (using $\tau = 0.5$ for the later).

view of the results obtained for each setting is provided in Table 2, where the balance between *TARMSE* and *F*1 is $\tau = 0.5$. Finally, we conduct an ablation study, showing the extent to which including misc samples impacts the performance of our seq2seq models.

*Setup 5-2* We first consider the results for training a seq2seq model to predict two future action durations based on five historic actions (setup 5-2). The TARMSE of the GRU and LSTM models is at 0.2 ± 0.05 and 0.22 ± 0.08 while the Transformer performs best with 0.41 ± 0.01. Yet the summarized F1 score is lower at 0.8 ± 0.01 while the GRU and LSTM are better with 0.94 ± 0.01 or 0.95 ± 0.01. Combined the CTA shows us that the GRU at 59.89 ± 3.02 and LSTM at 58.49 ± 4.08 are minimal worse w. r. t. mean than the Transformer at 60.55 ± 0.76. However, the standard deviation shows us that the Transformer is more consistent.

*Setup 5-5* In the next setup 5-5 we see a similar behavior to the 5-2 setup. The TARMSE is for the GRU and LSTM at 0.24 ± 0.00 and 0.23 ± 0.01 respectively, and for the Transformer it is 0.44 ± 0.00. The F1 is 0.84 ± 0.02 for the GRU, 0.93 ± 0.01 for the LSTM and 0.80 ± 0.02 for the Transformer. The CTA shows that the Transformer is better with 61.69 ± 0.85 than GRU's 58.67 ± 1.33 and LSTM's 58.11 ± 1.53.

*Setup 7-5* Next we keep the number of future predictions the same but consider a history of seven actions. The TARMSE for GRU is 0.22 ± 0.03, LSTM is 0.20 ± 0.04 and Transformer slightly increasing than the previous 5-5 setup to now 0.49 ± 0.01. The F1 score slightly decrease to 0.89 ± 0.02 for the GRU, 0.91 ± 0.03 for the LSTM and 0.81 ± 0.01 for the Transformer. We notice a slight improvement in the CTA for the Transformer at 64.75 ± 0.59 while the GRU at 55.83 ± 2.12 and LSTM at 55.80 ± 2.19 decrease and notably the standard deviation is significantly higher now compared to the 5-5 setup.

*Setup 7-7* Lastly we consider seven previous actions in a sequence and let the models predict seven actions into the future. The TARMSE of the GRU and LSTM are both at 0.21 ± 0.05 and the for Transformer at 0.48 ± 0.00. It should be noted that the standard deviation for the Transformer is considered that low that the rounding shows zero here. The F1 is for the GRU at 0.88 ± 0.02, for the LSTM at 0.92 ± 0.01 and for the Transformer similar to before 0.80 ± 0.01. For the GRU and LSTM the CTA are at 54.24 ± 3.58 and 56.22 ± 2.55 while the Transformer is at 63.88 ± 0.70. Across all setups we can observe that the the Transformer shows better performance when predicting future actions by considering the TARMSE. We see an improving trend in the TARMSE for the Transformer the more input actions are considered and prediction range increased. However the F1 score is higher for the GRU and LSTM models.

*CTA Weighting Factor* We note that the weighting factor $\tau$ influences our final result for the CTA. In Figure 4 we demonstrate the weighting factor between TARMSE and F1 for the chosen models in our setup with seven past actions to be considered and seven actions need to be predicted. GRU and LSTM demonstrate here that due to their higher F1 score they initially start higher than the Transformer model. With increasing $\tau$ the Transformer model surpasses the GRU model ($\tau = 0.229$) and LSTM model ($\tau = 0.308$) because of its better TARMSE.

*Ablation Study* As mentioned above, during preliminary trials we found that the inclusion of misc samples reduced the performance of the seq2seq models when used during training. We illustrate this in Figure 5, where we observe the RMSE of four model groups. In each group the model is the same, but differ in the
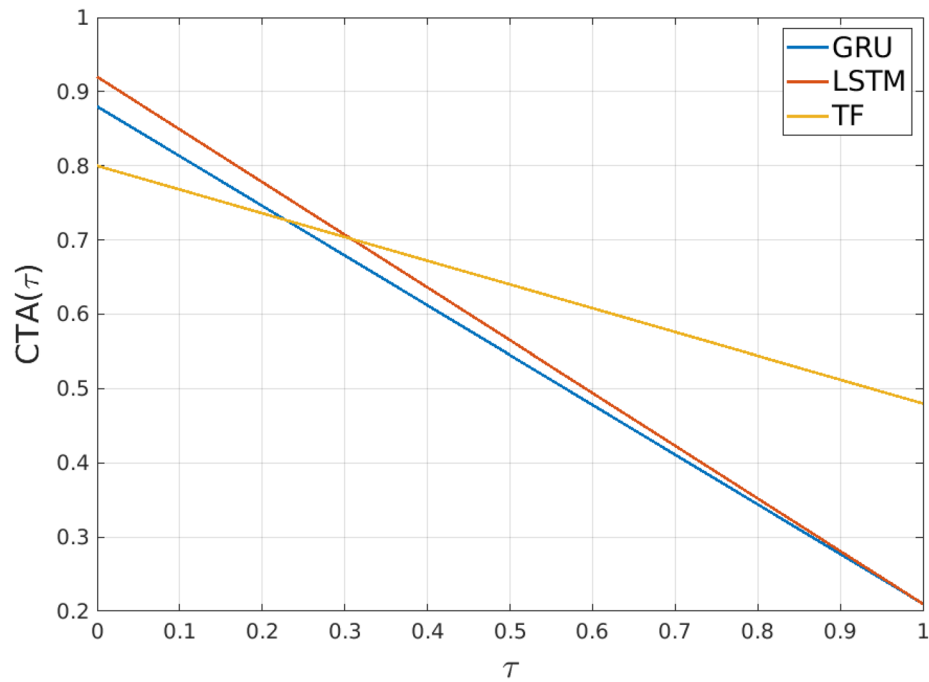
**Figure 4.** Impact of the weighting parameter $\tau$ for the composite time-weighted actions metric. Depicted are the results for GRU, LSTM and Transformer (TF) considering seven actions in the past and predicting seven future actions.
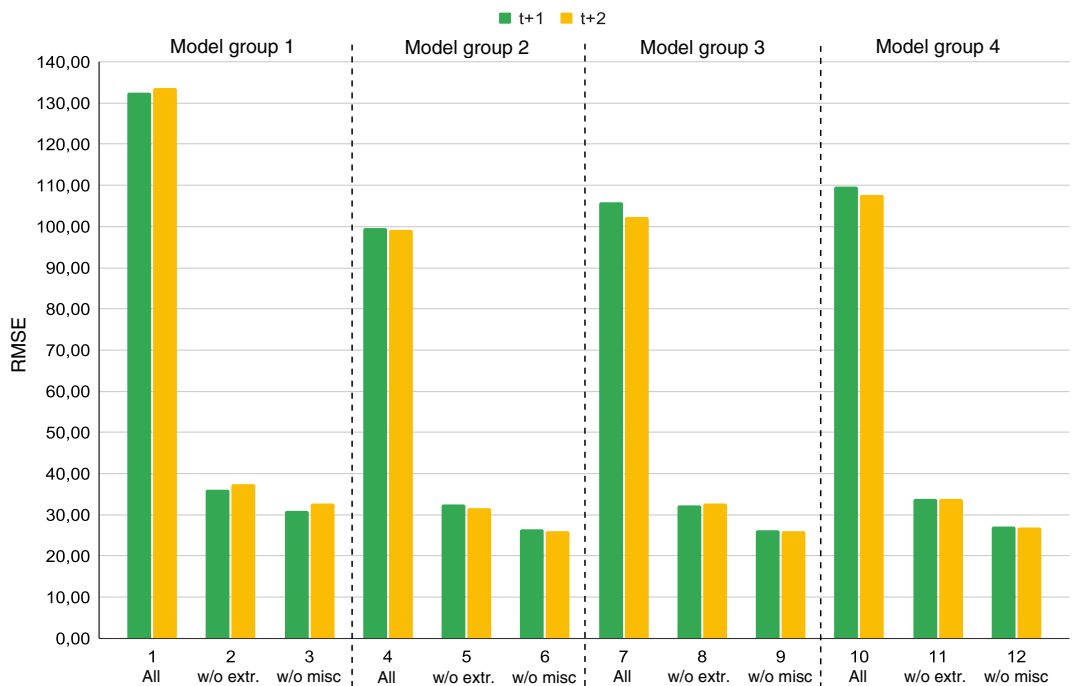


**Figure 5.** RMSE effect of model performances including misc samples (1, 4, 7, 10), samples where the extreme outlier element were removed (2, 5, 8, 11) and misc samples completely removed (3, 6, 9, 12).

training and test set. In each group the first experiment (1, 4, 7, 10) includes the misc samples. The second of each group (2, 5, 8, 11) has their extreme element in the sequence removed, effectively skipping one process steps always. The third of each group (3, 6, 9, 12) has the entire misc samples removed. The exclusion of the extreme element in the misc samples improves the model performance by a factor three to four. Since the

9

removal of an extreme element in the misc samples does not mirror the real world application we opted to remove the entire sequence and achieve in average an additional 18% model performance increase.

## Future work

The results show that our VMAS can deliver interesting insights on real world data obtained from a PDA system for car manufacturing. However, in order to measure the added value for stakeholders, the approach must be evaluated using key performance indicators. Only in this way is it possible to derive optimization processes from the results in a targeted manner. In practice, due to the extensive training time required for training seq2seq models, it makes sense to make use of components from our VMAS for a two-stage approach.

*Stage 1* In a first integration stage, the results of the automatic peak detection and source error identification are used for the automatic identification of work steps which are particularly critical based on the frequency with which faults occur. Here, however, only a superficial analysis based the proportions of errors is possible. The deep-dive into the cause-effect relationships of the errors and, thus the identification of particularly critical faults, must still be done manually.

*Stage 2* Use a trained seq2seq model to automatically identify cause-effect relationships, and investigate which source faults actually result in the most disruption times and should therefore be eliminated first. Here, a measure such as the sum over all disturbance times would be required against which the each source error can be measured, to determine how critical it is. This would allow us to create a ranking, replacing the manual analysis from phase 1, after complete integration and successful training of the ML model.

Our focus in the current paper is on sequence based approaches for identifying cause-and-effect relationships of manufacturing errors on a real world dataset. We used tried and tested methodologies for sequence-to-sequence learning; including LSTMs, GRU and the state-of-the-art Transformers. In future we also plan to work with additional sequence-to-sequence and time-series data, provided to us by Volkswagen Commercial Vehicles, basing out approach on state-of-the-art architectures such as TadGANs[26] and the Informer[27]. Finally, while addressing systematic performance improvement is outside of the scope of our current work, our methods could be used as an additional evaluation metric for optimization algorithms that aim to improve decision making in production scenarios[28].

## Conclusion

In car body production, the car body is processed according to the order requirements at interlinked production stations. Frequently, faults are detected at stations, where the resulting disturbances not only affect the station itself, but also have a negative impact on the downstream stations. To address this problem we introduce a novel vehicle manufacturing analyses system that can identify the fault cause-effect relationships, and predict future delays. The evaluation of our framework on data from the car body production of Volkswagen Commercial Vehicles shows that source and knock-on errors are surprisingly prevalent, occurring in 71.68% of action sequences. Furthermore, we show that the prediction component of our model does well at predicting the durations of up to seven actions into the future, using state-of-the-art sequence-to-sequence models, including the Transformer. Therefore deployable framework can be used to efficiently process data for identifying source and knock-on errors, as well as predicting future delays that can benefit from an early intervention.

## Data availability

The data that support the findings of this study are available from the Volkswagen Group but restrictions apply to the availability of these data, which were used under license for the current study, and so are not publicly available. Data are however available from the authors upon reasonable request and with permission of the Volkswagen Group. Please contact the corresponding author, Jeff Reimer (reimer@l3s.de) and Juergen Urdich (juergen.urdich@volkswagen.de) from Volkswagen Group.

## References

1. Chandra, R. & Chand, S. Evaluation of co-evolutionary neural network architectures for time series prediction with mobile application in finance. *Appl. Soft Comput.* **49**, 462–473 (2016).
2. Zhao, J., Liu, K., Wang, W. & Liu, Y. Adaptive fuzzy clustering based anomaly data detection in energy system of steel industry. *Inf. Sci.* **259**, 335–345 (2014).
3. Arena, F., Collotta, M., Luca, L., Ruggieri, M. & Termine, F. G. Predictive maintenance in the automotive sector: A literature review. *Math. Comput. Appl.* **27**, 2 (2022).
4. Denkena, B., Dittrich, M.-A., Fohlmeister, S., Kemp, D. & Palmer, G. Scalable cooperative multi-agent-reinforcement-learning for order-controlled on schedule manufacturing in flexible manufacturing systems. Simulation in Produktion und Logistik 2021: Erlangen, 15.–17. September 2021 305 (2021).
5. Frausto-Solís, J., Chi-Chim, M. & Sheremetov, L. Forecasting oil production time series with a population-based simulated annealing method. *Arab. J. Sci. Eng.* **40**(4), 1081–1096 (Springer, 2015).
6. Lehmann, M. & Kuhn, H. Modeling and analyzing sequence stability in flexible automotive production systems. *Flex. Serv. Manuf. J.* **32**, 366–394 (2020).
7. Huang, B., Wang, W., Ren, S., Zhong, R. Y. & Jiang, J. A proactive task dispatching method based on future bottleneck prediction for the smart factory. *Int. J. Comput. Integr. Manuf.* **32**, 278–293 (2019).
8. Ding, S., Morozov, A., Vock, S., Weyrich, M. & Janschek, K. Model-based error detection for industrial automation systems using lstm networks. In International Symposium on Model-Based Safety and Assessment, 212–226 (Springer, 2020).

9. Jiang, J.-R., Lee, J.-E. & Zeng, Y.-M. Time series multiple channel convolutional neural network with attention-based long short-term memory for predicting bearing remaining useful life. *Sensors* **20**, 166 (2020).
10. das Chagas Moura, M., Zio, E., Lins, I. D. & Droguett, E. Failure and reliability prediction by support vector machines regression of time series data. *Reliab. Eng. Syst. Saf.* **96**, 1527–1534 (2011).
11. Yadav, D. K., Chaturvedi, S. K. & Misra, R. B. Forecasting time-between-failures of software using fuzzy time series approach. In 2012 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS), 1–8, https://doi.org/10.1109/NAFIPS.2012.6291059 (2012).
12. Denkena, B., Dittrich, M.-A., Keunecke, L. & Wilmsmeier, S. Continuous modelling of machine tool failure durations for improved production scheduling. *Prod. Eng.* **14**, 207–215 (2020).
13. Oladokun, V., CHARLES, O. O. & Nwaouzru, C. An application of artificial neural network to maintenance management. *J. Ind. Eng. Int.* **2**, 19–26 (2006).
14. Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014).
15. Jerome, A., Ishii, T. & Chen, H. Forecasting and anomaly detection on application metrics using lstm. In 2018 IEEE International Conference on Big Data (Big Data), 2221–2227 (IEEE, 2018).
16. Malhotra, P., Vig, L., Shroff, G. & Agarwal, P. Long short term memory networks for anomaly detection in time series. *Proceedings* **89**, 89–94 (2015).
17. Vaswani, A. *et al.* Attention is all you need. In Advances in neural information processing systems, 5998–6008 (2017).
18. Parisotto, E. *et al.* Stabilizing transformers for reinforcement learning. In International Conference on Machine Learning, 7487–7498 (PMLR, 2020).
19. Nam, K. H., Song, Y. J. & Yun, I. D. Sss-ae: Anomaly detection using self-attention based sequence-to-sequence auto-encoder in smd assembly machine sound. *IEEE Access* **9**, 131191–131202. https://doi.org/10.1109/ACCESS.2021.3112545 (2021).
20. Li, J. *et al.* Dtdr-alstm: Extracting dynamic time-delays to reconstruct multivariate data for improving attention-based lstm industrial time series prediction models. *Knowl.-Based Syst.* **211**, 106508. https://doi.org/10.1016/j.knosys.2020.106508 (2021).
21. Li, X., Krivtsov, V. & Arora, K. Attention-based deep survival model for time series data. *Reliab. Eng. Syst. Saf.* **217**, 108033. https://doi.org/10.1016/j.ress.2021.108033 (2022).
22. Yuan, Y. *et al.* Using an attention-based lstm encoder-decoder network for near real-time disturbance detection. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **13**, 1819–1832. https://doi.org/10.1109/JSTARS.2020.2988324 (2020).
23. Chapman, P. *et al.* The crisp-dm user guide. In 4th CRISP-DM SIG Workshop in Brussels in March, vol. 1999 (sn, 1999).
24. Mishra, S. & Datta-Gupta, A. Chapter 3 - distributions and models thereof. In Mishra, S. & Datta-Gupta, A. (eds.) Applied Statistical Modeling and Data Analytics, 31–67, https://doi.org/10.1016/B978-0-12-803279-4.00003-1 (Elsevier, 2018).
25. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
26. Geiger, A., Liu, D., Alnegheimish, S., Cuesta-Infante, A. & Veeramachaneni, K. Tadgan: Time series anomaly detection using generative adversarial networks. In 2020 IEEE International Conference on Big Data (Big Data), 33–43 (IEEE, 2020).
27. Zhou, H. *et al.* Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proc. AAAI Conf. Artif. Intell.* **35**, 11106–11115 (2021).
28. Magnanini, M. C., Terkaj, W. & Tolio, T. Robust optimization of manufacturing systems flexibility. *Procedia CIRP* **96**, 63–68 (2021).

## Acknowledgements

## Author contributions

All authors provided critical feedback and helped shape the research. Y.W. and J.R. contributed to the development and evaluation of the proposed methodology. Y.W. and S.L. contributed towards the development of the evaluation metrics. J.U. provided invaluable domain knowledge and helped formulate the research problem. J.R., S.W., and G.P. equally conceived the original idea and experimental design. S.W. and G.P. coordinated and supervised the project. All authors reviewed the manuscript.

## Additional information

**Correspondence** and requests for materials should be addressed to J.R.

**Reprints and permissions information** is available at www.nature.com/reprints.

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.