A Multi-Output Deep Learning Model for Fault Diagnosis Based on Time-Series Data

Ahmed Al-Ajeli[1], and Eman S. Alshamery[2]

[1,2] *University of Babylon, Hilla, Babil, 51001, Iraq*
a.alajeli@uobabylon.edu.iq
emanalshamery@uobabylon.edu.iq

ABSTRACT

In this work, a method for fault diagnosis and localization is proposed. This method adopts the long short-term memory (LSTM) neural network to detect, isolate and determine the component of the system in which a fault has occurred. Unlike the traditional methods used for fault diagnosis, which first extract features from the raw data and then use a classifier in order to diagnose the fault; the LSTM-based method works directly on raw data and builds the classifier. This can be accomplished by training the neural network using the raw data resulting in a trained model (classifier) capturing generalized patterns from this data. This model is used online to diagnose faults and determine the faulty component. The performance of the resulting model is evaluated on testing data. The proposed method has been applied to real time-series data representing sensor readings in spacecraft electrical power distribution systems. The experimental results show promising performance in separating fault modes and identifying the faulty components.

1. INTRODUCTION

A fault can be described as an undesired deviation of a system or any of its components from its normal or intended operation. Fault diagnosis is the process of detecting and isolating such a fault (Feldman, Kurtoglu, Narasimhan, Poll, & Garcia, 2013; Zaytoon & Lafortune, 2013). Fault detection aims at deciding whether the system works in a normal state or a fault has occurred. Fault isolation aims to determine the fault modes. Then, we identify which system component causes the fault, if it has occurred, through fault localization. The problem of fault diagnosis and localization has received remarkable attention in recent years, where it plays an essential role in ensuring system safety in many different application fields, from industrial power plants (W. Zhang, Jha, Laftchiev, & Nikovski, 2020) to aerospace vehicles (Daigle, Roychoudhury, & Bregon, 2015) and computer systems (Elsisi et al., 2022). Thus, developing methods and algorithms for addressing such a problem and making a timely decision regarding the occurrence of faults are becoming necessary.

Generally speaking, there are three categories of faults: abrupt persistent, drift and abrupt intermittent (Poll et al., 2011; Zaytoon & Lafortune, 2013), as illustrated in Fig. 1. Each one of the three fault categories alters a distinctive property linked to the faulty component (for example, the sensor's output, the load's resistance, etc.). Further, under each category of fault, we have different fault modes. Our interest in this paper is to diagnose each fault mode and identify the faulty component that caused it. Mainly, these faults can originate from both sensors and system components.

Various methods from different communities have been proposed to address the fault diagnosis problem. Essentially, these methods can be listed under three categories: (a) expert systems (Angeli, 2008) in which knowledge about system behavior is encoded into a form that can be used for making diagnosis decisions, (b) model-based methods (Sampath, Sengupta, Lafortune, Sinnamohideen, & Teneketzis, 1996; Daigle et al., 2015; Al-Ajeli & Parker, 2021) where a model of the system being diagnosed already exists. This model captures the normal and fault behavior of the system and (c) data-driven methods (Dai & Gao, 2013; L. Zhang, Lin, & Karim, 2015, 2016) which use collected datasets to learn parameters that could be considered as a model based on which the diagnosis decisions are made.

In recent years, deep learning methods, which are also data-driven methods, come as an attractive direction to address the problem of fault diagnosis (Li, Li, & Kamarthi, 2023). Due to its enormous representing power and automated feature learning capability in solving complex problems, many researchers have adopted it for solving this problem. Also, these methods have shown their universal applicability to various types of input such as imagery (Xu, Zheng, Guo, Wu, & Zheng, 2019), structured data (Ma, Ni, Xie, & Dong, 2017) and time-series data (Yang & Kim, 2018). For fault diagno-

https://doi.org/10.36001/IJPHM.2024.v15i1.3829

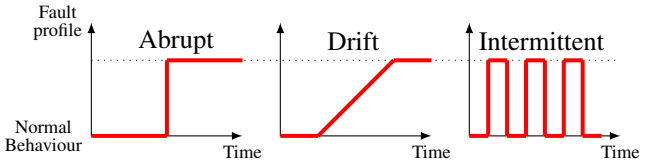


Figure 1. Fault types

sis, time-series data require more attention as they encapsulate temporal dependencies that are typically crucial for the process. This topic has become even more crucial in a multivariate case in which cross correlations amongst multiple measurements could arise.

In this work, we reduce the problem of fault diagnosis and localization to a multi-class multi-output classification problem. A multi-output deep learning model is adopted to tackle this problem by providing two outputs, one for classifying fault modes and the other to localize the component that caused such a fault. The LSTM and multi-perceptron neural networks are combined to achieve this goal. The input to these networks represents a set of observations in the form of sensors' readings. We assume that these sensors generate time-series data at different rates due to the difference in sensors' speed. An industrial use case (Sweet, Feldman, Narasimhan, Daigle, & Poll, 2013) that represents a single-string Unmanned Aircraft System (UAS) in which an ADAPT-Lite Electrical Power System (EPS) supplies power to vehicle systems is used to evaluate the performance of the proposed method.

The contributions of this work can be listed below:

- A deep learning method is proposed to create a learning model which is able to both diagnose faults and localize their components. In other words, this model can be considered as a diagnoser that is applied online to make a decision about whether a fault has occurred and its type, in addition to identifying the component at which it happened.
- This method adopts end-to-end learning in which the raw data is directly fed to the model which outputs the class labels. This implies that feature engineering and classification are jointly trained, resulting in better performance.
- The created model is applied to effectively solve a real problem that represents an electrical power system.

The remainder of this paper is structured as follows. Section 2 gives related work in the field of fault diagnosis. A background on the recurrent neural network is covered in Section 3. Section 4 presents a description of the problem and its formulation. The proposed approach which includes the preprocessing step and how to prepare the dataset to be presented to the deep learning model, in addition to the architecture of the model is detailed in Section 6. The results obtained working on the case study and its discussion are provided in Section 7. Section 8 summarises the conclusion and future work.

2. RELATED WORK

Nowadays, the development of fault diagnosis methods is of great interest to many applications. A variety of diagnostic methods have been proposed and applied in different domains. Also, several methods have particularly been proposed to address the ADAPT use case. A method based on artificial immune systems has been developed to detect and classify sensor faults (Mange, Daniszewski, & Dunn, 2011). Then, a rule-based expert system is used to identify the system components that have produced those faults.

Furthermore, a number of model-based methods have been introduced to address the fault diagnosis problem. For instance, RODON (Lunde, Lunde, & Münker, 2006) uses conflicts between the simulated and the observed behaviour to generate hypotheses about possible causes for the observed behavior. If the model contains fault modes besides the normal behavior, these can be used to verify the hypotheses. Another work has been presented in Daigle et al. (2015) in which a qualitative event-based method for fault isolation is applied. This method is based on the analysis of residual signals, where residuals are computed as the difference between observed and predicted system behavior. These works have been implemented as solutions to the ADAPT use case.

As mentioned earlier in this paper, deep learning models have been adopted to address the fault diagnosis problem. A general method using recurrent neural network (RNN) has been proposed to tackle the fault diagnosis problem considering the imbalance classes in time-series datasets (W. Zhang et al., 2020). The authors conducted experiments on two publicly available benchmark datasets: one originated from the PHM Society Data Challenge, focused on an industrial plant, and the other was a dataset for human activity recognition.

An alternative approach presented in the literature is the introduction of a weighted long recurrent Convolutional LSTM model, which incorporates a sampling policy to address automatic feature extraction and imbalanced time-series datasets (Wu, Guo, Lin, Yu, & Ji, 2018). The model is designed with 2-layer CNNs, 2-layer inner LSTMs, and 2-layer outer LSTMs. Additionally, it includes an under-sampling policy and a weighted cost-sensitive loss function. This model has been experimented on PHM 2015 challenge dataset of Cyber-Physical Systems (CPS). This dataset consists of a time series of sensor measurements including six fault types.

Similar to our work, the fault diagnosis problem has been addressed using the LSTM model (Zhao, Sun, & Jin, 2018). The application of the LSTM to this problem is evaluated in the Tennessee Eastman benchmark of a chemical process where the challenge was to handle the "curse of dimensionality"

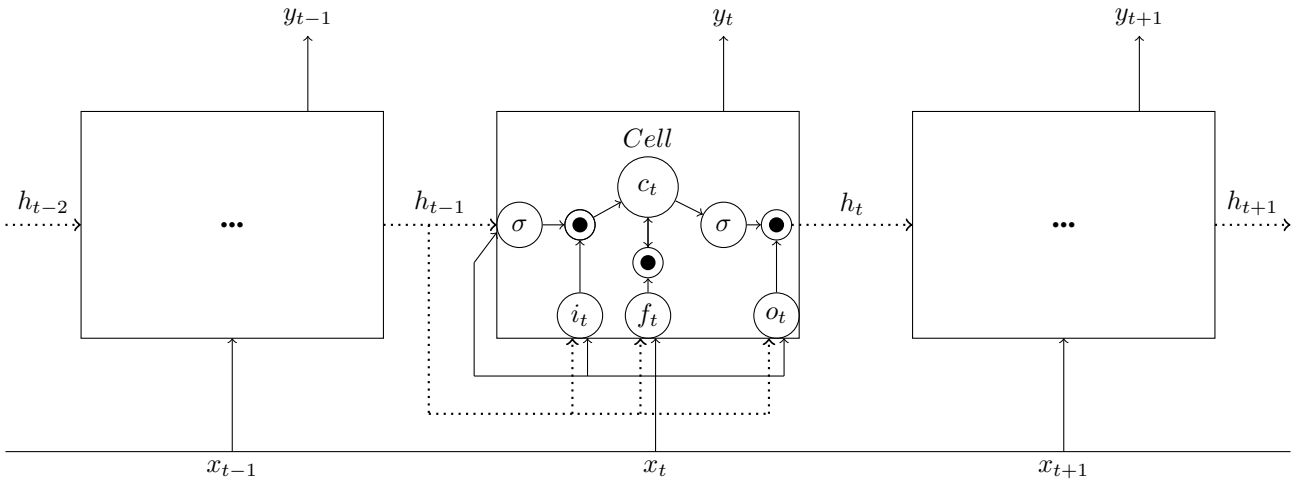


Figure 2. The architecture of the LSTM.

problem and the "data rich but information poor" problem. To reduce the covariate shift problem, this method is incorporated with batch normalization. Different from the present work, the dataset was not a time series and has no significant missing values, in addition, it has no imbalance classes. Recently, the work in Mustafa, Awad, Azzouz, and Azab (2023) has applied the multi-output deep learning approach for fault diagnosis and localization. This work has presented a new approach to detect, classify, and locate different faults in photovoltaic systems. This has been achieved by analyzing voltage measurements taken from a single voltmeter. Our approach is different from this work in that three different types of faults are addressed using a unified approach. This implies that the present approach provides an umbrella under which different types of faults are considered. Also, a robust approach to handle missing values as a result of the difference in sensor rates has been introduced in this paper.

Contrary to the existing approaches, where feature extraction of raw data is performed separately from building the classifier, our approach adopts an end-to-end framework in which both feature extraction and classifier design are combined in the neural network model. As a result, the neural network automatically learns both the features of raw data and the classifier which gives better results regarding the fault diagnosis task and hence leads to improved performance. Although many approaches use machine learning and deep learning, there does not exist a unified approach that tackles different types of faults simultaneously. Further, the present approach represents a robust way to localize fault whether it has occurred in system components or sensors.

3. LONG SHORT-TERM MEMORY NEURAL NETWORKS

The Recurrent Neural Network (RNN) is a frequently employed neural network technique for time series prediction and sequence processing. It accomplishes time series predictions by enabling the state to propagate through feedback connections. Nevertheless, when dealing with long-time series predictions, RNNs often encounter the problem of vanishing gradients, where information retained at time t cannot be told about the inputs obtained many timesteps before. To address this problem, the LSTM was introduced. As proposed by Hochreiter and Schmidhuber (1997), the LSTM is an altered version of the RNN that incorporates memory cells into hidden layers, allowing better management of memory information within the time series data. Inside the LSTM, information is saved for later to prevent the signal from gradually vanishing during processing.

To this end, the LSTM introduces a total of three multiplication gates to form the memory cell. The input gate determines which dimensions of the state will receive updates with new information, the forget gate determines which dimensions will retain their old values and gradually move toward zero, and the output gate determines which dimensions will contribute to the computation of the output value. This architecture is depicted in Fig. 2. It can be seen that the LSTM unit has many inputs and many outputs (many-to-many). In our case, we are only interested in the last output, i.e. many-to-one. Let $x_t \in \mathbb{R}^N$ be the input at time t (current time), $W \in \mathbb{R}^{N\times H}$ and $U \in \mathbb{R}^{N\times H}$ are the weights of input and recurrent layers, $b \in \mathbb{R}^H$ is the bias vector, N is size of inputs and H is the number of hidden layer cells. Then,

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$

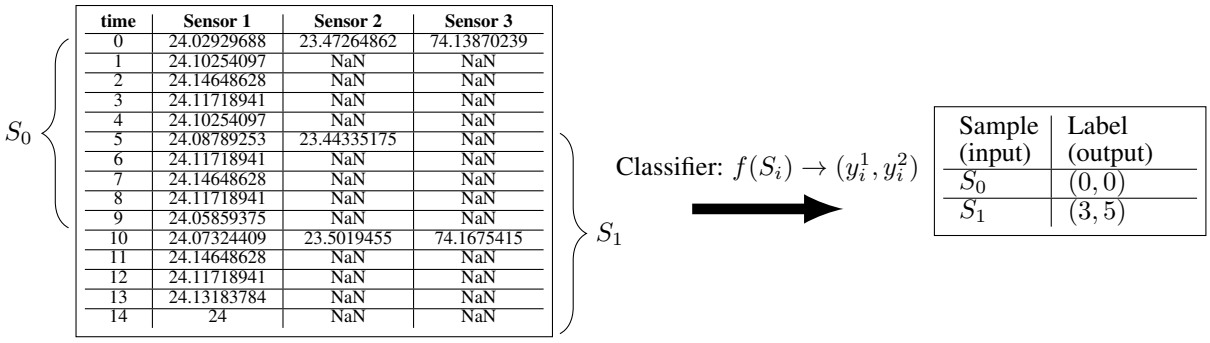
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$

$$h_t = o_t \odot tanh(c_t) \quad (5)$$

where f_t, i_t and o_t refer to activation vectors for the forget,



time	Sensor 1	Sensor 2	Sensor 3
0	24.02929688	23.47264862	74.13870239
1	24.10254097	NaN	NaN
2	24.14648628	NaN	NaN
3	24.11718941	NaN	NaN
4	24.10254097	NaN	NaN
5	24.08789253	23.44335175	NaN
6	24.11718941	NaN	NaN
7	24.14648628	NaN	NaN
8	24.11718941	NaN	NaN
9	24.05859375	NaN	NaN
10	24.07324409	23.5019455	74.1675415
11	24.14648628	NaN	NaN
12	24.11718941	NaN	NaN
13	24.13183784	NaN	NaN
14	24	NaN	NaN

Sample (input)	Label (output)
S_0	$(0, 0)$
S_1	$(3, 5)$

Figure 3. Problem description: a snapshot time-series data and its classifier.

input and output gate, while c_t and h_t denote the vectors for the cell state and hidden state, respectively. In addition, σ and $tanh$ correspond to the sigmoid and hyperbolic tangent activation functions, respectively.

4. PROBLEM STATEMENT

Suppose we have a multivariate time series dataset $D = \{(x_1, Y_1), \ldots, (x_M, Y_M)\}$, where $x_i \in R^N$ represents a N-dimensional vector in which each element is a sensor measurement at timestamp t_i. We also have a 2-tuple $Y_i = (y_i^1, y_i^2)$ of labels associated with x_i: $y_i^1 \in \{0, 1, \ldots, F\}$ and $y_i^2 \in \{0, 1, \ldots, C\}$ where $F,\ C \in \mathbb{Z}^+$ are the number of fault modes and the number of system components, respectively. Labels y_i^1 and y_i^2 represent the normal state or the fault mode (if any) and system component that caused it respectively. Assume that D is divided into a set of samples $S_1, \ldots, S_K$, where $K < M$. Each S corresponds to a sliding window that contains L timestamps. In our work, these samples could be interleaved.

In this paper, the problem of fault diagnosis is formulated as follows. Given a sample S_i, we need to simultaneously assign two labels, one from F and another from C. In this formulation, we reduce the problem of fault diagnosis and localization to a multi-class multi-output classification problem. This involves predicting multiple target variables (outputs) for each sample (instance) in a time-series dataset, where each target variable can have multiple possible classes. To address such a problem, we aim to build a classifier that takes S_i as an input and outputs two labels; to recognize whether the system being analyzed is in a faulty state and then determine the fault mode as well as the component that caused it. Consequently, building this classifier can be viewed as training a classifier that could predict the correct output within reasonable accuracy. This problem is tackled under the following assumptions:

1. Due to the sensor measurements being received at different rates, we have limited observations which lead to significant missing values.
2. Different types of fault are considered: abrupt, drift and intermittent.
3. Dataset classes are imbalanced, where a large part of the dataset represents the normal class. While the fault types are differently represented with fewer samples.

5. CASE STUDY: THE ADVANCED DIAGNOSTICS AND PROGNOSTICS TESTBED

The Advanced Diagnostics and Prognostics Testbed (ADPT) represents spacecraft electrical power distribution systems (EPS) and serves as a diagnostic benchmark created at NASA Ames Research Center (Poll et al., 2007). This paper focuses on a specific part of ADAPT known as ADAPT-Lite. This benchmark involves utilizing ADAPT-Lite hardware to simulate the functioning of an electrical power system on a Single-String Unmanned Aircraft System (UAS) (Kurtoglu et al., 2010; Sweet et al., 2013).

A system layout for the ADAPT-Lite is illustrated in Fig. 4. A portion of components and sensors are utilized to imitate the functioning of an electrical system on UAS. This diagnosis problem does not pertain to any specific UAS model but can be visualized as a generic UAS equipped with instruments designed to gather scientific information. In the figure, **BAT2** refers to a battery that supplies electrical power to multiple loads (**AC483**, **FAN416** and **DC485**) in the UAS setup, where there exists one path from the power source to the loads. The power is transferred via a set of circuit breakers (identified by names starting with **CB**, relays **EY** and an inverter **INV2** that produces AC power). The system also integrates sensors distributed across various points to monitor electrical voltage (recognized by names starting with **E**, electrical current **IT** and the statuses of relays and circuit breakers (**ESH**, **ISH**)). Lastly, there is a sensor for reporting the operational status of a load, specifically the fan speed (**ST**), in addition to the sensor (**TE228**) to report the battery temperature.

Table 1 shows 11 sensors for ADAPT-Lite, where the rate at which the data will be presented to the diagnosis algorithm is indicated. Notice that these sensors have different rates

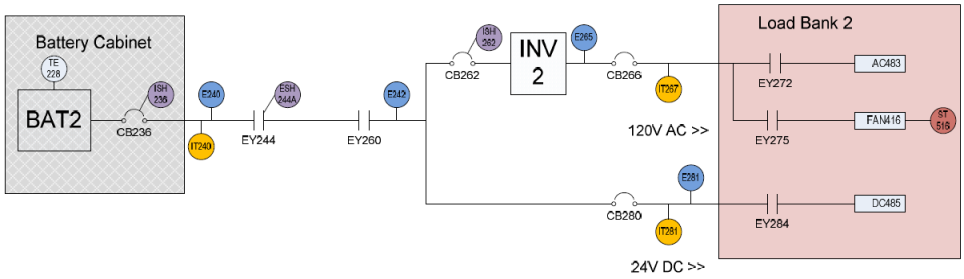


Figure 4. System Schematic for the ADPT.

leading to missing values when collecting measurements as mentioned earlier. The ADPT case study corresponds to a hybrid system that has different types of fault (abrupt, drift and intermittent) with 12 different fault modes as summarised in Table 2. We perform this case study of the fault diagnosis on a dataset generated from the NASA ADAPT-Lite Electrical Power System (EPS). This dataset contains 227 scenarios of sensor observations, 30 of them represent normal scenarios and 197 are fault scenarios. Each scenario takes 4 minutes of time at a maximum rate of 10 observations (time steps) per second. The dataset is provided as DXC'13 Diagnostic Problem I Sample Data (Competition Data from DXC'11: https://c3.ndc.nasa.gov/dashlink/resources/717/).

6. RESEARCH METHODOLOGY

In this section, we describe the steps of the proposed method. This method presents end-to-end learning in which feature extraction and learning are jointly obtained, where the raw data is directly fed into the model, and the outputs are generated about the fault diagnosis. This approach includes three main steps: preprocessing, building the trained model and evaluation. Each of these steps is detailed in the following sections.

6.1. Preprocessing

In our method, we assume that the sensor data have missing values. Also, we assume that input variables (sensor measurements) are not scaled to a standard range. Using deep learning, missing values can be replaced by 0 and the model is still capable of finding the pattern that represents a certain fault type and the component that produced it. To handle the different ranges of input variables, normalization techniques are applied. The application of these techniques results in the rescaling of the data from the original range giving a new range whose values within 0 and 1. In the present method, this can simply be obtained as follows. Let x be the input variable being normalized, and y is the new x after normalization as defined in (6) (Aggarwal et al., 2015).

$$y = \frac{x - min}{max - min} \tag{6}$$

where min and max represent the minimum and maximum values for x.

Table 1. Sensor rate groups for the ADAPT.

Sensor Rate Group		
1 Hz	**2 Hz**	**10 Hz**
TE228	E265	E240
	E281	E242
	IT267	ESH244A
	IT281	ISH236
	ST516	IT240

6.2. Windowing

To prepare the data in a specific form to fit a model, we use the windowing approach. This approach involves dividing the input time series data into windows each of which has a fixed number of observations (time steps) captured by sensors. Also, each window is associated with a fault mode and the component that caused it in a supervisor learning way. Two parameters to control this approach are the size of the window and the stride (how many time steps the window is shifted). Thus, based on this approach the shape of our prepared sensor time-series data will be in terms of the number of samples (windows) K, the number of time steps (observations) in a window L, and the number of features observed at each time step N as a $(K \times L \times N)$ array.

Another thing to be considered when using this approach is the risk of missing the transition from one window to another. In our approach, as traditionally applied, the data is divided into windows with overlapping content. For instance, with 50% overlap, the first half of a window would contain observations from the latter half of the preceding window.

6.3. The LSTM-based Deep Learning Model

The basic architecture of the deep learning model in this work is illustrated in Fig. 5. Two LSTM layers are used each of which has H units. The output of these layers is input to a single fully connected (dense) layer containing P units from which the output is fed to two output layers simultaneously. Such an architecture creates so-called a multi-output model,

Table 2. Fault modes and their components.

Components	Fault mode
AC483, DC485	Failed off, Resistance offset, Resistance drift, Intermittent resistance offset
CB236, CB262, CB266, CB280	Failed open
E240, E242, E265, E281, TE228, IT240, IT267, IT281, ST516	Offset, Stuck, Drift, Intermittent offset
ESH244A, ISH236	Stuck
EY244, EY260, EY272, EY275, EY284	Stuck open
FAN416	Underspeed, Overspeed, Failed off
INV2	Failed off

where given input data, we can predict the fault mode and its component automatically. In other words, this model provides two classifiers: one for the classification of the fault mode and the other for the identification of the faulty component. Since we are interested in solving the classification problem, we ignore all intermediate outputs generated by the second LSTM layer and send only the last output to the next dense layer.

The activation function used in all layers apart from the output layers is the $tanh$ function. To achieve multi-class classification, the softmax activation function is used in both output layers. To train the model, the adaptive moment estimation algorithm (Adam optimization algorithm) (Kingma & Ba, 2014) and categorical cross-entropy as a loss function are applied. In addition, the dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) layer is added after each of the LSTM layers to avoid the overfitting problem.

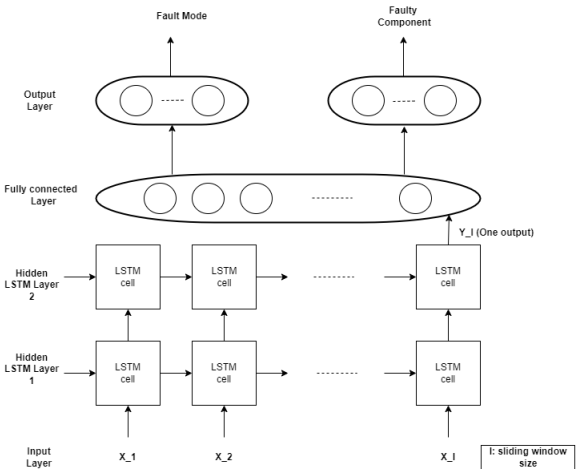


Figure 5. The architecture of the multi-output model.

6.4. Fault Diagnosis Algorithms

Detailed steps of the proposed approach are illustrated in Algorithms 1 and 2. We start offline by creating the fault diagnoser via two main steps: preprocessing the dataset and building the trained model which represents the diagnoser. Afterward, the built model is applied online to diagnose and localize faults or notify that the system is normal.

Algorithm 1 Offline Modeling: Creating Fault diagnoser

Input: D, a time series dataset of S scenarios each of which has T time steps with N features (sensors measurements).
Output: μ and $Accuracy$, a trained multiclass-multioutput deep learning model and its accuracy, respectively.

Let X be the preprocessed dataset.
for each scenario s in S **do**
  for each feature k in s **do**
    if k has missing values **then**
      Replace all missing values by 0
    end if
    Normalise k to be within $[0, 1]$ using (6)
  end for
  for each time step t in s **do**
    Associate to each t a 2-tuple $Y = (y_1, y_2)$ as a label.
  end for
  Segment s to R samples (windows) such that each sample has L time steps.
  Label each sample using the label of the last time step in each window.
  Add these samples to X
end for
Split X into two subsets X_{train} and X_{test}.
$\mu \rightarrow Training\_Model(X_{train})$
$Accuracy \leftarrow Evaluate\_Model(\mu, X_{test})$

Algorithm 2 Online Monitoring: Fault diagnosis

Input: μ, the LSTM-based fault diagnoser model of Algorithm 1; a data sample s of L observations (time steps).
Output: $Y = (y_1, y_2)$, a 2-tuple where y_1 is the fault mode and y_2 is the faulty component.

Replace all missing values of each feature (sensor) in s by 0.
Normalize s according to the maximum and minimum values of training features.
$Y \leftarrow \mu(s)$

During the offline modeling (Algorithm 1), the scenarios of the dataset generated in the case study described in Section 5 are initially prepared for building the model. This includes filling in missing values, applying normalization of features and windowing to create data samples and then labeling them. Then, the created data samples are grouped to generate a new dataset which will be split into a training set and a testing set

Table 3. Parameters setup.

Parameter	Value
Window length	30
Stride	15
Minibatches size	32
Maximum epochs	150
Learning rate	0.001
Dropout rate	0.5
Number of units in LSTM layer 1	132
Number of units in LSTM layer 2	132
Number of units in fully connected layer	132

Table 4. Fault and normal classes percentage.

Class	Fault mode	Percentage %
0	No fault (normal)	44.583
1	Offset	5.813
2	Stuck	4.012
3	ResistanceOffset	4.963
4	FailedOff	1.865
5	OverSpeed	0.499
6	UnderSpeed	0.084
7	StuckOpen	1.963
8	FailedOpen	1.333
9	IntermittentOffset	11.997
10	IntermittentResistanceOffset	8.582
11	Drift	7.093
12	ResistanceDrift	7.213

via the *holdout* method. We used 70% of the dataset for the training and 30% for the testing. Precisely, the number of samples was 25106 and 10760 for the training set and testing set, respectively. The training set is input to the deep learning model in Fig. 5 to construct the trained model μ which contains the optimal parameters (weights). The resulting model is evaluated on the testing set to compute its accuracy.

The online monitoring step (Algorithm 2 takes the trained model μ and inputs sample s of sensor observations to make a diagnosis decision whether a fault occurs to then give its mode and the component that caused it or no fault has occurred (normal state).

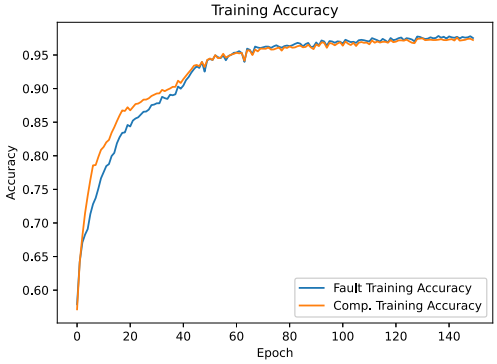


Figure 6. Classification accuracy of both fault modes and faulty components against training iterations.

Table 5. Faulty component classes and their percentage.

Class	Faulty component	Percentage %
0	None (No fault)	44.583
1	E242	0.764
2	IT240	5.387
3	E265	1.475
4	E240	1.904
5	ESH244A	0.354
6	ISH236	0.728
7	E281	1.469
8	IT267	5.459
9	TE228	1.210
10	IT281	5.055
11	ST516	5.111
12	AC483	10.570
13	CB262	0.095
14	DC485	11.167
15	INV2	0.354
16	FAN416	1.115
17	CB236	0.507
18	CB266	0.535
19	CB280	0.195
20	EY260	0.413
21	EY272	0.700
22	EY275	0.181
23	EY284	0.290
24	EY244	0.379

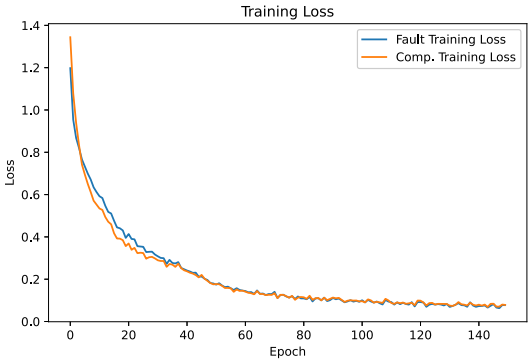


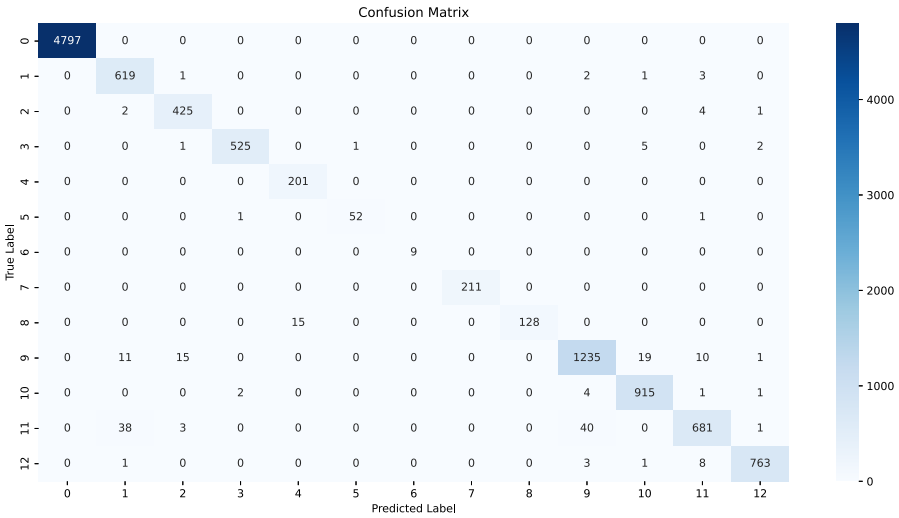
Figure 7. Classification loss of both fault modes and faulty components against training iterations.

7. EXPERIMENTS AND RESULTS

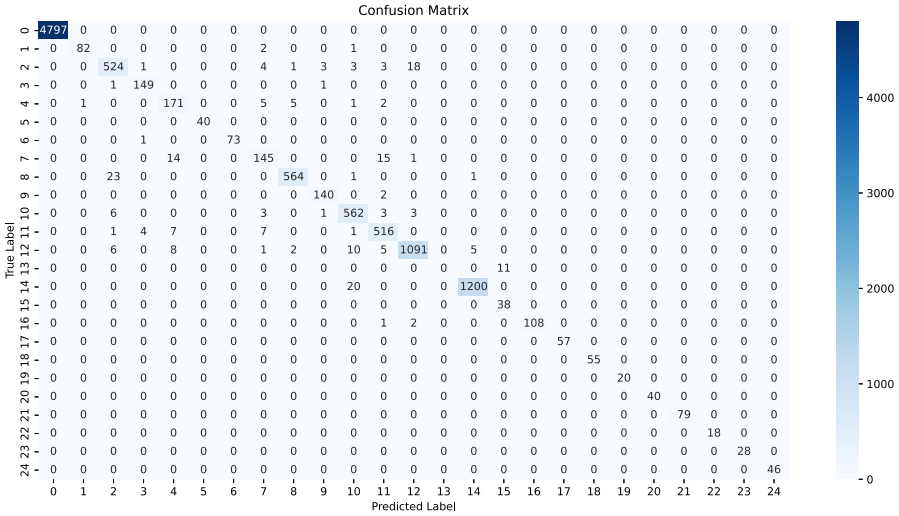
This section evaluates the performance of the present approach to the ADPT example. We implement our work using Python and Tensorflow/Keras library. The dataset generated by the ADPT has a significant number of missing values, in addition, it is an imbalanced dataset as shown in Table 4. The normal class in both tables represents the majority class with 44.583% of the whole dataset. Regarding the fault mode, it can be seen that the $UnderSpeed$ mode is the minority class with 0.084%. On the other hand, the component $CB262$ corresponds to the minority class of faulty components with 0.095%.

Table 6. A comparison between the baseline RNN and LSTM models.

Training model	Classification task	Evaluation Metrics				
		Loss	Accuracy	Precision	Recall	F1-score
baseline RNN	Fault Mode	0.7478	0.7033	0.9668	0.5493	0.5339
	Faulty Component	0.7090	0.7621	0.9684	0.6469	0.6518
LSTM	Fault Mode	0.0606	0.9801	0.9813	0.9788	0.9743
	Faulty Component	0.0691	0.9756	0.9779	0.9730	0.9257



(a)



(b)

Figure 8. Confusion Matrix. (a) In case of fault modes classification and (b) In case of faulty component classification.

A total of 11 fault modes have been injected during the simulation experiments as summarised in Table 2 beside the corresponding faulty component in Table 5. Our experiments have been conducted using hyperparameters listed in Table 3. The values of these parameters have been chosen based on the trial and error method. A set of experiments is carried

out to investigate the change in classification accuracy and loss (classification error) over the training iterations (epochs). Fig. 6 shows the relationship between classification accuracy for both fault modes and faulty components. The results indicate that the proposed approach has achieved a high performance in terms of classification accuracy and loss. During 150 epochs, it has reached an accuracy of around 0.98 in both diagnosing faults and determining faulty components. Also, a high reduction in classification error (around 0.05) has been obtained as depicted in Fig. 7.

We also provide the confusion matrix in order to give a detailed analysis of the fault diagnosis algorithm presented in this paper. In the confusion matrix, the rows represent the actual labels (classes) and the columns correspond to the predicted labels by the algorithm. Each cell in the diagonal represents a match between the actual class and the predicted one. Whereas the other cells denote places where the algorithm misdiagnoses the faults. In Fig. 8a, the confusion matrix shows that all samples of class 0 (normal class), class 4 ($FailedOff$ fault), class 6 ($UnderSpeed$ fault) and class 7 ($StuckOpen$ fault) have correctly been diagnosed. In contrast, the algorithm misdiagnoses 82 samples of class 11 ($Drift$ fault) which has 763 samples. This is the highest rate of misclassification given by our algorithm.

The diagnosis results of faulty components are illustrated in Fig. 8b. As it can be seen, in 11 classes $(0, 5, 15, 17-24)$, all samples have correctly been determined. On the other hand, all 11 samples of class 13 ($CB262$ component) have been misdiagnosed. Also, we observe that all misdiagnosed samples in the experiment of Fig. 8a are still diagnosed as faults, i.e. the rate of the false negative is 0.

Table 6 summarises the experimental results obtained by comparing our method to the baseline RNN using testing data for both fault diagnosis and faulty component localization. As the table shows, our algorithm roughly results in a classification error (loss) of 0.06 for both diagnosis and localization of faults. However, the other four evaluation metrics point out that the diagnosis algorithm slightly produces better performance for fault diagnosis than faulty component localization. In either case, the LSTM-based model significantly outperforms the baseline RNN in terms of all evaluation metrics. In particular, the present method has achieved up to 98% of accuracy and precision metrics.

8. CONCLUSION

In this paper, the problem of fault diagnosis and localization has been addressed. We proposed an LSTM-based method to create an end-to-end multi-output deep learning model in which feature extraction and learning stages are integrated. This model is capable of producing two classifiers. One classifier was used to diagnose the fault mode whereas the other one was to identify the faulty component. The proposed method has been applied to solve a real industrial problem and the experimental results have shown high performance in terms of accuracy. This problem has many challenges such as having different types of faults (abrupt, drift and intermittent), significant missing values and imbalances classes. Using the LSTM networks can learn from the minority class instances over time, even if they are infrequent, and incorporate this information into the model's internal state. Based on the obtained results, this method opens the door to further applications where we have the plan to extend it to solve a multiple fault diagnosis problem aiming to detect and isolate one or more faults at the same time.

REFERENCES

Aggarwal, C. C., et al. (2015). *Data mining: the textbook* (Vol. 1). Springer.

Al-Ajeli, A., & Parker, D. (2021). Fault diagnosis in labelled petri nets: A fourier–motzkin based approach. *Automatica*, *132*, 109831.

Angeli, C. (2008). Online expert systems for fault diagnosis in technical processes. *Expert Systems*, *25*(2), 115–132.

Dai, X., & Gao, Z. (2013). From model, signal to knowledge: A data-driven perspective of fault detection and diagnosis. *IEEE Transactions on Industrial Informatics*, *9*(4), 2226–2238.

Daigle, M. J., Roychoudhury, I., & Bregon, A. (2015). Qualitative event-based diagnosis applied to a spacecraft electrical power distribution system. *Control Engineering Practice*, *38*, 75–91.

Elsisi, M., Tran, M.-Q., Mahmoud, K., Mansour, D.-E. A., Lehtonen, M., & Darwish, M. M. (2022). Effective iot-based deep learning platform for online fault diagnosis of power transformers against cyberattacks and data uncertainties. *Measurement*, *190*, 110686.

Feldman, A., Kurtoglu, T., Narasimhan, S., Poll, S., & Garcia, D. (2013). Empirical evaluation of diagnostic algorithm performance using a generic framework. *International Journal of Prognostics and Health Management Volume 1 (color)*, 24.

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*.

Kurtoglu, T., Narasimhan, S., Poll, S., Garcia, D., Kuhn, L., de Kleer, J., & Feldman, A. (2010). *Second international diagnostic competition (dxc'10).*

Li, W., Li, G., & Kamarthi, S. (2023). The study of trends in ai applications for vehicle maintenance through keyword co-occurrence network analysis. *International Journal of Prognostics and Health Management*, *14*(2).

Lunde, K., Lunde, R., & Münker, B. (2006). Model-based

failure analysis with rodon. In *Proceedings of the 2006 conference on ecai 2006: 17th european conference on artificial intelligence august 29–september 1, 2006, riva del garda, italy* (pp. 647–651).

Ma, J., Ni, S., Xie, W., & Dong, W. (2017). Deep auto-encoder observer multiple-model fast aircraft actuator fault diagnosis algorithm. *International Journal of Control, Automation and Systems*, *15*, 1641–1650.

Mange, J., Daniszewski, D., & Dunn, A. (2011). Artificial immune systems for diagnostic classification problems. In *Proceedings of 21st international workshop of principles of diagnosis.*

Mustafa, Z., Awad, A. S., Azzouz, M., & Azab, A. (2023). Fault identification for photovoltaic systems using a multi-output deep learning approach. *Expert Systems with Applications*, *211*, 118551.

Poll, S., de Kleer, J., Abreau, R., Daigle, M., Feldman, A., Garcia, D., & Sweet, A. (2011). Third international diagnostics competition–dxc'11. In *Proc. of the 22nd international workshop on principles of diagnosis* (pp. 267–278).

Poll, S., Patterson-Hine, A., Camisa, J., Garcia, D., Hall, D., Lee, C., ... others (2007). Advanced diagnostics and prognostics testbed. In *Proceedings of the 18th international workshop on principles of diagnosis (dx-07)* (pp. 178–185).

Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., & Teneketzis, D. (1996). Failure diagnosis using discrete-event models. *IEEE Transactions on Control Systems Technology*, *4*(2), 105-124. doi: 10.1109/87.486338

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958.

Sweet, A., Feldman, A., Narasimhan, S., Daigle, M., & Poll, S. (2013). Fourth international diagnostic competition–dxc'13. In *Proc. of the 24th international workshop on principles of diagnosis* (pp. 224–229).

Wu, Z., Guo, Y., Lin, W., Yu, S., & Ji, Y. (2018). A weighted deep representation learning model for imbalanced fault diagnosis in cyber-physical systems. *Sensors*, *18*(4), 1096.

Xu, X., Zheng, H., Guo, Z., Wu, X., & Zheng, Z. (2019). Sdd-cnn: Small data-driven convolution neural networks for subtle roller defect inspection. *Applied Sciences*, *9*(7), 1364.

Yang, J., & Kim, J. (2018). An accident diagnosis algorithm using long short-term memory. *Nuclear Engineering and Technology*, *50*(4), 582–588.

Zaytoon, J., & Lafortune, S. (2013). Overview of fault diagnosis methods for discrete event systems. *Annual Reviews in Control*, *37*(2), 308–320.

Zhang, L., Lin, J., & Karim, R. (2015). An angle-based subspace anomaly detection approach to high-dimensional data: With an application to industrial fault detection. *Reliability Engineering & System Safety*, *142*, 482–497.

Zhang, L., Lin, J., & Karim, R. (2016). Sliding window-based fault detection from high-dimensional data streams. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*(2), 289–303.

Zhang, W., Jha, D. K., Laftchiev, E., & Nikovski, D. (2020). Multi-label prediction in time series data using deep neural networks. *arXiv preprint arXiv:2001.10098*.

Zhao, H., Sun, S., & Jin, B. (2018). Sequential fault diagnosis based on lstm neural network. *Ieee Access*, *6*, 12929–12939.

BIOGRAPHIES

Ahmed Al-Ajeli received the BSc and MSc degrees in Computer Science from the University of Babylon, Iraq, in 1999 and 2002, respectively. After completing his MSc, he worked as an assistant lecturer at the Department of Computer Science, the University of Babylon. In 2017, he received his PhD in Computer Science from the University of Birmingham, the UK. Currently, he holds a senior lecturer position at the Information Security Department, the University of Babylon. Also, he is the head of the Department. His current research interests include fault diagnosis/prognosis in discrete-event systems, artificial intelligence, machine learning, deep learning and software development.

Eman S. Alshamery received the BSc and MSc degrees in Computer Science from the University of Babylon, Iraq, in 1998 and 2001, respectively. After completing her MSc, she worked as an assistant lecturer at the Department of Computer Science, the University of Babylon. In 2013, she received her PhD in Computer Science from the University of Babylon. Currently, she holds a professor position at the Information Security Department, the University of Babylon. Her current research interests include artificial intelligence, bioinformatics, machine learning, neural networks, deep learning and data mining.