An Ensemble of LSTM Networks for Fault Detection, Classification, and Root Cause Identification in Quality Control Line

Gürkan Aydemir [1], Adem Avcı[2], Mustafa Kocakulak [3], Tahir Bekiryazıcı [4]

[1,2,3,4] *Bursa Technical University, Bursa, 16310, Turkey*
gurkan.aydemir@btu.edu.tr
adem.avci@btu.edu.tr
mustafa.kocakulak@btu.edu.tr
tahir.bekiryazici@btu.edu.tr

ABSTRACT

Industrial systems with multiple subsystems are monitored via various sensors to control the ongoing process. If the number of monitoring signals collected from these sensors is high and the number of faulty samples is low, then the machine learning methods may fail to provide effective solutions for fault detection and root cause identification. This paper proposes an efficient feature selection model based on the regularized LSTM neural networks, and fault detection and classification using an ensemble of binary LSTM classifiers. The model is verified in PHME Data Challenge 2021 which provides quality-control-pipeline monitoring data.

1. INTRODUCTION

Fault detection, identification, and prediction in complex production systems with several subsystems are still challenging tasks. Deep learning algorithms try to provide effective and efficient solutions to these problems (Zhang et al., 2019). In the literature, various deep learning architectures have been proposed to detect faults and estimate failure time. Han et al. employed convolutional neural networks-based spatio-temporal feature learning for fault diagnosis in complex systems (Han, Liu, Wu, Sarkar, & Jiang, 2019). Li et al. utilized Hidden Markov models for rotating machinery to detect and classify faults (Li, Wei, Wang, & Zhou, 2017). Deep belief networks were also proposed to detect multiple faults in a complex system, namely a cryogenic propellant loading system (Ren, Chai, Qu, Ye, & Tang, 2018).

Machine learning (ML) algorithms, particularly neural networks, need large training datasets to model fault detection, classification, and estimation appropriately (Zhang et al., 2019; Han et al., 2019; Li et al., 2017; Ren et al., 2018). The training of ML models becomes challenging if the number of samples is limited in any task (Ren et al., 2018). If there are also a large number of monitoring signals besides limited samples, data-driven ML models suffer from the curse of dimensionality. The model may easily overfit in training in such cases since the number of training parameters is high as compared to the available measurements.

This study uses a regularized long short-term memory(LSTM) network to select prominent features and suggests using an ensemble of LSTM networks to detect the faults in streaming data. Firstly, an LSTM network for each fault is trained using a cost function in which the ℓ_1 norm of the weight coefficients is added to binary classification loss. This approach provides sparse weight coefficients in the LSTM network, i.e., the network uses only a few input features to detect each fault. Once the prominent features are determined, a separate LSTM network that uses only the related prominent features as input is trained as a binary classifier to detect defined faults. In the validation dataset, the classifiers are ranked according to their accuracy. The final fault decision is made according to the priority of the classifiers. If none of the classifiers have a fault alarm, the state is determined to be faultless. The proposed model is verified on real-world industrial testbed data provided by PHME Data Challenge 2021.

The rest of the paper is organized as follows: Section 2 briefly reviews the provided dataset. Section 3 presents the suggested feature extraction and fault detection approach in detail. Section 4 evaluates the performance of the proposed model on given data. Lastly, Section 5 concludes the paper and discusses the future research direction.

2. ABOUT PHME DATA CHALLENGE 2021

The dataset for PHME Data Challenge 2021 was generated by the collaboration with Swiss Centre of Electronics and Microtechnology (PHMEurope21, 2021). This dataset consists of data points collected for the quality control of elec-

trical fuses. All fuses on this system are assessed in a two-stage quality control mechanism. The data has been acquired from a real-world industrial test-bed that is used for a fuse test bench.

The overall system is monitored via 50 signals, which contain information about the environment, the system's health status, and some other factors. The dataset has been acquired in different experimental sections. Each experiment is run from 1 to 3 hours. During the data acquisition process, the experimental data is gathered over a time window of 10 seconds. While experimental data is acquired, features of each signal are extracted automatically using statistical methods. These features are shown in Table 1.

Table 1. Features and description

Features	Description
vCnt	Number of samples recorded within time-window
vFreq	Sampling frequency within time-window
vMax	Maximum value recorded within the time-window
vMin	Minimum value recorded within the time-window
vTrend	Time series trend within the time-window
value	Mean value recorded within the time-window

The experimental dataset has been collected under fault-free operating conditions. However, with the help of domain experts, a variety of faults under controlled conditions were created. 8 different fault labels were defined in total and these faults were created with the help of one or more signals in fault-free experiments using 2 operating conditions. The training set consists of 5 faulty classes, each containing 4 experiments, and faulty-free classes containing 50 experiments. In addition, there is the model refinement set that consists of 3 faulty classes, each containing 3 experiments, and faulty-free classes containing 20 experiments.

The first objective of PHME Data Challenge 2021 is to identify and classify faults by developing a model to predict predefined faults in unlabeled test data. The second objective is to rank the effect of input signals on identified fault occurrence. The third objective is to accomplish these 2 objectives in the shortest time interval. Lastly, the fourth objective of this challenge is to develop unsupervised solutions to identify experiments with different system configurations. In this paper, we propose an architecture to identify the related signals for all fault types and detect them using LSTM networks.

3. Proposed Method

LSTM neural network architecture is utilized in feature selection and fault detection. The input data should be preprocessed appropriately to feed into the neural networks. In this section, firstly the preprocessing procedure is described in detail. Secondly, the LSTM network is briefly reviewed and discussed. Then, the feature selection approach is presented. Lastly, the section is ended with the fault detection model.

3.1. Preprocessing

The dataset includes two linearly dependent summary statistical features, namely vCnt and vFreq, as seen in Table 1. Therefore, vFreq is chosen and excluded to reduce the number of features. Furthermore, since all other features are missing when the value of vCnt is equal to 0, all missing data is replaced by 0 to feed them into the neural networks in such cases.

There are both nominal and ordinal categorical input features in the dataset. However, the number of different values in ordinal categorical features is high, and so, it is assumed that all features are nominal. Normalization/Standardization is another critical preprocessing step that remarkably improves the performance of neural networks (Goodfellow et al., 2016). Min-max normalization is the most popular way of handling nominal data in LSTM networks. However, in the provided data described in Section 2, the input values have outliers. In this case, min-max normalization confines most of the values around a specific value (0.5 in case of $[0, 1]$ normalization and 0 in case of $[-1, 1]$ normalization), which decreases the information content of the features. Therefore, standardization which is described in the following equation is preferred instead of min-max normalization to overcome this problem:

$$x'_i = (x_i - \mu_i)/\sigma_i \tag{1}$$

where x'_i, x_i are standardized and original ith feature values, μ_i and σ_i are mean and standard deviation of ith feature values, respectively.

Theoretically, the LSTM networks are capable of handling streaming data of any length. However, the maximum length of the input sequence should be limited for the training of LSTM. Thus, the LSTM network is employed with a sliding window approach. A time window of features that includes the current and past measurements is fed into the LSTM model at each step to detect the fault. These windows are created from the input sequences in an overlapping manner in the last stage of preprocessing in training.

3.2. Long short-term memory network architecture

LSTM network is a special type of recurrent neural network to capture long short-term dependencies in time sequence data (Hochreiter & Schmidhuber, 1997). The architecture of an LSTM cell with a forget gate is illustrated in Figure 1.

The cell state c_t and hidden state h_t is computed in recurrent

Table 2. Priority order of classifiers and selected features

Priority Order	Class id	Sequence Length	Feature 1	Feature 2	Feature 3	Feature 4
1	11	5	SmartPosition-Error	DurationRobotFrom-FeederToTestBench		
2	3	10	FeederBackground-IlluminationIntensity	VacuumValve-Closed		
3	5	15	VacuumValve-Closed	Vacuum	NumberFuse-Detected	
4	12	15	DurationRobotFrom-FeederToTestBench	SmartMotor-Speed		
5	9	15	SmartMotor-Speed	SmartMotor-PositionError		
6	2	25	FeederAction2	VacuumFuse-Picked	Vacuum	Temperature-ThermoCam
7	7	60	FusePicked	FuseIntoFeeder	FuseOutside-OperationalSpace	
8	4	40	LightBarrierActive-TaskDuration1	EPOSPosition	LightBarrierPassive-TaskDuration1	IntensityTotal-ThermoImage

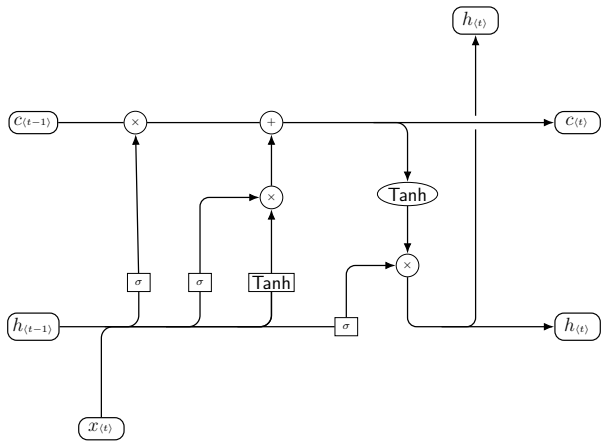

Figure 1. A single LSTM cell

manner with the following equations:

$$
\begin{aligned}
f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \\
i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \\
o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \\
\tilde{c}_t &= tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\
h_t &= o_t \circ tanh\left(c_t\right)
\end{aligned}
\tag{2}
$$

where x_t, h_t, and c_t are input, hidden state and cell state vectors at time instance t, respectively. $\sigma()$ and $tanh()$ are sigmoid and hyperbolic tangent activation functions that are defined as:

$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \tag{3}$$

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x}-1}{e^{2x}+1} \tag{4}$$

Ws, Us and bs are trainable weight parameters of input and recurrent connections and bias vectors, respectively. The trainable parameters in the LSTM network are trained using gradient descent methods. However, as opposed to other feed-forward neural networks, a special type of backpropagation, namely Backpropagation Through Time (BPTT), is employed in parameter training (Werbos, 1990).

3.3. Feature selection using regularized LSTM networks

If a system is composed of multiple subsystems, its health is monitored via multi-sensors. These sensors' measurements or features may lead to a curse of dimensionality for machine learning models if the number of collected samples from some classes (notably for faulty cases) is low. A machine learning model, especially a neural network designed to detect a specific fault, might overfit the training samples and fail to detect the fault in the test. LSTM networks also suffer from such a curse of dimensionality.

A regularized LSTM architecture to select the prominent features related to the faults may overcome the high dimensionality of input data. An LSTM cell with 16 hidden and cell states and with a maximum sequence length of 15 is designed for this task. A fully connected layer with only one hidden neuron and sigmoid activation is added to the end of the LSTM cell to determine the class of the input sequence.

The training and refinement samples are divided into train-

ing and validation sets. The samples with the shortest length from the faulty classes are left for the validation set to increase training data size. 4 random samples from the fault-free class are put into the validation set. The training and validation datasets are preprocessed by following the procedure defined in Section 3.1 with a sliding window length of 15 and a sliding step of 1.

In an ordinary classification problem, the model parameters can be trained using the binary cross-entropy loss (cost) described as follows:

$$\mathrm{L}(\mathrm{y}, \hat{\mathrm{y}}) = -\frac{1}{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i) \quad (5)$$

where $\hat{y}_i$ and y_i are the model's prediction and the actual label for the ith sample. ℓ_1 norm of the Ws in (2) is added to loss function as follows:

$$L_{total} = L(y, \hat{y}) + \lambda \|W\|_1 \quad (6)$$

Penalization of ℓ_1 norm of W, which are the weight coefficients that are related to inputs, enforces several near 0 values in W (Tibshirani, 1996). It is assumed that if all of the weight coefficients related to an input feature are close to 0, then the feature is not important for this fault. Therefore, the selection of the prominent features is achieved via the penalization of the input weight coefficients. The number of near zero values in W is controlled by the constant λ.

A separate binary LSTM classifier with the same hyperparameters is trained for each type of fault by using the regularized loss (6). The binary cross-entropy part in the regularized loss (6) is calculated by weighting the different classes inversely proportional to the number of samples in each class since there is a considerable class imbalance between the faulty and faultless classes. BPTT is employed with Adam optimizer (Kingma & Ba, 2014). Different λs from the candidate values $\{0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001\}$ are used in training, the best performing ones in the validation set are selected for each class. If the performance is similar for more than one λ value, a higher one is preferred. Because the high λ values provide more near 0 values in Ws. All of the weight coefficients for each feature are added for each classifier, and the ones that are smaller than 1% of the maximum are approximated as 0. As a result, the features that are corresponding to the nonzero W values in the best performing classifiers are selected as the prominent features for the related faults. Table 2 demonstrates the features that are selected for each fault type.

3.4. Fault detection model

The fault detection model consists of 8 independent fault-specific binary LSTM classifiers. Each classifier takes only fault-related features as input. For instance, a binary LSTM classifier for fault number 11 is designed to operate with the measurements only from the sensors, SmartMotorPositionError and DurationRobotFromFeederToTestBench, as seen in Table 2.

The number of hidden and cell states is fixed to 16. A fully connected layer, with a single hidden neuron and sigmoid activation, follows the LSTM layer. Sequence lengths of LSTM classifiers are optimized by starting at a high value (80) and gradually decreasing by 5 to a point where the performance reduces drastically. The shortest sequence length with the best performance is selected as sequence length for each binary LSTM classifier as illustrated in Table 2. The fault alarms are counted, and the fault decision is taken when the number of fault alarms reaches the window length of the classifier to increase the accuracy further.

Binary LSTM classifiers are ordered according to their validation data set accuracy as shown in Table 2. It is assumed that the classifier's decision with the highest precision in the validation set is more reliable than the others. Thus, it is preferable to the others in case of multiple fault alarms. This priority-based ensemble of binary classifier models is summarized in Figure 2.

The importance order of the fault-related features is also of interest to identify the root cause of the fault. The sensor measurement whose absence decreases the classification performance at most is selected as the most critical signal. The features are given according to their importance order in Table 2.

4. RESULTS

4.1. Validation performance

In the experiments, leave-one-out-cross-validation is used as stated in Section 3.4. The samples with the shortest time periods are left for the validation from each faulty class, and four healthy samples are chosen among the faultless class. The other samples are used for training to maximize the training size.

The model is implemented using the Keras library in Python on Jupyter Notebook (Chollet et al., 2015). Scikit-learn API's K-means clustering algorithm is directly used for the bonus part instead of designing a custom clustering model (Pedregosa et al., 2011; Hartigan & Wong, 1979).

The validation performance is evaluated in terms of single-point accuracy and presented with the confusion matrix in Table 3. However, a fusion of fault alarms is used for giving a fault decision. Multiple fault alarms are waited to be caught as mentioned in Section 3.4 to increase the accuracy further. With this strategy, 100% accuracy is achieved in classification, but the required time for the fault decision is increased. Table 4 shows the fault decision time for each class on vali-

Table 3. Single point classification performance of the proposed model on validation data.

	Faultless	**Fault 2**	**Fault 3**	**Fault 4**	**Fault 5**	**Fault 7**	**Fault 9**	**Fault 11**	**Fault 12**
Faultless	0.97	0	0	0.01	0	0.02	0	0	0
Fault 2	0.02	0.98	0	0	0	0	0	0	0
Fault 3	0	0	1	0	0	0	0	0	0
Fault 4	0.08	0	0	0.90	0	0.02	0	0	0
Fault 5	0	0	0	0	1	0	0	0	0
Fault 7	0.12	0	0	0.06	0	0.82	0	0	0
Fault 9	0	0	0	0	0	0	1	0	0
Fault 11	0	0	0	0	0	0	0	1	0
Fault 12	0	0	0	0	0	0	0	0	1

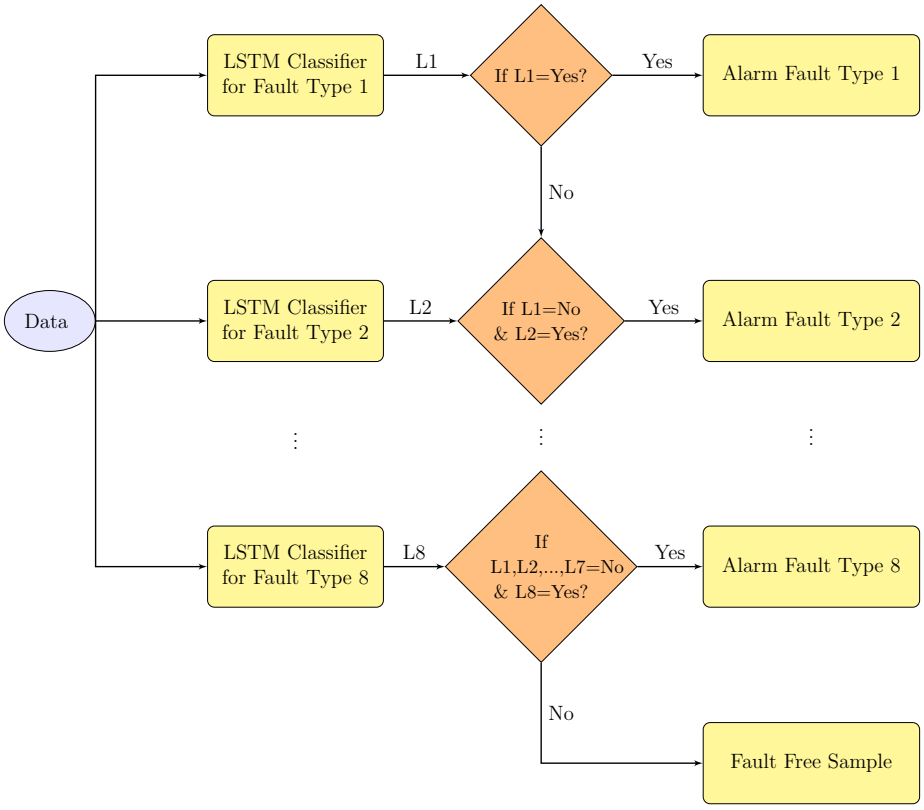

Figure 2. LSTM ensemble classifier

dation data.

4.2. Test performance

Three main performance indexes are given to assess the performance of the fault detection models, classification accuracy, identification of the signal related to the faults, and the time required for detection. Identification of the fault-related signal is assessed using the following criteria:

$$\text{Ranking}(x) = \begin{cases} 1 - 0.15(x-1) & \text{if } x < 4 \\ 0.5 & \text{otherwise} \end{cases} \quad (7)$$

where x is the rank of the most important signal in the proposed model. Time performance of the designed detection algorithm is evaluated as the average relative time with respect to the time of the model that detects the fault in the shortest time as follows:

$$\text{Relative Time}(e,i) = \frac{ST(e)}{T(e,i)} \quad (8)$$

where ST is the time of the fastest detection model among the competitors and T is the proposed model's first fault detection time. A bonus point is provided for the clustering of

Table 4. Detection time for each class in validation data.

Class	Detection Time
Faultless	120
Fault 2	50
Fault 3	20
Fault 4	80
Fault 5	30
Fault 7	120
Fault 9	30
Fault 11	10
Fault 12	30

the system configuration parameters. Bonus point is calculated using the adjusted rand index as follows:

$$\text{Bonus} = \begin{cases} 1.3 & \text{if adjusted rand index } \geq 0.8 \\ 1.2 & \text{if } 0.7 \leq \text{adjusted rand index} < 0.8 \\ 1.1 & \text{if } 0.5 \leq \text{adjusted rand index} < 0.7 \\ 1.0 & \text{otherwise} \end{cases} \tag{9}$$

$$\text{Score} = \text{Accuracy} * \text{Ranking} * \text{ Relative time} * \text{ Bonus} \tag{10}$$

adjusted rand index The proposed model achieved an accuracy score of 0.8235, a ranking score of 0.7, a time score of 0.767, a clustering score of 1.0, an overall score of 0.4422 and lastly, ranked 2$^{\text{nd}}$ in the data challenge.

5. Conclusion

This paper presents a model with an ensemble of LSTM networks for fault selection and fault detection on systems with multiple subsystems. The efficiency of this model is verified on a benchmark dataset of PHME Data Challenge 2021. Based on the obtained results, the proposed model provides a satisfying performance in classification, timely detection, and root cause analysis. However, the rough clustering method fails to identify system configuration parameters properly. The study shows that the developed methodology is an effective tool for diagnostics and prognostics of industrial systems with several subsystems. As future work, the performance of this methodology can be verified using well-known diagnostics and prognostics datasets. Moreover, the effect of the weight penalization parameter λ for feature selection performance should be analyzed further.

References

Chollet, F., et al. (2015). *Keras.* GitHub. Retrieved from `https://github.com/fchollet/keras`

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1) (No. 2). MIT press Cambridge.

Han, T., Liu, C., Wu, L., Sarkar, S., & Jiang, D. (2019). An adaptive spatiotemporal feature learning approach for fault diagnosis in complex systems. *Mechanical Systems and Signal Processing*, *117*, 170–187.

Hartigan, J. A., & Wong, M. A. (1979). Ak-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *28*(1), 100–108.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, C., Wei, F., Wang, C., & Zhou, S. (2017). Fault diagnosis and prediction of complex system based on hidden markov model. *Journal of Intelligent & Fuzzy Systems*, *33*(5), 2937–2944.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

PHMEurope21. (2021). *Data Challenge – PHME21.* Retrieved from `https://phm-europe.org/data-challenge`

Ren, H., Chai, Y., Qu, J., Ye, X., & Tang, Q. (2018). A novel adaptive fault detection methodology for complex system using deep belief networks and multiple models: A case study on cryogenic propellant loading system. *Neurocomputing*, *275*, 2111–2125.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560.

Zhang, L., Lin, J., Liu, B., Zhang, Z., Yan, X., & Wei, M. (2019). A review on deep learning applications in prognostics and health management. *IEEE Access*, *7*, 162415–162438.