ORIGINAL ARTICLE
Journal Section

Understanding the importance of process alarms based on the analysis of deep recurrent neural networks trained for fault isolation

Gyula Dorgo | Peter Pigler | Janos Abonyi PhD

MTA-PE Lendület Complex Systems Monitoring Research Group, Department of Process Engineering, University of Pannonia

[1]MTA-PE Lendület Complex Systems Monitoring Research Group, Department of Process Engineering, University of Pannonia

Correspondence
Department of Process Engineering, University of Pannonia, Egyetem str, 10. POB 158, Veszprém, H-8200, Hungary
Email: janos@abonyilab.com

Funding information
National Research, Development and Innovation Office – NKFIH, OTKA – 116674

The identification of process faults is a complex and challenging task due to the high amount of alarms and warnings of control systems. To extract information about the relationships between these discrete events, we utilise multi-temporal sequences of alarm and warning signals as inputs of a recurrent neural network (RNN) based classifier and visualise the network by principal component analysis. The similarity of the events and their applicability in fault isolation can be evaluated based on the linear embedding layer of the network, which maps the input signals into a continuous-valued vector space. The method is demonstrated in a simulated vinyl acetate production technology. The results illustrate that with the application of RNN based sequence learning not only accurate fault classification solutions can be developed, but the visualisation of the model can give useful hints for hazard analysis.

KEYWORDS
Fault classification, Deep learning, Visualization of discrete events

1 | INTRODUCTION

Chemometric models are widely applied to fault detection and isolation of chemical processes [27]. Although most of these models utilise continuous multivariate data, plant operators are required to make decisions based on hundreds of discrete data generated by the control systems as warning and alarm signals. In a complex system faults may co-occur in several states, [15], and can generate long sequences of warning and alarm signals, therefore giving effective responses to abnormal situations is a challenging task for even the well-trained operators. Intelligent fault diagnostics, therefore, requires the analysis of temporal relationships of discrete events. To meet this requirement, we propose a sequence-based fault classification algorithm that utilises the high amount of unexploited event type data of alarm systems.

Discrete event-based fault diagnosis is an important area of research, as this type of information like alarms and warnings frequently occur in the process industry. According to the Engineering Equipment and Materials User's Association (EEMUA), the purpose of an alarm system is to redirect the operator's attention towards plant conditions requiring timely assessment or action [4]. Therefore, a properly designed and operated alarm system helps the operator to keep the processes in the normal operation range by indicating the presence of abnormal situations. Blanke *et al.* give an extensive overview of fault diagnosis [1], while Zaytoon *et al.* focuses on the diagnosis methods of discrete event systems [29]. The central concepts of the diagnosability and fault diagnosis of discrete event systems were defined by Sampath *et al.* [23], [24].

When we build data-driven models for fault detection and isolation purposes, we not only focus on the prediction accuracy, but we also would like to understand the mechanism of the faults by unfolding the relationships between the faults and the process-variables [7]. When events of different states occur at the same time, they can be visualised with the use of a time series cross-sectional data matrix, as it is presented by Chen *et al.* [3]. Correlated events can be visualised with the utilization of a Hinton diagram of joint distributions [31]. The recently developed high-density alarm plot (HDAP) chart highlights top alarms over a given period, and the alarm similarity color map (ASCM) explores the related and redundant alarms [14]. The methods are used for the detection of correlated alarms in ref. [11], while the application of ASCM and correlation colour maps were also reported in ref. [28]. From the tools of chemometrics, dendrograms were employed in [28] and [3].

Our key idea is to develop a supervised visualisation algorithm to evaluate the similarities of the alarms from the viewpoint of the faults. We assume that, as in the case of natural language processing applications of deep learning, the visualisation of the network will support the understanding of the long- and short-term dependencies of the alarm signals and the faults. To test the proposed methodology, we have built a sequential data based classifier applying deep learning solutions.

The complexity of the problems and size of the available datasets tend to be bigger and bigger, resulting in the increased application of deep learning solutions in engineering [12], chemistry [17], computational biology [26], process engineering [22], machine health monitoring [30], anomaly detection [19] and fault detection and isolation [20], [18]. For a detailed description of artificial neural network-based approaches including the distinguish of classical (shallow) neural networks, and deep learning solutions see ref. [25]. From the wide range of models, we apply recurrent neural networks (RNNs) [10] using long short-term memory (LSTM) units [9] . Our model uses an embedding layer, a layer with linear transformations, to map the one-hot encoded events into a continuous-valued vector space. Using such embedding is a state-of-the-art approach to sentiment analysis of texts. The main benefit of this linear mapping is that the analysis of the vector space can be used to study the contextual meaning of the words [16], [21]. Based on this analogy, we assume that similar warnings and alarm signals will be close to each other in this embedding space [6]. We apply a linear embedding, assuming that the weights of the similar events will be correlated, so principal component

analysis (PCA) can be used to visualise the hidden structure of the events and evaluate the significance of these signals.

To demonstrate the applicability of the proposed approach, we extended the existing simulator of a vinyl acetate production technology [2] by implementing 11 different malfunctions to the process to generate easily reproducible results and stimulate the development of fault diagnosis and event prediction algorithms. The 11 malfunctions were chosen based on process relevant knowledge, as we wanted to implement malfunctions with a significant effect on the operation in various locations of the process. Using this simulator, we can record the dynamic characteristic following these faults and we can create the log file of the occurring alarms and warnings.

The roadmap of the paper is as follows. In Section 2.1, we define the input of the classifiers as sequences of the temporal relationships of the events. Section 2.2 presents the classification task, and Section 2.3 describes why we analyse the embedding layer of the model. Although mainly the prediction accuracy is in the focus of the application of deep learning models, we study the applicability of PCA to extract information related to the hidden structure of the problem in Section 2.3. We introduce the case study in Section 3.1. The results are discussed in Sections 3.2 -3.4.

2 | FAULT CLASSIFICATION AND VISUALISATION OF PROCESS ALARMS

2.1 | Formulation of the event sequence based fault classifier

Our key idea is that the sequences of process alarms and warnings contain enough information about the technology to serve as an input of a classifier designed to estimate the $y_k = \{c_1, \ldots, c_{n_c}\}$ class label of the faults, where $k = 1, \ldots, N$ and represents the currently examined sequence of the process.

$$\hat{y}_k = f(\mathbf{\Phi}_k) \tag{1}$$

We define the $\mathbf{\Phi}_k$ sequences based on *states* represented by $s =< pv, a >$ data couples, where pv is the index of the process variable and a represents the related state signals, such as $a \in \{Low\ Alarm, Low\ Warning, Target\ Range, High\ Warning, High\ Alarm\}$, e.g.: $s_e :=< Column\ Top\ Temperature, High\ Alarm >$. An *event* can be represented by a triplet, such as $e =< s, st, et >$ defining which s state is taking place in a time interval between the st starting time and the et ending time, e.g.: $< s_e, 12, 14 >$. A visual illustration of the discrete events and their possible temporal relationships can be seen in Figure 1 and the example event log database formed from these events can be seen in Table 1. A temporal database contains the event identification number (event id), the identification number of the occurring state (state id), and the starting and ending time of the given event (in arbitrary, but unified unit of time).

We are looking for sequences, which can reflect temporal relationships among the states and the effect of faults. Therefore we define a *time window* in which we assume causal dependency. For the description of the type of the temporal relationship between the sequence elements, we connect the states by one of the following four temporal predicates:

- If $st_1 = st_2$ and $et_1 = et_2$, then $e_1\ equal\ e_2$,
- If $0 \leq st_2 - et_1 \leq window$, then $e_1\ before\ e_2$
- If $st_2 < st_1 < et_1 \leq et_2$ or $st_2 \leq st_1 < et_1 < et_2$, then $e_1\ during\ e_2$
- If $st_1 \leq st_2 < et_1 < et_2$ or $st_1 < st_2 < et_1 \leq et_2$, then $e_1\ overlap\ e_2$

For simplicity, we will use the notation E, B, D and O for $equal$, $before$, $during$ and $overlap$ respectively. Using these symbols, we can define temporal instances as $\phi := e_1 \overset{R}{\Rightarrow} e_2$, where $R \in \{E, B, D, O\}$ is a temporal predicate.

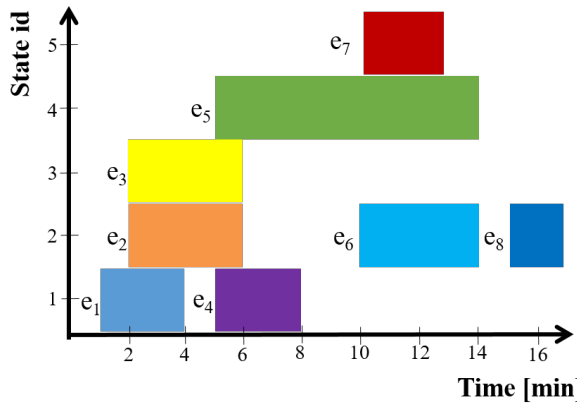

FIGURE 1 The visual illustration of discrete events in an example event log database. The horizontal axis represents the time, while the vertical axis illustrates the different states of a process (e.g. $s_e := < Column\ Top\ Temperature, High\ Alarm >$). We should note that a state can occur in multiple events.

TABLE 1 Example for the event log database representing an event with its id, the occurring state in the specific event and the starting and ending time of the event.

Event id	State id	Starting time [min]	Ending time [min]
e_1	s_1	1	4
e_2	s_2	2	6
e_3	s_3	2	6
e_4	s_1	5	8
e_5	s_4	5	14
e_6	s_2	10	14
e_7	s_5	10	13
e_8	s_2	15	17

For example, based on the event log database of Table 1, the following temporal instances can be defined: $e_1 \overset{O}{\Rightarrow} e_2, e_2 \overset{E}{\Rightarrow} e_3, e_4 \overset{D}{\Rightarrow} e_5, e_5 \overset{B}{\Rightarrow} e_8$. Similarly, longer sequences can be generated, e.g.: $\Phi := e_2 \overset{E}{\Rightarrow} e_3 \overset{O}{\Rightarrow} e_5 \overset{B}{\Rightarrow} e_8$. The sequences generated from the event log database are stored in a sequence database.

To utilise the sequences of the symbols as inputs of the neural network, we encode the symbols of the states $S = \{s_1, s_2, ..., s_{n_S}\}$ and the temporal predicates R={E, B, D, O}, into vectors of numerical values. From a technical view, first we encode every element in the sequences (event and temporal predicates as well) to a numerical form, and this numerical form is transformed to the sequence of one-hot encoded vectors.

The one-hot encoding is based on binary vectors, $\mathbf{oh}_k^t$ where only one bit related to the encoded signal is fired among the $n_o = n_S + n_R$ bits, where n_S represents the number of states and n_R stands for the number of different temporal predicates.

The embedding layer is a linear transformation, which transforms the one-hot encoded vector of each sequence element into a vector with the specified dimension (n_e) with continuous values.

$$\mathbf{x}_k^t = \mathbf{W}\,\mathbf{oh}_k^t \tag{2}$$

Figure 2 gives a simple, didactic example of the formulation of the input dataset of the embedding layer. Suppose that the k-th analysed sequence is the simple overlapping relationship of the events e_1 and e_2 illustrated in Figure 1. The states represented in these events and the overlapping temporal relationship are transformed into a one-hot encoded vector with the dimension of $n_o = 2 + 1 = 3$, supposing we have no other states or temporal predicates in our examined dataset, therefore $n_S = 2$ and $n_R = 1$. Each of these elements will fire one specific bit of the one-hot encoded vector forming $\mathbf{oh}_k^1$, $\mathbf{oh}_k^2$ and $\mathbf{oh}_k^3$ respectively. We transform these one-hot encoded vectors of each sequence element to vectors of continuous values with a specified dimension (n_e, the embedding dimension) in the embedding layer. These vectors form the input variables of the deep learning layer.

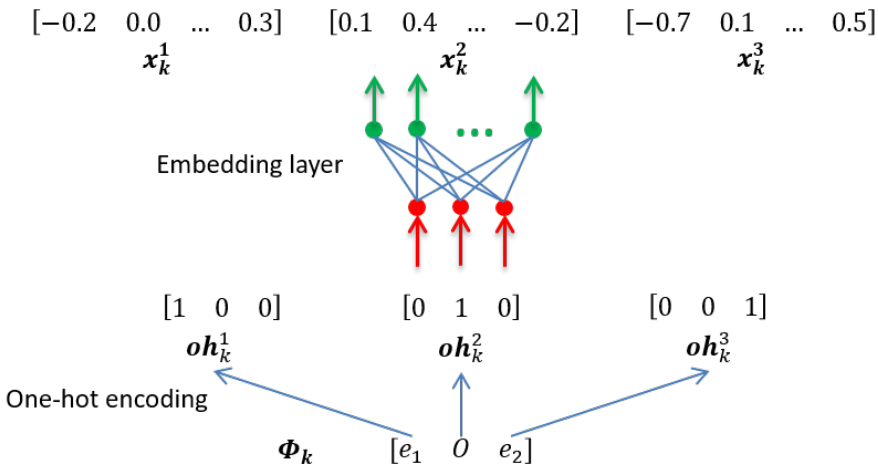

FIGURE 2 A simple and didactic example for the formulation of the input variables of the deep learning layer. The elements of the $\mathbf{\Phi}_k$ sequence are transformed to the form of one-hot encoded vectors, with the dimension of the number of the type of sequence elements in the examined database. We transform these one-hot encoded vectors to vectors of continuous values with the help of the embedding layer.

Therefore the deep learning based classifier can be formulated as

$$\hat{y}_k = f\,(\mathbf{x}_k) \tag{3}$$

where $\mathbf{x}_k$ represents the sequence of $\mathbf{x}_k^i$ ($i = 1 \ldots T$) vectors, the continuous-valued representations of the sequence elements (events and their temporal predicates) calculated by the linear embedding layer and $\hat{y}_k$ stands for the estimated class label.

We train a neural classifier to calculate the $P(y_k = c_j|\mathbf{x}_k)$ fault probabilities and assign the class label that has the highest probability (see Figure 3).

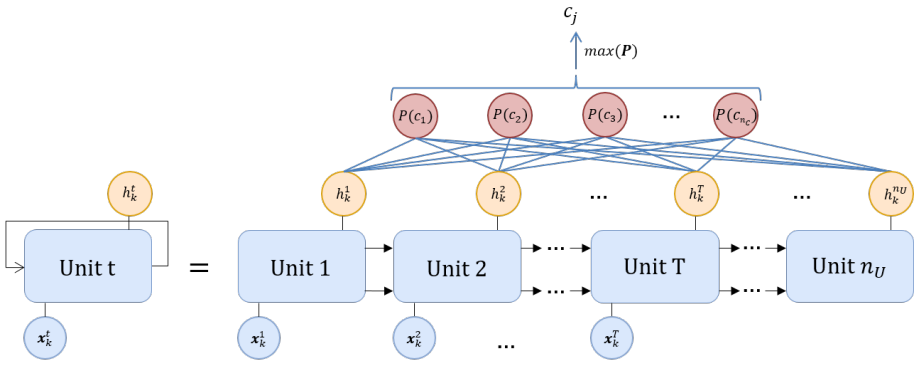

FIGURE 3 The structure of the long short-term memory unit-based neural classifier. The input variables of the layer of the long short-term memory units, represented by the $\mathbf{x}_k^i$ $(i = 1...T)$ vectors, are the continuous-valued representations of the sequence elements calculated by the linear embedding layer. We transform the output activity values of this long short-term memory (LSTM) unit-layer ($\mathbf{h}_k$) to probability values by a linear layer, and we accept the class with the highest probability as the root cause of the sequence.

$$\hat{y}_k = \max_j P(y_k = c_j | \mathbf{x}_k) \tag{4}$$

The recurrent neural network maps the $\mathbf{x}_k$ input sequence into a $\mathbf{h}_k$ sequence of real values. This mapped sequence of "hidden variables" is represented as a $\mathbf{h}_k$ vector of the activities of the LSTM units, $\mathbf{h}_k = \left[h_k^1, \ldots, h_k^{n_U}\right]$ and used to calculate the fault probabilities by the softmax activation function (where n_U is the number of LSTM units),

$$P(y_k = c_j | \mathbf{x}_k) = P(y_k = c_j | \mathbf{h}_k) = \frac{exp\left((h_k)^T \mathbf{w}_{s,j} + b_j\right)}{\sum_{j=1}^{N_c} exp\left((\mathbf{h}_k)^T \mathbf{w}_{s,j} + b_j\right)} \tag{5}$$

where $\mathbf{w}_{s,j}$ represents the j-th column vector of the $\mathbf{W}_s$ weight matrix of the output layer of the network and b_j represents the bias. The denominator of Equation 5 is the sum of the elements with different j indices, therefore constant and the resulting class label is proportional to the nominator.

$$\hat{y}_k = \max_j P(y_k = c_j | \mathbf{x}_k) \propto \max_j \left(exp\left((h_k)^T \mathbf{w}_{s,j} + b_j\right)\right) \tag{6}$$

Figure 3 highlights that the sequences are processed by a deep recurrent neural network, where the network is unrolled n_U times and each of the first T units process a single sequence element represented as a continuous-valued vector with dimension n_e. Although, the units from T to n_U seem to have no significant effect on the classification task, the core principle of the LSTM units is the handling of long-term dependencies and these added units can improve the accuracy of the classification.

In the following subsection, we present how the hidden layer ensures the efficient calculation of the $\mathbf{h}_k$ vector that is informative to build the classifier based on sequential data.

2.2 | Recurrent neural network layer with long short-term memory units

Recurrent neural networks (RNNs) are designed to capture time-dependency in sequential data [10]. These models were popular after the introduction of RNNs with LSTM (long short-term memory) units [9], proposed to overcome the difficulties of handling long-term dependency and vanishing gradient [8].

The LSTM layer maps the $\mathbf{x}_k$ sequence into $\mathbf{h}_k$, the vector of the activities, so it has equal or more units than the sum of the number of events and the corresponding temporal predicates between them in the input sequence. Therefore, sequences containing more events than the specified T value are truncated, and the length of the input sequence is $2 \times T - 1$ (the length of a vector is equal to the number of elements in it, here T, and we need to count the corresponding temporal predicates as well). This is due to the fact that a simple sequence $e_1 \overset{R}{\Rightarrow} e_2$ containing only two process states, is consist of $2 \times 2 - 1 = 3$ sequence elements including the temporal relationship. Similarly, the sequence of $e_1 \overset{R}{\Rightarrow} e_2 \overset{R}{\Rightarrow} e_3$ contains only three process states but the number of elements in the sequence is equal to $2 \times 3 - 1 = 5$ pieces. Therefore the "-1" part is implemented to take into consideration the missing temporal predicate after the last process state of the sequence.

Figure 4 shows the structure of the LSTM unit. The key feature of the model is that all LSTM units have a cell state (C_k^t) [5] that can be used to forward information to the next unit. Therefore, an LSTM unit interacts with its neighbouring cells by gates by either adding or removing information to this memory flow.

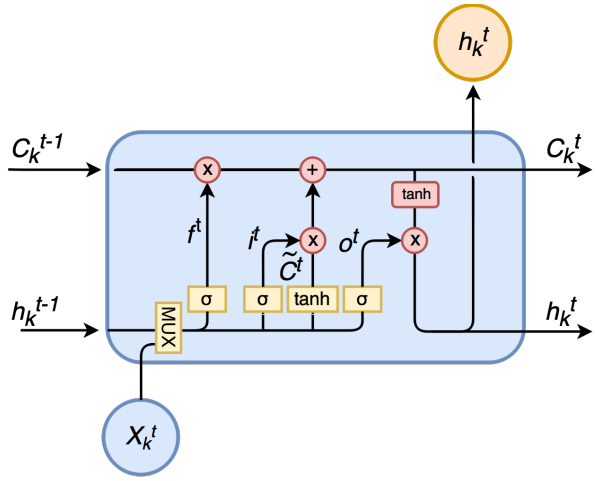

FIGURE 4 Structure of a single long short-term memory unit. The input sequence element ($\mathbf{x}_k^t$), and the activity (h_k^{t-1}) and cell state (C_k^{t-1}) values of the previous long short-term memory unit are modified with use of the sigmoid (σ) and hyperbolic tangent ($tanh$) functions of the forget (f_k^t), input (i_k^t) and output (o_k^t) gates. The calculated activity (h_k^t) and cell state (C_k^t) values are transfered to the next long short-term memory unit, while the activity value is directly output as well. The MUX box in the figure indicates the forming of one signal from the h_k^{t-1} and $\mathbf{x}_k^t$ expressions as in $[h_k^{t-1}, \mathbf{x}_k^t]$.

The LSTM unit receives the activation of the previous cell h_k^{t-1}, the $\mathbf{x}_k^t$ input that relates to the tth element of the sequence, and the C_k^{t-1} cell state of the previous unit. The forget gate f_k^t determines how much information of the earlier units should be kept.

$$f_k^t = \sigma(\mathbf{W}_f[h_k^{t-1}, \mathbf{x}_k^t] + b_f) \tag{7}$$

b represents the bias vector of the neurons (and in all of the following equations as well), and σ represents the applied sigmoid function.

The state and the activity of the unit will be updated using the $\mathbf{x}_k^t$ and the preceding cell activation h_k^{t-1}. This is realised with the sigmoid function of the input gate. The whole process is illustrated in Figure 4.

$$i_k^t = \sigma(\mathbf{W}_i[h_k^{t-1}, \mathbf{x}_k^t] + b_i) \tag{8}$$

$$\tilde{C}_k^t = \tanh(\mathbf{W}_c[h_k^{t-1}, \mathbf{x}_k^t] + b_c) \tag{9}$$

The LSTM unit updates its old cell-state C_k^{t-1} using the forget-gate f_k^t and the filtered input gate i_k^t:

$$C_k^t = f_k^t C_k^{t-1} + i_k^t \tilde{C}_k^t \tag{10}$$

The activity of the LSTM unit is calculated based on the cell-state and the output gate signals:

$$o_k^t = \sigma(\mathbf{W}_o[h_k^{t-1}, \mathbf{x}_k^t] + b_o) \tag{11}$$

$$h_k^t = o_k^t \tanh(C_k^t) \tag{12}$$

2.3 | Embedding layer based analysis of the alarms and the analysis of node activities

Our work is mainly motivated by the purpose to analyse the similarities of alarm and warning signals. Therefore we introduce an embedding layer to the network, as the concept of the embedding layer is to map the sequence elements into a continuous vector space, which we apply for the visualisation of the events. First, we transform every character of the sequence into one-hot encoded vectors. Then with the linear transformation of the embedding layer, we transform these one-hot encoded vectors of each sequence element into a vector of continuous values as it is presented in Equation 2. According to the presented equation, this linear transformation means a simple matrix multiplication in practice, where the dimension of the resulted vector is n_e, the embedding dimension (chosen according to preferences, see Figure 10) with continuous values between -1 and 1.

The weights of the embedding layer are trained simultaneously with the LSTM units, therefore the resulted weight matrix, $\mathbf{W}$, stores information related to the contextual connection between the symbols and the classification problem.

$\mathbf{W}$ is multiplied by one-hot vectors; therefore every row of the matrix represents a given symbol, $\mathbf{w}_i$. The similarities (sim) of the alarms (s_i, s_j) can be evaluated based on the Euclidean distances of these vectors ($d\ (\mathbf{w}_i, \mathbf{w}_j)$):

$$sim(s_i, s_j) = 1 - \frac{d\ (\mathbf{w}_i, \mathbf{w}_j)}{d_{max}} \tag{13}$$

where d_{max} represents the maximum Euclidean distance between the rows of the $\mathbf{W}$ matrix and $\mathbf{w}_i$ and $\mathbf{w}_j$ represents the i-th and j-th rows of the $\mathbf{W}$ matrix respectively.

We use the resulted pairwise similarities to generate a dendrogram to form clusters of the alarm signals.

When the dimensionality of the embedding layer is two, it can be directly used to visualise the relative positions of the alarms. When $n_e > 2$ we can use one of the several data visualisation techniques of chemometrics. In the present work, we utilised principal component analysis (PCA) since it can also visualise the correlation of the columns of the **W** matrix, which information is useful for the selection of the proper size of the embedding dimension.

The output layer of the network is based on the linear combination of node activities as it was described in Equations 5-6.

To get an insight into the classification problem we can perform a principal component analysis of the $\mathbf{H} = [h_1, \ldots, h_N]^T$ matrix of the activities, where, $\mathbf{h}_k = \left[h_k^1, \ldots, h_k^{n_U}\right]$, $k = 1, \ldots, N$ and map the sequences into a two dimensional space. When these sequences are labeled, the similarity of the faults can be revealed from the resulted plot, similarly to Figure 15.

3 | CASE STUDY

We developed a well-documented simulation study by the extension of an existing VAc technology simulator to provide a reproducible benchmark problem of alarm based fault classification. In the following, we will describe the technology, the simulator and the log files. The definition of the fault classification problem will be followed by the details of the RNN and the experiments for the determination of the optimal model structure. Finally, based on the resulting model we visualise the alarms and the faults by the principal component analysis of the embedding layer and the node activities. The discussion of the results will illustrate that the information generated by the proposed methodology can be useful in the risk analysis of complex processes.

3.1 | Fault classification problem of the vinyl acetate process

The used dynamic simulator of the vinyl acetate (VAc) process contains 27 controlled and 26 manipulated variables, therefore it is complex enough to define fault classification problems [2]. The process contains 10 basic unit operations and seven chemical components (ethylene (C_2H_4), oxygen (O_2) and acetic acid (HAc, CH_3COOH) are converted to vinyl acetate (CH_2=$CHOCOCH_3$) with water (H_2O) and carbon dioxide (CO_2) byproducts, ethane (C_2H_6) enters with the ethylene feed as inert) (see Figure 5).

The vaporiser is implemented as a well-mixed unit with seven components, with a gas input containing the mixture of the fresh C_2H_4 stream and the absorber vapour effluent, and a liquid input from the HAc tank. The catalytic plug flow reactor is implemented as a distributed system with ten elements in the axial direction. Inside of the reactor the exothermic reactions of Equation 14 and 15 take place.

$$C_2H_4 + CH_3COOH + 1/2\, O_2 \rightarrow CH_2 = CHOCOCH_3 + H_2O \tag{14}$$

$$C_2H_4 + 3\, O_2 \rightarrow 2\, CO_2 + 2\, H_2O \tag{15}$$

The process contains a feed-effluent heat exchanger (FEHE), where a small time constant is added to the exit temperature sensors to simulate the dynamics of the process. After a pressure letdown valve (which is not shown in Figure 5), the effluent is led to a separator. The separator is modeled as a partial condenser, and the leaving liquid and

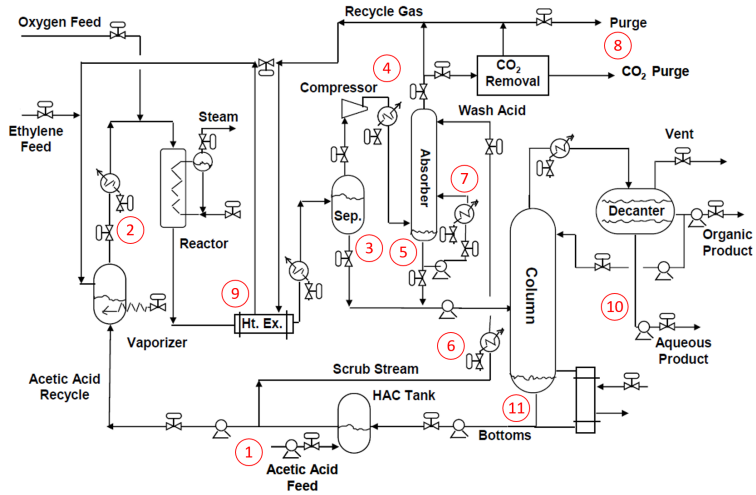

FIGURE 5 Flow chart of the vinyl acetate production technology (The numbers in circle (red) show the type of the implemented fault)

gas stream flow rates are calculated with a steady state equilibrium-flash equation. The gas stream enters to the bottom part of the absorber unit after compression. The absorber is divided into two parts. The liquid stream entering the bottom part is the liquid stream leaving the top part and a circulation stream. The gas inlet of the top part is leaving from the bottom of the absorber, while the liquid inlet comes from the HAc tank. There is a CO_2 removal system implemented after the absorber. A gas removal system takes place before the azeotropic distillation tower to remove all the light components from the inlet of the tower, which comes from the bottoms of the separator and absorber units. The gas removal system is modeled as an ideal component separator, which completely separates the gas components (O_2, CO_2, C_2H_4, C_2H_6) to send them back to the inlet of the compressor, while the liquid stream (VAc, H_2O, HAc) enters the distillation tower. The column is highly nonlinear, with 20 theoretical stages, whose liquid holdup can vary. After the condenser, a decanter is implemented for the separation of the liquid phases. The liquid recirculation stream and the HAc inlet stream are mixed in the HAc tank.

The original MATLAB model of the simulator (published in ref. [2]) contained five disturbances (1.) step change in the composition of C_2H_6 in the fresh C_2H_4 feed stream from 0.001 to 0.003 mole fraction, 2.) loss of column feed for 5 minutes, 3.) loss of fresh HAc feed stream for 5 minutes, 4.) loss of fresh O_2 feed stream, 5.) an analyzer is off-line (except the O_2 analyzer)), while Károly and Abonyi [13] studied the effects of three malfunctions (1.) Loss of *HAc* feed, 2.) Loss of O_2 feed and 3.) Loss of column feed), and studied the effect of product changes with the use of the so-called Operating State Matrix (OSM), containing the randomly generated values of the following manipulated variables:

- Operating state start time (min)
- Operating state end time (min)
- Setpoint of the reactor output temperature (150-165 °C)
- H_2O composition in the column's bottom (9 - 18%)
- Vaporizer feed (2.2 - 2.6 $\frac{kmol}{min}$)
- Change of the C_2H_6 concentration of the C_2H_4 feed from 0.1% to 0.3% (not range based, only two states)

TABLE 2 The manipulator values in the case of the 11 implemented faults (measurement units are neglected as the original code and documentation does not contain them neither)

Tag of fault	Controlled variable	Manipulated variable	Man. variable value in case of fault
1	HAc Tank Level	HAc fresh feed Flow Rate	0.3
2	Heater Exit Temp	Reactor Preheater Heat Flow	2000
3	Separator Level	Separator Liquid Exit Flow Rate	0
4	Compressor Exit Temp.	Compressor Heater Heat Flow	20000
5	Absorber Level	Absorber Liquid Exit Flow Rate	0
6	Circulation Stream Temp.	Absorber Scrub Heat Flow	5000
7	Scrub Stream Temp.	Circulation Cooler Heat Flow	1000
8	C_2H_6 in the Gas Recycle	Purge Flow Rate	0
9	FEHE Hot Exit Temp.	Bypass Flow Rate	0.4
10	Decanter Aqueous Level	Aqueous Product Flowrate	0
11	Coloumn Bottom Level	Coloumn Bottom Exit Flowrate	0

Modern chemical plants often have historical log files of incoming alarms and warnings, and these log files can be structured to build training data for fault classifiers. Historical process data can be enriched or replaced by events generated by simulators since most of the advanced process technologies are also supported by operator training systems or other model-based solutions.

To test our methodology, we generated a database of 200 different event sequences. The inserted faults were related to the malfunction of the controller or manipulator, as the manipulated value of the process variable remained constant for a specified time. Table 2 shows the values of the manipulators in case of faults.

A 100-minute time window was used to utilise events that we consider as direct consequences of the malfunction. The threshold values of normal operating conditions for each measured process variables were determined based on the analysis of normal operation.

To illustrate the information content of discrete events we generated three different datasets and tested their applicability in the solution of the classification problem.

- *Dataset A*: Signals related to low and high alarms (two states / process variable)
- *Dataset B*: Signals related to low and high alarms and warnings (four states / process variable)
- *Dataset C*: Identical to B, but the normal operation is also defined as an event (five states / process variable)

To examine the information content of temporal relationships between events we generated two cases: (1) in the first we include the temporal relationships and (2) in the second we neglect them. Therefore we generated six datasets labeled as *Dataset A/1*, ..., *Dataset C/2* respectively.

The horizontal axis of Figure 6 shows the number of sequences that can follow the given malfunction, while the vertical axis shows the maximal length of these sequences. Figure 6 illustrates the faults based on the length of the

longest sequence that follows the given fault and as the number of different sequences that can follow the given fault (the size of the sequence is equal to the number of events in it, we do not count the temporal relationships). The core principle of the figure is that faults with fewer types and shorter length of characteristic sequences can usually be treated as trivial by the process experts. Considering this the more complicated sequences are in the upper right part of the graphs. The information content of the databases increases from *Dataset A* to *Dataset C*, as the number and size of the sequences show an increasing trend.

The numbering of the different alarm tags follows an ascending order from 1 to 27 according to Table 3. During the generation of discrete events, we added an extra digit to the end of each variable tag indicating the type of the event happened on that given variable as it can be seen in Table 4. Of course from *Dataset A* the events of warnings and the event of the target operating range are missing (therefore the last digit cannot be 2, 3 or 4), while from *Dataset B* the event of the target operating range is missing (consequently the last digit cannot be 3).

Therefore Table 2 shows the causes why the different events on the variables presented in Table 3 occur, as events on the controlled variables can be considered as the effects of malfunctions.

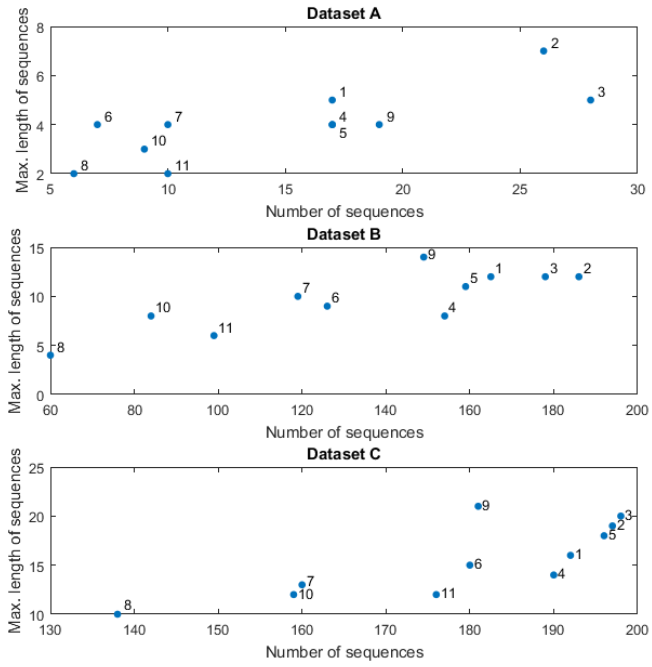

FIGURE 6 The characterisation of faults based on the number and maximal length of the following sequences. The identification and classification of malfunctions that generate longer and more type of sequences of events are most likely to be problematic. The length of the sequence is equal to the number of events in it, and we do not count the temporal predicates.

3.2 | Application of the proposed recurrent neural network for fault classification

The implementation of the simulator and the data preprocessing was carried out in MATLAB environment. The implementation of the structure of the deep neural network and the training of it was carried out in Python applying Keras and using Tensorflow as backend. We trained the model using a Nvidia GeForce GTX 1060 6GB GPU with the application of CUDA. During the testing of the different model structures 7-fold cross-validation was applied and evaluated, the number of epochs was set to 500, with 512 as batch size.

3.2.1 | The applicability of different datasets

Figure 6 highlights the information content of datasets. The number of sequences that characterise a given fault, their length, and the presence of temporal relationships can all influence the effectiveness of the proposed LSTM network. To determine the most appropriate set of symbols, we tested the efficiency of networks under uniform conditions, with 11 LSTM units and five events in a sequence (longer sequences are truncated) and with four as the dimension of embedding. Firstly, we used datasets including temporal predicates. According to Figure 7, *Dataset B/1* is the most applicable for further investigations, with approximately 91.2 % of average accuracy (correct classification rate). We can conclude that the incorporation of warnings can improve the effectiveness of the proposed methodology, but the normal operating range as an event showed a decreased correct classification rate. We proved the effect of different datasets by statistical variance analysis (one-way ANOVA) and found a very low significance value ($p = 2.2E-05$), therefore we neglect the null hypothesis, that the type of the dataset has no significant effect on the correct classification rate.

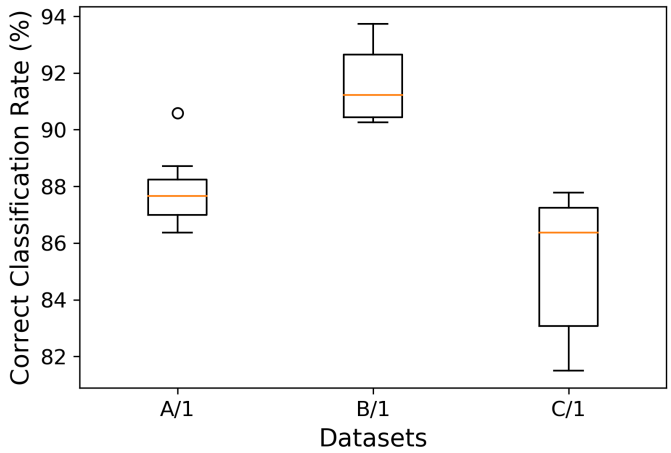

FIGURE 7 The effect of the information content of datasets on the effectiveness of the proposed network with 11 long short-term memory units, five events in a sequence and four as embedding dimension, datasets with temporal predicates served as the basis of analysis. According to the results, *Dataset B/1* is the most applicable for further investigations, as it shows the highest correct classification rate. Therefore the characterisation of the variables with alarm and warning signals showed improved accuracy comparing to the result of *Dataset A/1*, with only the alarm signals, however the including of target operating ranges as events showed decreased correct classification rate.

We studied the effect of the number of events (i.e. the length of the sequence (T), the events after T are truncated)

and the incorporation of temporal predicates for *Dataset B*, which showed the highest correct classification rate in the previous analysis. According to Figure 8, the incorporation of more than three events indicated no improvement in the performance of the classifier, and the temporal predicates do not significantly influence the results. We applied two-way ANOVA for the determination of differences in performance, but the analysis indicated that the including of temporal predicates or the application of more events does not result in better performance, as can be seen in Table 5. However standard deviations are not within 5%, in the case of the number of events it is very close to it. The good correct classification rate after only a few events shows that a well-trained neural classifier can classify a fault after only a few alarms, suggesting promising industrial application possibilities in the future. To reduce model complexity we applied four events and the dataset without temporal predicates in the following investigations.

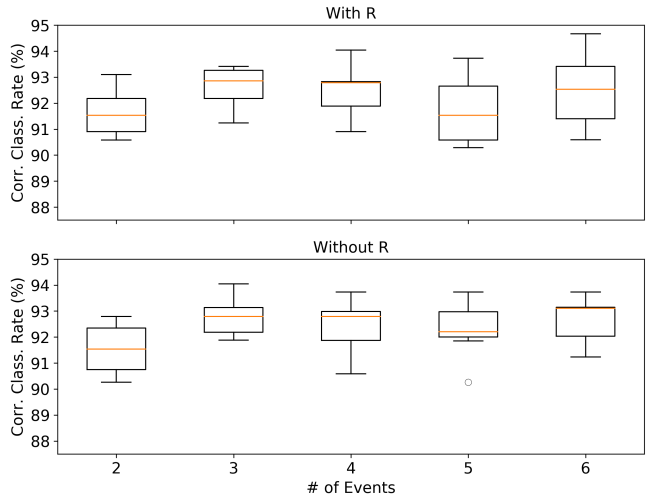

FIGURE 8 The effect of the number of events with and without the including of temporal predicates. The sequences longer than the specified number of units are truncated. According to the applied two-way ANOVA the application of more events nor the temporal predicates result in better correct classification rate neither. In the following investigations we applied four events without the including of the temporal relationships.

Figure 9 illustrates the effect of LSTM unit number. According to the applied one-way ANOVA, we can neglect the null hypothesis, that the number of LSTM units has no significant effect on the correct classification rate of the model, with a significance value of ($p = 0.001$). According to Figure 9, the model with 17 LSTM units slightly outperforms the others, therefore we applied this structure for further analysis.

The size of the embedding layer can also greatly affect the accuracy of the model since this layer maps the one-hot binary vector represented symbols of the states into a continuous vector space. According to Figure 10, the highest accuracy is reached by mapping into a four-dimensional embedding space. However, the one-way ANOVA did not verify this increased correct classification rate as the significance value was above 5% ($p = 0.21$), we applied this structure for the testing of the model.

We demonstrate the performance of the described neural classifier with a confusion matrix in Figure 11. Only two similar faults are difficult to be distinguished (the 8^{th} and the 9^{th} faults). This result is in good correspondence with the

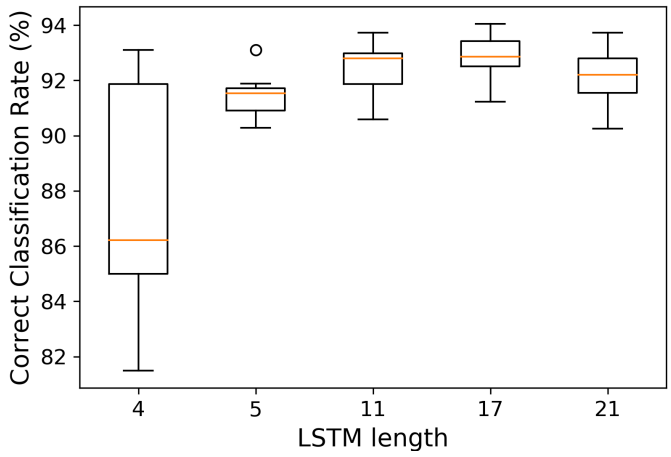

FIGURE 9 The effect of long short-term memory unit number in case of *Dataset B/2* using 7-fold cross-validation. The incorporation of more than 11 units had no significant effect on the correct classification rate.

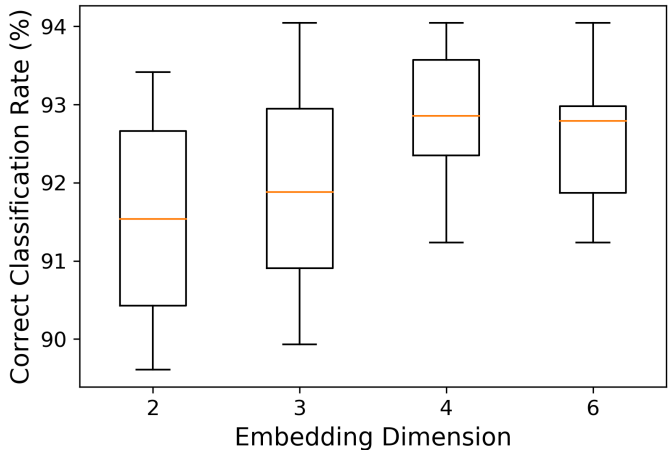

FIGURE 10 The effect of the number of units in the embedding dimension in case of *Dataset B/2*

results presented in Figure 8, namely the proposed algorithm can accurately predict the root cause of events after only a few sequence elements, which is highly advantageous from the view of the industrial application. This result can also imply that the few characteristic variables, that should be monitored in order to effectively identify the faults can be determined with the analysation of the model.

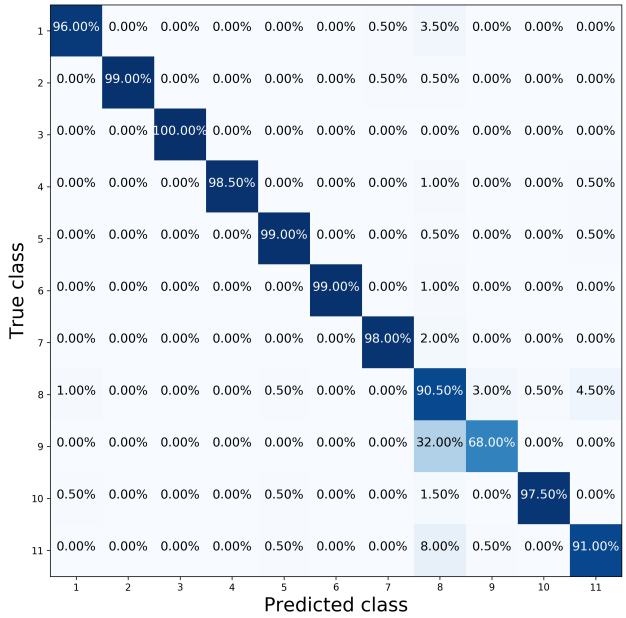

FIGURE 11 Confusion matrix showing the accuracy of the classifier. The percentages were calculated after the classification of 200 sequences following each of the faults. The results indicate that faults 8^{th} and the 9^{th} have similar effects making them difficult to identify.

3.3 | Embedding layer based analysis of process alarms

We visualise the contextual information of sequence elements by the principal component analysis of the **W** weight matrix.

The results of PCA not only shows the neighbouring relations of the alarms and warnings in this space with reduced dimension, but it also represents its significance in the classification problem. Figure 12 illustrates that the most significant events, having large Hotelling's T-Square values are mainly the lower and upper alarms. However, the figure can seem to be overcrowded, the aim of the figure is the visual illustration of the results and all the presented datasets and the programs are available in the supplementary material of the article.

This result offers an outstanding opportunity for the prioritisation of incoming warnings and events to facilitate the work of the operators. Table 6 shows the fault that these alarms can follow. According to the table, most of these events appear in the case of only one or only a few malfunctions. Therefore the visualisation highlights that the principal component analysis of the given alarms can explore the characteristic events of these faults. From an operational point of view, we can see that the high compressor exit temperature (state 115) and the high circulation stream temperature (state 145) event tags are in the same direction at the upper left corner of the diagram, therefore they should indicate similar effects. According to Table 6, the high compressor exit temperature (state 115) indicate the presence of the 4^{th} fault, which means bad compressor heat flow. The high circulation stream temperature (state 145) also indicates the

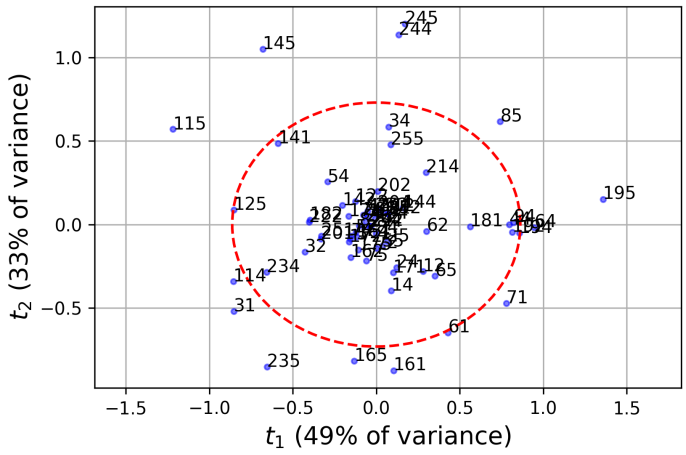

FIGURE 12 The result of principal component analysis on the weight matrix of the embedding layer (the last digit of event tags according to Table : 1 - Low Alarm, 2 - Low Warning, 4 - High Warning, 5 - High Alarm).

presence of the 4^{th} fault, together with the presence of the 6^{th} fault, the bad circulation stream temperature. Therefore as bad compressor exit temperature is experienced the controller of the scrub stream tries to stabilize the temperature of the absorber (the stream coming from the compressor enters the absorber column). However, it is not an outlier of the results of PCA, it is interesting to see that the low circulation stream temperature (state 141) is in the same direction as the high compressor exit temperature (state 115) and the high circulation stream temperature (state 145), and indicates similarly the presence of the 4^{th} and 6^{th} faults. Similarly, the separator level high alarm (state 85), heater exit temperature high warning (state 64) and the FEHE hot stream exit temperature high warning and high alarm (state 194 and 195) show similarity. This similarity can also be explained by an engineering point of view; these temperatures highly influence the level of the separator. If we take into consideration the results of Table 6, it appears all these events are in connection with the 9^{th} fault.

A similar analysis can be carried out on the dendrogram in Figure 13, that represents the Euclidean distance between the row vectors of the embedding layer. The colours in Figure 13 indicate the clusters of process signals, but no former process knowledge-based information was found to be relevant to explain the formed clusters.

For example the pair of the high alarm of gas recycle stream pressure (state 25) and the high alarm of vaporizer pressure (state 55) highlights the problems of the gas cycle, a bad pressure can effect the whole process.The low alarm level for the H_2O content of the column bottom (state 201) and the column bottom level (state 251) are also close to each other since these states are both connected to the bottom of the separation column. A similar connection can be seen between the high warning of reactor exit temperature (state 74) and the scrub stream temperature (state 164), they are connected to each other according to the dendrogram. Similarly, the high warning of heater exit temperature (state 64) and FEHE hot exit temperature (state 194) draw attention on the control problems of the reactor inlet, since a high alarm before the reactor still appears in the outlet temperature of the FEHE unit.

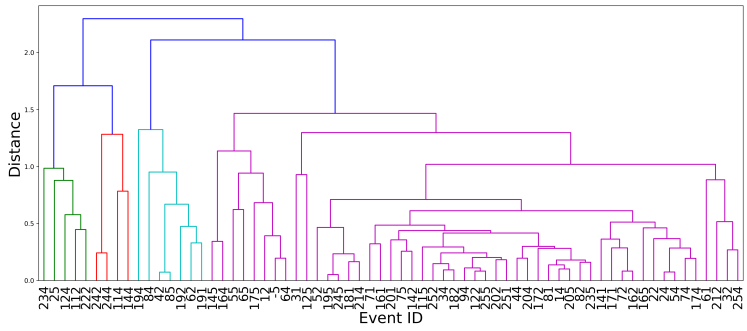

FIGURE 13 The dendrogram of alarms and warnings calculated applying mean linkage of Euclidean distances of the row vectors of the weight matrix of the embedding layer (the last digit of event tags according to Table : 1 - Low Alarm, 2 - Low Warning, 4 - High Warning, 5 - High Alarm). The colours of the different states indicate the found clusters, but no process-relevant knowledge has been paired to the clusters.

3.4 | Analysis of the similarity of process faults based on the activities of the long short-term memory units

To better understand the results of the analysis of the embedding layer and to study how the activation values of the hidden layer of LSTM units represent the process faults, we applied principal component analysis to these activation values as well.

The analysis of the eigenvalues of the **H** matrix of the node activities can give information about the proper number of LSTM units (Section 2.3). According to Figure 14, the eigenvalues confirm the result of the examination of the effect of LSTM unit number (see Figure 9): five units are sufficient to represent the classification problem, while eleven units describe almost entirely the cumulative variance of the variables.

The results of the principal component analysis of the activity values of LSTM units (Figure 15) are in good correspondence with the classification performance illustrated by confusion matrix shown in Figure 11. It is interesting to see that the 11^{th} fault has two distinct clusters according to the results of PCA. The figure confirms the difficult isolation of the 9^{th} fault. The fact that this fault is mostly situated at the centre of PCA projection suggests the lack of an informative state for this fault. The PCA analysis of the alarm and warning signals (Figure 12) confirms this result (also see Table 6). Such analysis may imply the lack of informative alarms and warnings for the detection of a given fault. Therefore the proposed approach can give hints for the development of the control system.

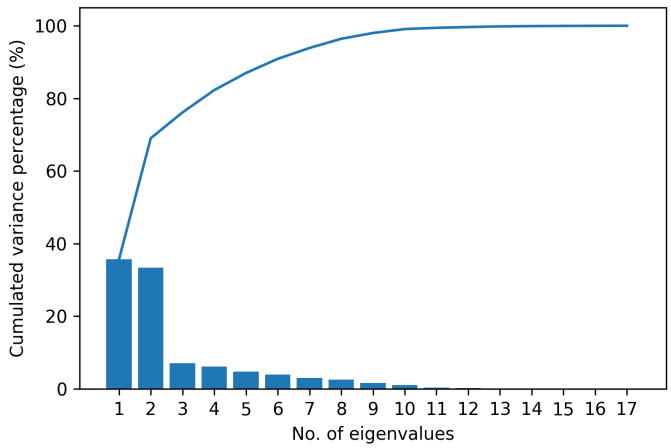

FIGURE 14 Eigenvalues of the **H** activity matrix containing the activity values of the layer of the long short-term memory units. According to the figure, the application of four components can sufficiently describe the variance of the activity values.

TABLE 3 The numbering of controlled variables

Name of Variable	Tag of Variable
$\%O_2$ in the Reactor Inlet	1
Gas Recycle Stream Pressure	2
HAc Tank Level	3
Vaporizer Level	4
Vaporizer Pressure	5
Heater Exit Temperature	6
Reactor Exit Temperature	7
Separator Level	8
Separator Temperature	9
Separator Vapor Flowrate	10
Compressor Exit Temperature	11
Absorber Level	12
Absorber Scrub Flowrate	13
Circulation Stream Temperature	14
Absorber Circulation Flowrate	15
Scrub Stream Temperature	16
$\%CO_2$ in the Gas Recycle	17
$\%C_2H_6$ in the Gas Recycle	18
FEHE Hot Exit Temperature	19
$\%H_2O$ in the Column Bottom	20
5^{th} tray Temperature	21
Decanter Temperature	22
Decanter Organic Level	23
Decanter Aqueous Level	24
Column Bottom Level	25
Liquid Recycle Flowrate	26
%VAc E-3	27

TABLE 4 The digit added to the number of variable indicating the type of event

Type of event	Added digit
Low alarm	1
Low warning	2
Target operating range	3
High warning	4
High alarm	5

TABLE 5 Two-way ANOVA analysis shows that the temporal predicates do not significantly influence the accuracy of the classifier.

Factors	Significance Value (p)
(1) Temporal predicates	0.595
(2) Number of events	0.053
1 by 2	0.924

TABLE 6 Outlier events of PCA and the faults causing these events. Most of the events occur after only one or very less number of faults, therefore the proposed neural classifier could highlight the characteristic alarms of the process giving a good opportunity for the prioritisation of these signals. The tag of the faults can be seen in Table 2.

Event	Name of event	Type of event	Faults
145	Circulation Stream Temperature	High alarm	4, 6
245	Decanter Aqueous Level	High alarm	11
244	Decanter Aqueous Level	High warning	1, 2, 3, 4, 5, 6, 7, 8,9, 10, 11
85	Separator Level	High alarm	3, 9
195	FEHE Hot Exit Temperature	High alarm	9
64	Heater Exit Temperature	High warning	2, 9
194	FEHE Hot Exit Temperature	High warning	4, 6, 7, 9
71	Reactor Exit Temperature	Low alarm	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
61	Heater Exit Temperature	Low alarm	2
161	Scrub Stream Temperature	Low alarm	1, 7
165	Scrub Stream Temperature	High alarm	1, 3, 7
235	Decanter Organic Level	High alarm	10
31	HAc Tank Level	Low alarm	1
114	Compressor Exit Temperature	High warning	2, 4, 9
115	Compressor Exit Temperature	High alarm	4
125	Absorber Level	High alarm	1, 2, 3, 4, 5, 6, 7, 9, 11

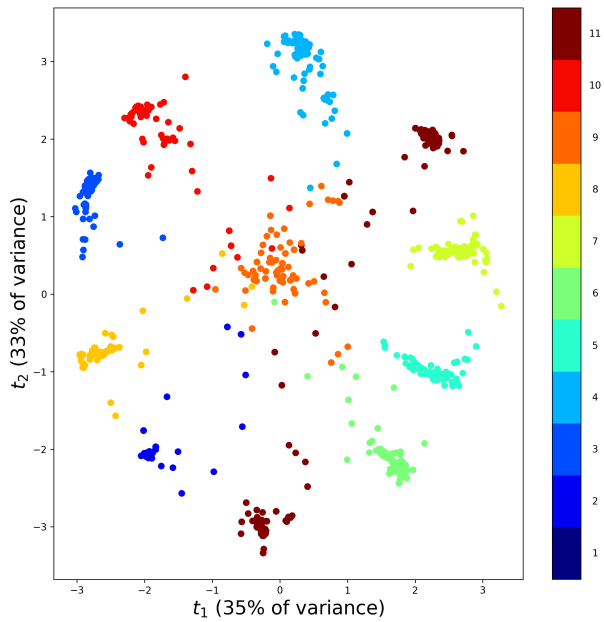

FIGURE 15 The principal component analysis of the activity values of the long short-term memory unit layer. The visualisation of the activity values highlights the well-separable faults and can give suggestions on poorly monitored malfunctions. In this case, the 8^{th} fault reasonably needs more informative alarms and warnings as this fault is in the centre of the graph and its elements show a high distribution in this reduced-dimension space.

4 | CONCLUSIONS

The high amount of alarms and warnings generated by the control systems overload the operators, so the identification and proper handling of malfunctions is a complex and challenging task.

To extract the sometimes hidden information of discrete events in modern chemical plants and the temporal relationships connecting them, we utilised multi-temporal sequences of alarm and warning signals as inputs of long short-term memory (LSTM) unit-based recurrent neural networks trained for classification of process faults and visualised the input and output layers of the model.

The embedding layer is trained simultaneously with the layer of the long short-term memory units and the output layer, resulting in a supervised mapping of the alarm and warning signals into a continuous vector space. This supervised mapping ensures that the principal component analysis of the weight matrix of the embedding layer can highlight the significance of the events from the viewpoint of the process faults.

With the principal component analysis of the activity values, not just the faults become comparable, but it also gives useful information about the proper number of long short-term memory units. The resulted model complexity is in good consistency with the analysis of the effect of the number of units in the embedding layer.

According to our knowledge, the proposed case study is the first that demonstrates that the analysis of the embedding layer and the activity values of LSTM units can give recommendations for the monitoring of malfunctions and offers an outstanding opportunity for the prioritisation of incoming warnings and events to facilitate the work of the operators.

NOMENCLATURE

RNN - recurrent neural network
LSTM - long short-term memory
PCA - principal component analysis
VAc - vinyl acetate
y - type of fault, $y = \{c_1, \ldots, c_{n_c}\}$, related to the k-th sequence of events.
$\hat{y}_k$ - predicted class label of the faults
c_j - j^{th} type fault
n_c - number of fault types
s - *state* of the technology
pv - index of the process variable
a - the related state signals
e - event
st - starting time of an event
et - ending time of an event
$\mathbf{\Phi}_k$ - k^{th} sequence of states
S - the set of states
R - arbitrary temporal predicate between events
E - equal temporal predicates between events
B - before temporal predicates between events
D - during temporal predicates between events
O - overlap temporal predicates between events

$\mathbf{x}_k$ - the numerical representation of $\mathbf{\Phi}_k$ sequence
$P(y_k = c_j|\mathbf{x}_k)$ - the conditional probability of the given fault
$\mathbf{h}_k$ - vector of the activities of the LSTM units
n_U - the number of LSTM units in the RNN
T - length of the input sequence
$\mathbf{W}_s$ - weight matrix of the output layer of the network (referring to the applied softmax function)
C_k^t - cell-state of the t^{th} LSTM unit
f_k^t - forget gate of the t^{th} LSTM unit
$\mathbf{W}_f$ - weight matrix of the forget gate
b - the bias vector of the corresponding neurons
i_k^t - input gate of the t^{th} LSTM unit
o_k^t - output gate of the t^{th} LSTM unit
$\mathbf{W}_o$ - weight matrix of the output gate
$\mathbf{oh}_k^t$ - one-hot binary vector representation of the t^{th} symbol of the k^{th} sequence
n_s - the number of states in the $\mathbb{D}$ temporal database
n_R - number of types of temporal predicates between events
n_o - number of bits in the one-hot binary vector
n_e - dimension of the embedding layer
$\mathbf{W}$ - weight matrix of embedding layer
$sim(s_i, s_j)$ - similarity of the alarms
d_{max} - the maximum Euclidean distance among the rows of the $\mathbf{W}$ matrix

ACKNOWLEDGEMENTS

The research has been supported by the National Research, Development and Innovation Office NKFIH, through the project OTKA 116674 (Process mining and deep learning in the natural sciences and process development) and the EFOP-3.6.1- 16-2016- 00015 Smart Specialization Startegy (S3) Comprehensive Institutional Development Program.

REFERENCES

[1] Blanke M, Kinnaert M, Lunze J, Staroswiecki M. Diagnosis and Fault-Tolerant Control. 3 ed. Springer-Verlag Berlin Heidelberg; 2016. http://gen.lib.rus.ec/book/index.php?md5=ab151ae09b582030adc554aa729a429f.

[2] Chen R, Dave K, McAvoy TJ, Luyben M. A Nonlinear Dynamic Model of a Vinyl Acetate Process. Industrial & Engineering Chemistry Research 2003;42(20):4478–4487. http://dx.doi.org/10.1021/ie020859k.

[3] Chen Y, Lee J. Autonomous mining for alarm correlation patterns based on time-shift similarity clustering in manufacturing system. In: 2011 IEEE Conference on Prognostics and Health Management; 2011. p. 1–8.

[4] Equipment E, Association MU, Equipment E, Staff MUA. Alarm Systems: A Guide to Design, Management and Procurement. EEMUA publication, E E M U A (Engineering Equipment & Materials Users Association); 2015. https://books.google.hu/books?id=SrY3MwAACAAJ.

[5] Gers FA, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM 1999;.

[6] Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press; 2016. http://www.deeplearningbook.org.

[7] Guo W, Banerjee AG. Identification of key features using topological data analysis for accurate prediction of manufacturing system outputs. Journal of Manufacturing Systems 2017;43(Part 2):225 – 234. http://www.sciencedirect.com/science/article/pii/S0278612517300286, high Performance Computing and Data Analytics for Cyber Manufacturing.

[8] Hochreiter S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 1998;06(02):107–116. http://www.worldscientific.com/doi/abs/10.1142/S0218488598000094.

[9] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Comput 1997 Nov;9(8):1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[10] Hopfield JJ. Neural networks and physical systems with emergent collective computational abilities. Proceedings of the national academy of sciences 1982;79(8):2554–2558.

[11] Hu W, Afzal MS, Brandt G, Lau E, Chen T, Shah SL. An Application of Advanced Alarm Management Tools to an Oil Sand Extraction Plant, This work was supported by an NSERC CRD project with Suncor Energy as an industrial partner. IFAC-PapersOnLine 2015;48(8):641 – 646. 9th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2015.

[12] Jing L, Zhao M, Li P, Xu X. A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. Measurement 2017;.

[13] Karoly R, Abonyi J. Multi-temporal sequential pattern mining based improvement of alarm management systems. In: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC); 2016. p. 003870–003875.

[14] Kondaveeti SR, Izadi I, Shah SL, Black T, Chen T. Graphical tools for routine assessment of industrial alarm systems. Computers & Chemical Engineering 2012;46(Supplement C):39 – 47. http://www.sciencedirect.com/science/article/pii/S0098135412002268.

[15] Lin F. Diagnosability of discrete event systems and its applications. Discrete Event Dynamic Systems 1994 May;4(2):197–212. https://doi.org/10.1007/BF01441211.

[16] Ludwig O, Liu X, Kordjamshidi P, Moens MF. Deep Embedding for Spatial Role Labeling. arXiv preprint arXiv:160308474 2016;.

[17] Lusci A, Pollastri G, Baldi P. Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. Journal of chemical information and modeling 2013;53(7):1563–1575.

[18] Lv F, Wen C, Bao Z, Liu M. Fault diagnosis based on deep learning. In: 2016 American Control Conference (ACC); 2016. p. 6851–6856.

[19] Malhotra P, Ramakrishnan A, Anand G, Vig L, Agarwal P, Shroff G. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. CoRR 2016;abs/1607.00148. http://arxiv.org/abs/1607.00148.

[20] Mandal S, Santhi B, Sethumadhavan S, Vinolia K, Swaminathan P. Nuclear Power Plant Thermocouple Sensor-Fault Detection and Classification Using Deep Learning and Generalized Likelihood Ratio Test 2017 04;PP:1–1.

[21] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781 2013;.

[22] Mitchell JB. Machine learning methods in chemoinformatics. Wiley Interdisciplinary Reviews: Computational Molecular Science 2014;4(5):468–481.

[23] Sampath M, Sengupta R, Lafortune S, Sinnamohideen K, Teneketzis D. Diagnosability of discrete-event systems. IEEE Transactions on Automatic Control 1995 Sep;40(9):1555–1575.

[24] Sampath M, Sengupta R, Lafortune S, Sinnamohideen K, Teneketzis DC. Failure diagnosis using discrete-event models. IEEE Transactions on Control Systems Technology 1996 Mar;4(2):105–124.

[25] Schmidhuber J. Deep learning in neural networks: An overview. Neural Networks 2015;61(Supplement C):85 – 117. http://www.sciencedirect.com/science/article/pii/S0893608014002135.

[26] Sønderby SK, Sønderby CK, Nielsen H, Winther O. Convolutional LSTM networks for subcellular localization of proteins. In: International Conference on Algorithms for Computational Biology Springer; 2015. p. 68–80.

[27] Wise BM, Gallagher NB, Butler SW, White DD, Barna GG. A comparison of principal component analysis, multiway principal component analysis, trilinear decomposition and parallel factor analysis for fault detection in a semiconductor etch process. Journal of Chemometrics 1999;13(3-4):379–396. http://dx.doi.org/10.1002/(SICI)1099-128X(199905/08)13:3/4<379::AID-CEM556>3.0.CO;2-N.

[28] Yang F, Shah SL, Xiao D, Chen T. Improved correlation analysis and visualization of industrial alarm data. ISA Transactions 2012;51(4):499 – 506. http://www.sciencedirect.com/science/article/pii/S0019057812000341.

[29] Zaytoon J, Lafortune S. Overview of fault diagnosis methods for Discrete Event Systems. Annual Reviews in Control 2013;37(2):308 – 320. http://www.sciencedirect.com/science/article/pii/S1367578813000552.

[30] Zhao R, Wang D, Yan R, Mao K, Shen F, Wang J. Machine Health Monitoring Using Local Feature-based Gated Recurrent Unit Networks. IEEE Transactions on Industrial Electronics 2017;PP(99):1–1.

[31] Zhu J, Wang C, Li C, Gao X, Zhao J. Dynamic alarm prediction for critical alarms using a probabilistic model. Chinese Journal of Chemical Engineering 2016;24(7):881 – 885. http://www.sciencedirect.com/science/article/pii/S1004954116303044.