Annals of Nuclear Energy 162 (2021) 108521

Contents lists available at ScienceDirect

Annals of Nuclear Energy

journal homepage: www.elsevier.com/locate/anucene

A multiple-architecture deep learning approach for nuclear power plants accidents classification including anomaly detection and “don’t know” response

Marcelo C. Santos [a,*], Claudio M.N.A. Pereira [a,b], Roberto Schirru [a]

[a] Nuclear Engineering Program, COPPE, Federal University of Rio de Janeiro, CP 68509, CEP 21.941-972 Rio de Janeiro, RJ, Brazil
[b] National Commission of Nuclear Energy, (CNEN/IEN), CP 68550, CEP 21941-906 Rio de Janeiro, RJ, Brazil

A R T I C L E I N F O

Article history:
Received 10 December 2020
Received in revised form 11 May 2021
Accepted 24 June 2021
Available online 3 July 2021

Keywords:
Nuclear power plant
Nuclear accidents identification
Deep neural network
Deep learning
Anomaly detection
Time series classification

A B S T R A C T

Nuclear power plants (NPPs) are complex systems that are monitored by a team of highly trained operators, that in case of an anomalous event on the NPP, such as an accident, must quickly analyze a high number of variables in order to identify the event. However, due to the great volume of information that must be rapidly analyzed in those situations, misidentifications can occur. Therefore, this study proposes the development of a modular structured, deep learning (DL)-based, NPP accident identification system including anomaly detection and “don’t know” response capability. In order to assess the system capacity, experiments were conducted using a realistic study case using 16 operational situations of main operational Brazilian PWR NPP. As a result, the best configuration of DL models for the system achieved an average accuracy of 100% in terms of correct “don’t know” identification, while still retaining the accuracy of 99.82% on correct event identifications.

1. Introduction

A Nuclear Power Plant (NPP) is a highly complex system, formed by a wide range of subsystems and components that are designed, built, and operated following strict safety standards and guidelines. The high safety standard of a NPP aims to ensure that despite the great danger posed by the radioactivity inventory in the reactor core, the possibility of an accident with the release of a radioactive plume out of the containment is minimal. In this sense, a NPP, during its operation, is constantly monitored and controlled by a team of highly trained and qualified operators. Thereby, in the occurrence of an anomalous event, the operators must quickly and correctly identify the occurring event, in order to initiate appropriate corrective actions, ensuring the integrity and safety of the NPP.

Due to the fact that during an anomalous event in a NPP, the operators must quickly analyze a large amount of information, even considering that the operators team is greatly prepared, given simply to the high number of variables in constant evolution, which must be constantly monitored during the event, the performance of the operators can be impaired, with the possibility of failures in the diagnosis. Therefore, in order to assist the operators in their decision-making process during these situations, and consequently, ensure the safe and reliable operation of the NPP, one of the main challenges of the Human Factors Engineering field is the development of a Nuclear Accident Identification System (NAIS) capable of autonomously identifying and diagnosing an anomalous event, such as an accident or transient, in a NPP. In this context, for a NAIS to be really considered to be implemented in a NPP, the system must be fast, efficient, and reliable, being able to solve two main tasks, which are Nuclear Accidents Classification (NAC) and “don’t know” response generation.

The main problem that a NAIS proposes to solve is the NAC, which can be characterized as a Time Series Classification (TSC) problem. Since, in the NAC problem, each NPP accident is represented by the temporal pattern of the several state variables of the NPP, that is, each accident can be defined by a set of time series (state variables), that presents a unique pattern for each accident. In this way, the resolution of this problem is found through methods capable of correctly classifying the accidents, based on the time series that describe the behavior of these accidents. Over the years, in order to solve the NAC problem, many methods have been proposed in the scientific literature, especially methods based on Artificial Intelligence (AI) techniques (Bartlett and Uhrig, 1992; Kwon and Kim, 1999; Mol et al., 2002; Nicolau and Schirru, 2017;

* Corresponding author.
E-mail addresses: marcelo@lmp.ufrj.br (M.C. Santos), cmnap@ien.gov.br (C.M.N.A. Pereira), schirru@lmp.ufrj.br (R. Schirru).

https://doi.org/10.1016/j.anucene.2021.108521
0306-4549/© 2021 Elsevier Ltd. All rights reserved.

Pereira et al., 2007; Pinheiro and Schirru, 2019; Rocco and Zio, 2007).

Among the AI techniques used to solve the NAC problem, the Artificial Neural Networks (ANNs) have been one of the most explored, considering that the ANNs are one of the most efficient techniques to solve complex, non-linear, pattern recognition problems. That is especially true when considering the modern ANN paradigm called deep learning, that was introduced in the last decade, and is characterized by the use of Deep Neural Network (DNN) models that are formed by multiple, hierarchically organized, interconnected layers of hundreds to thousands artificial neurons. Thus, although in the past, achieving a high accuracy level in the NAC problem, especially in scenarios with a great number of accidents, was an extremely difficult task. Nowadays, although there are still few works that explore this approach, the use of DNN models in the NAC, is enabling that the solution of this problem to be really achieved, with accuracy levels around 99% being achieved in scenarios with many accidents (Kim et al., 2021; Peng et al., 2018; Pinheiro et al., 2020; Saeed et al., 2020; Santos et al., 2019; Yang and Kim, 2020a, 2020b).

However, although through the use of DNN models, the capacity of NAIS in solving the NAC task has evolved remarkably. This is still not enough to guarantee the viability and security of the system. For a NAIS to be considered complete it must also be able to solve the “don’t know” response generation task. This task is characterized as the problem of providing to a classifier model, such as a DNN, the ability to generate a “don’t know” response when confronted with an unknown input data pattern. In other words, it means to supply the model the ability to identify new, unknown, or unobserved data patterns, that is, data to which the model was not exposed during its training. The capacity of generating “don’t know” response is essential for a NAIS. Considering that during an anomalous event in the NPP, incorrect information provided to the operators can be worse than an undetermined situation of the plant. Recently, as for the NAC problem, approaches based on DNN models, are being used to supply NAIS the “don’t know” response capacity and promising results are being achieved (Kim et al., 2021; Pinheiro et al., 2020; Yang and Kim, 2020a, 2020b).

Despite the promising results that DNN models can achieve in the main tasks of a NAIS, one of the fundamental problems of this type of system still remains, which is structuring the system in such a way that it reaches a satisfactory level of accuracy (close to 99%), both when confronted with data from known classes, and when confronted with unknown data (in this case generating “don’t know” response). Considering that, a problematic behavior that usually is demonstrated by a NAIS, is that when the system is able to achieve a high level of accuracy in identifying known data, its level of detection of unknown data decreases. In contrast, when the system is able to achieve a high level of accuracy in detecting unknown data, its level of identification of known data decreases. Such a problem is still one of the barriers that prevent the real implementation of that a NAIS can be really implemented in a NPP.

Therefore, aiming to develop a NAIS capable of achieving a high level of performance in both tasks, NAC and “don’t know” response. In this work, it is proposed the development of a system with a modular structure, based on DNN models, able to classify NPP design basis accidents and with the ability to identify anomalies and generate “don’t know” response.

As demonstrated by Saeed et al. (2020) the use of a modular structure for a NAIS, grants more flexibility and scalability to the system, since in this structure each task, as the NAC and the “don’t know“ response, of a NAIS is performed by an individual and specific module. Moreover, in the proposed system, one or more DNN models are implemented as the operation core of each system’s module, in order to achieve satisfactory performance in each module’s task. Therefore, the development of the system following this structure, with individual modules for each task that are executed by DNN models, will prevent, for example, that a high level of accuracy in identifying known data, negatively influences the detection of unknown data. Since, the modules only communicate with each other to inform their output, so that the individual operation of a module, will not directly influence the other modules.

Furthermore, another important characteristic of this study is the comparative analysis of the performance of several contemporaneous and state-of-the-art DNN models in the main tasks of a NAIS. Therefore, in order to conduct this comparative analysis, a comprehensive and complex dataset, in terms of the number of events contemplated and similarity between them, was used in the experiment to validate the performance of the system modules. Considering that the dataset contains the design basis accidents (DBAs) of an operational Brazilian PWR NPP, that approaches the experiments realized in this study to the real scenario that a NAIS will face if implemented in a NPP.

2. The main problems tackled by nuclear accidents identification systems

2.1. Nuclear accidents classification

The main challenge of a NAIS is being able to resolve the task of classifying accidents/transients of a NPP. Since, when an accident occurs, it is possible to observe the temporal evolution of the physical state variables involved, starting from the steady-state operation, through monitoring instrument readings. The evolution in time of each of these variables, such as steam flow, hot leg temperature; cold leg temperature; pressurizer level; among others, provide a specific curve (time series pattern), which are unique with respect to the accident type, making them useful to identify the accident. In order to illustrate this definition, Fig. 1 shows the 61-second temporal evolution of 16 state variables during the BLACKOUT accident in a PWR reactor. While Fig. 2 shows the temporal evolution of the same variables during the LOCA accident in the same reactor.

Comparing Figs. 1 and 2, it is possible to notice that there is a clear difference in the behavior of most variables during each accident. Thus, the resolution of this task occurs through approaches capable of correctly classify the accidents, based on the time series pattern of the variables that describe them. Therefore, given this nature of the NAC problem, it fits into a broader class of problems, the Time Series Classification (TSC), which means, that in a general scope, the main objective of a NAIS is to solve a TSC problem.

The TSC can be summarized in a simple way, as the task where the objective is to classify one or multiple time series, in one of two or several predefined classes. A TSC problem can deal with univariate or multivariate time series. In that way, a univariate time series, $X = [x_1, x_2 \cdots x_T]$, is defined as a sequence of real values ordered in time, with X length being equal to the number of real values T. Similarly, a multivariable time series, $X = [X^1, X^2 \dots X^M]$, consists of M different univariate time series, com $X^i \in \mathbb{R}^T$ (Fawaz et al., 2019; Susto et al., 2018).

The TSC task is considered an extremely challenging and widely researched problem, covering a wide range of applications in several fields of study. In the last few decades, interest in TSC has increased even more due to the greater availability of temporal data. As a result, several new methods are being used to deal with the task of TSC, among these new approaches are the DNNs.

Studies involving contemporaneous deep learning models to solve TSC problems are still few, however, the results presented are promising. In this context, a work of great relevance is one of the Fawaz et al, (2019), which is, to date, the largest empirical

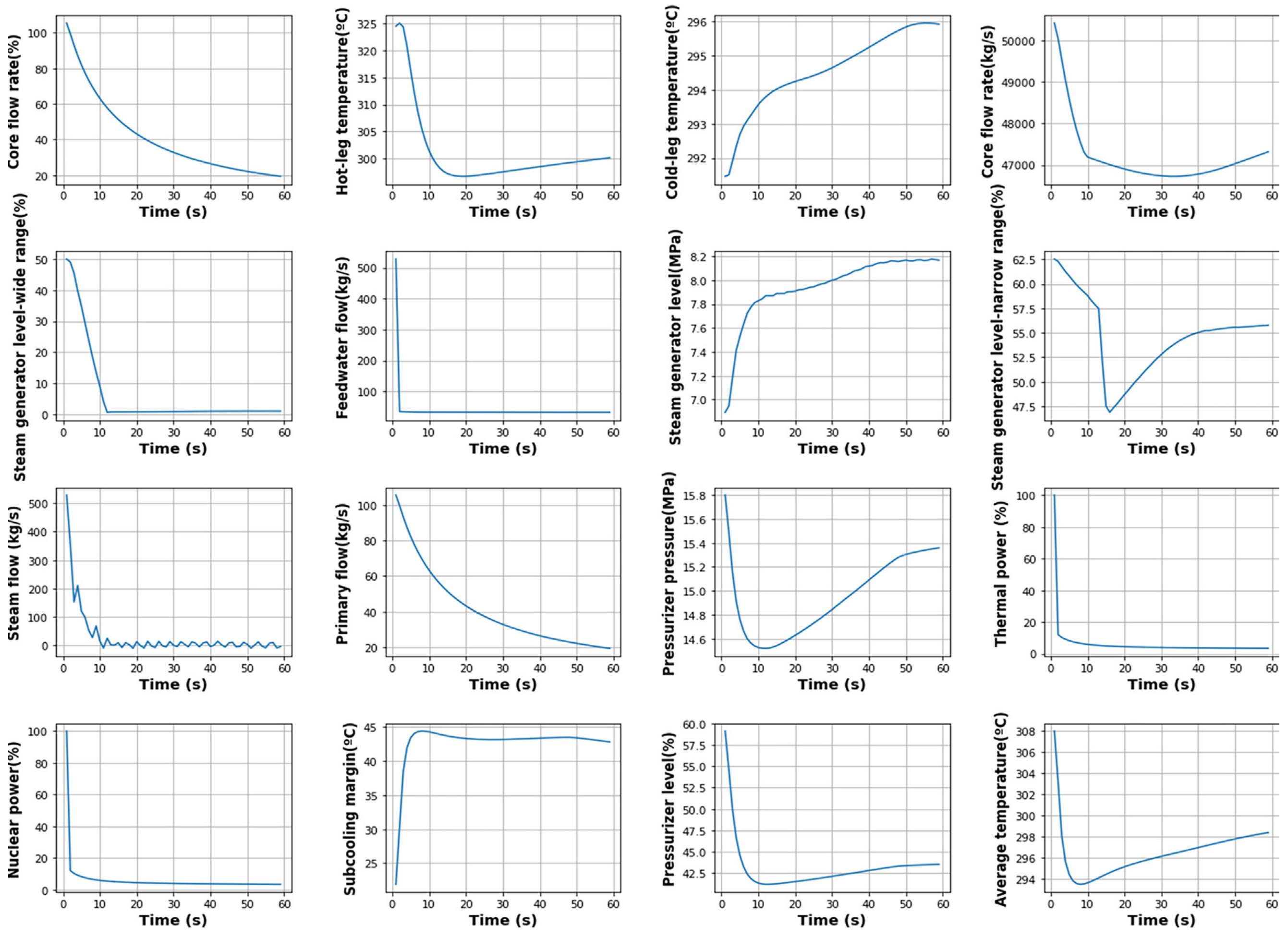


Fig. 1. Temporal evolution of 16 state variables during a BLACKOUT accident in a PWR reactor.

study of DNNs for the TSC task. The study involved specific models of the three main DNN architectures: multilayer perceptron, convolutional, and recurrent. In total, 9 DNN models were used, 1 based on multilayer perceptron architecture, 1 on recurrent architecture, and 7 based on convolutional architecture. These 9 DNN models had already been published in previous scientific works, where they demonstrated good performance on TSC problems.

The performance of the 9 DNN models was compared using 97 traditional datasets used to assess the overall performance for TSC problems (Bagnall et al., 2018; Dau et al., 2019), each of these datasets represented a different TSC problem. Among the models, the Fully Convolutional Network (FCN) was the one that demonstrated the best results, that is, obtained a higher accuracy in the classification in most datasets. In addition, the DNN models were also compared with classic methods used to solve TSC problems, with the same 97 datasets being used for the comparison. The result was that the performance of the DNNs was superior in most datasets, showing that deep learning models, such as the FCN, can achieve state-of-the-art performance on TSC problems.

2.2. "Don't Know" response

The "don't know" response generation is defined as a problem of supplying to a classifier model the ability to generate a "don't know" response when confronted with an unknown input data. In a broader scope the "don't know" response generation problem can be categorized as a novelty detection problem, which is described as the problem of "novelties" identification, in this context "novelties" means new, unknown, or unobserved data patterns, that a machine learning classifier, such as a DNN, was not exposed during its training (Kwon et al., 2019; Markou and Singh, 2003). The "don't know" response generation capability is essential for a classification or identification system based on machine learning techniques, without this functionality the system is not considered really complete (Markou and Singh, 2003). Since, given the open-world characteristic of the vast majority of classification problems, regardless of the quality of the training set used to train a machine learning model, this set is still finite. Thus, it is practically impossible to ensure that the trained model, despite having a high capacity for generalization and operating in a limited scope, can provide a correct answer for all situations presented to it during its operation in the real world.

Therefore, considering a NAIS in which the classifier model is an ANN, assigning the system the ability to generate "don't know" response is even more necessary. Bearing in mind that, when an ANN is trained to classify a set of classes, after the training process, when the respective ANN receives an input, this input will be classified as one of the trained classes of the ANN, even if the level of reliability of this classification is low. Therefore, for a NAIS which implements an ANN model as a classifier, it is vital that a "don't know" response generation approach is also added to the system, in order to validate the classifier's response, ensuring that the network does not classify an unknown input pattern as belonging to one of its known classes.

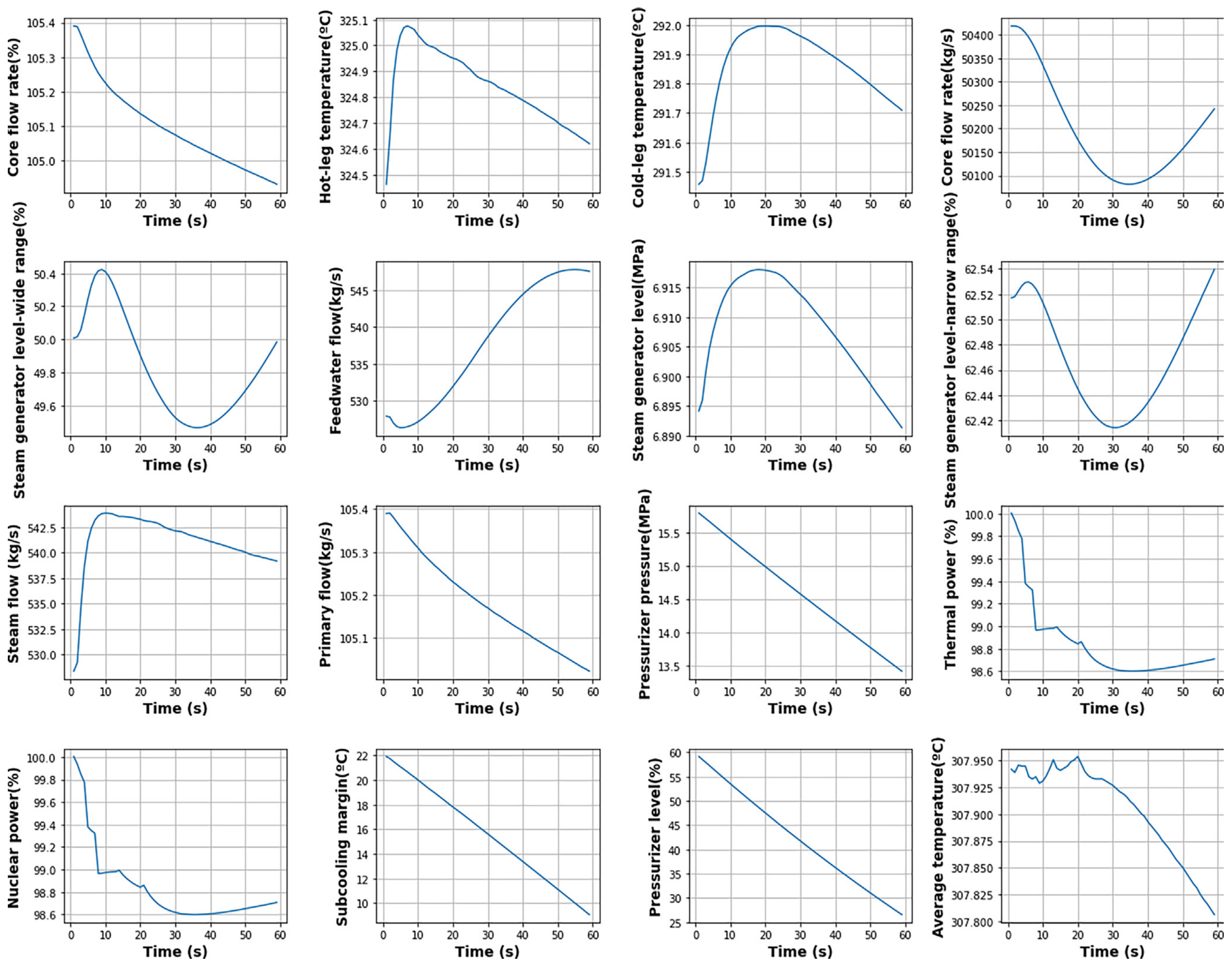


Fig. 2. Temporal Evolution of 16 state variables during a LOCA accident in a PWR reactor.

There are several approaches to deal with novelty detection tasks and, as with many other tasks, in the last few years, approaches based on deep learning have become increasingly popular, as in many cases they are completely overcoming the classic approaches. Among the DNN-based methods, one that has stood out, achieving performance comparable or better than the existing state-of-the-art methods for complex datasets, are the one-class neural networks (OC-NNs) (Chalapathy et al., 2018; Ruff et al., 2018).

An OC-NN consists of training a DNN model, usually an autoencoder (Gutoski et al., 2017; Pinheiro et al., 2020; Kim et al., 2021), with several examples belonging to a single class. So that the model, through the hidden layers, extracts and learns only the common variation factors in the distribution of the data of that specific class. So that after training, when the OC-NN model is presented to an input that does not belong to the class to which it was trained, the hidden layers of the model will not be able to represent that entry correctly, and this inability to represent the entry is shown in some numeric way as a novelty score. Based on this premise, for known entries, the novelty score will be low, while for unknown entries the score will be high, so by defining a threshold value, the novelty score can be used to identify novelties, with entries that present a score greater than the threshold value being considered unknown observations (Chalapathy and Chawla, 2019; Markou and Singh, 2003; Pimentel et al., 2014).

3. Deep learning architectures

Artificial Neural Networks (ANNs) are machine learning models, conceptually inspired by the operation of the human brain, which the main feature is the ability to learn by experience, being able to extract knowledge via a finite set of examples (training samples) and generalize adequately to situations that are not present in the training samples.

In order to learn how to represent the complex non-linear relationships present in the training samples, the ANNs use an approach based on layers of learning, also called hierarchical learning. The premise of this approach is that the learning of complicated concepts happens from the learning of simpler ones (Goodfellow et al., 2016). Therefore, to implement the hierarchical learning approach, the ANNs are comprised of multiple interconnected layers of non-linear parameterized processing units, called neurons. Starting from the input layer, that receives the input signal, each successive hidden layer transforms the signal, via its neurons, and with enough parameterized transformations, the network can learn to represent very complex non-linear relationships in the inputs (Schmidhuber, 2015; LeCun et al., 2015; Benuwa et al., 2016).

In this context, an important aspect of an ANN is the number of hidden layers, in such way that, ANNs that have only one hidden layer are called Shallow Neural Networks (SNNs), while ANNs with more the one hidden layer (usually between 2 and 10) are known as Deep Neural Networks (DNNs) or deep learning models. There is a substantial difference in the learning capacity between a SNN and a DNN. Since the quantity of hidden layers and neurons defines the number of parameterized transformations an input signal encounters as it propagates through the network. Besides that, only the DNNs are able to effectively exploit the hierarchical learning approach, with the deeper layers of the network learning higher-level complex features of a complicated problem, based on lower-level simpler features learned by the early layers. Therefore, DNNs have the ability to represent and learn complex functions far more efficiently than the SNNs (Schmidhuber, 2015; Srivastava et al., 2015; LeCun et al., 2015; Benuwa et al., 2016).

Nowadays, there are several DNN models, however, this work will focus on models derived from the following DNN architectures: Multilayer Perceptron (MLP), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), and Autoencoder (AE). This choice was made due to the fact that, in the literature, DNN models based on these four architectures have already demonstrated good performance on the solution of the main tasks tackled by a NAIS.

3.1. Deep learning models applied in the nuclear accident classification task

3.1.1. Deep Rectifier Neural network

The Deep Rectifier Neural Network (DRNN) (Glorot et al., 2011) is a deep learning model, with a Multilayer Perceptron architecture, in which the main characteristic is the replacement of the traditional sigmoid functions with rectified linear functions, for activation of the neurons of the hidden layer. Such characteristic permits DRNNs to achieve a learning capacity far above the DNNs activated with sigmoid functions. Since, due to the use of linear functions, the DRNNs are not susceptible to the vanishing gradient problem (Glorot et al., 2011).

Recent studies in the literature, including ones in the nuclear area, has been demonstrating that the DRNN is a model superior, in precision and training time, when compared with sigmoid activated DNNs (Pedamonti, 2018; Desterro et al., 2020; Santos et al., 2019; Pinheiro et al., 2020; Teixeira et al., 2020; Dam et al., 2021). Therefore, although models based on the MLP architecture, usually, are not the most popular choice for TSC tasks, as the works of Santos et al. (2019) and Pinheiro et al. (2020) presented, a DRNN model was able to achieve an accuracy of around 99% in the classification of 13 operational situations, simulated for a PWR NPP. Demonstrating that such model can be a viable approach to tackle the NAC problem.

3.1.2. Long Short-Term memory

The RNN is a DNN architecture specialized in processing sequential data, where the elements appear in a certain order and are not independent of each other, e.g. time series. A RNN process an input sequence one element at a time, preserving in their artificial neurons a 'state vector' that implicitly carries information about the history of all the past elements of the sequence.

In this sense, the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is a RNN model that augments the memorization capability of the vanilla RNN by the implementation of a special hidden unit, called memory cell, which functions as long-term memory, preserving relevant information of the input sequence, as well as short-term memory, since it also discards non-relevant information of the sequence. The LSTM memory cell, presented in Fig. 3, operates as an accumulator or a gated leaky neuron since it has a connection to itself at the next time step which has a weight of one. Therefore, it copies its own real-valued state and accumulates the external signal. However, this self-connection is multiplicatively gated by another unit that learns to decide when to clear the content of the memory (LeCun et al., 2015).

Several studies in the literature show the capacity of LSTM models for time series forecasting problems. However, LSTMs are rarely applied to time series classification problems. Still, in works in the nuclear area, where LSTMs were used for the classification of nuclear accidents (Kim et al., 2021; Saeed et al., 2020; Yang and Kim, 2020a, 2020b), good results were found.

3.1.3. Fully Convolutional network

The Convolutional Neural Network (CNN) (LeCun, 1989), also called ConvNet, is a DNN architecture whose principle of operation was inspired by the functioning of the visual cortex of mammals, being particularly designed for automatic feature extraction of data that have the form of multiple arrays. Most data modalities of the real world are formed by multiple arrays, thus, currently, CNN is one of the most widely used DNN architectures, working with different multiple array data structures, and excelling in solving different tasks involving such structures.

Among the many CNN models, the Fully Convolutional Network (FCN), although initially devised for image segmentation tasks (Long et al., 2015), nowadays is considered one of the best models for TSC tasks, more specifically the FCN model developed by Wang et al. (2017). Considering that, in experiments conducted by Wang et al. (2017), using 44 datasets of TSC problems for univariate time series, the FCN achieved the best performance, surpassing other classic and modern deep learning approaches. Moreover, posteriorly, in the work of Fawaz et al., 2019, the FCN's capacity to deal with TSC problems was again confirmed, in a comparative experiment containing 97 datasets of TSC problems, this time containing multivariable time series problems, the FCN once again was one of the DNN models that achieved the best results. However, despite the great capacity that the FCN has been showing to solve TSC problems, such model has not yet been applied in the context of the NAC task. Thus, this work, proposes to evaluate, for the first time, the Wang et al. (2017) FCN in the classification of nuclear accidents task.

3.1.4. Long Shot-Term memory Fully Convolutional network

The LSTM Fully Convolutional Network (LSTM FCN) is a DNN model for TSC tasks, developed by Karim et al. (2017), which tries to improve the performance of Wang et al. (2017) FCN, via the addition of an LSTM sub-module. The internal structure of the LSTM FCN, is based on two DNN architectures, as presented in Fig. 4, and divided into two parallel blocks, one formed by FCN structures and the other by LSTM structures.

In experiments with univariate TSC problems the LSTM FCN achieved performance comparable to the FCN, however, the capacity of the model for multivariate TSC tasks was limited. Therefore, Karim et al. (2019) proposed the Multivariate LSTM Fully Convolutional Network (MLSTM-FCN), to tackle multivariate TSC problems. The main difference of the MLSTM-FCN to the LSTM FCN, as Fig. 4 show, is the introduction of Squeeze-and-Excitation layers (Hu et al., 2018), that uses compression and excitation operations to perform a recalibration of features, allowing the network to learn to use global information, to selectively emphasize the most relevant and informative features of the input, and to suppress the less useful ones.

The MLSTM FCN when evaluated in 35 multivariable time-series datasets, reached a state-of-the-art result in 28 of them (Karim et al., 2019). However, as with FCN, this DNN model has not yet been evaluated in a NAC task and therefore this work intends to make such an assessment.

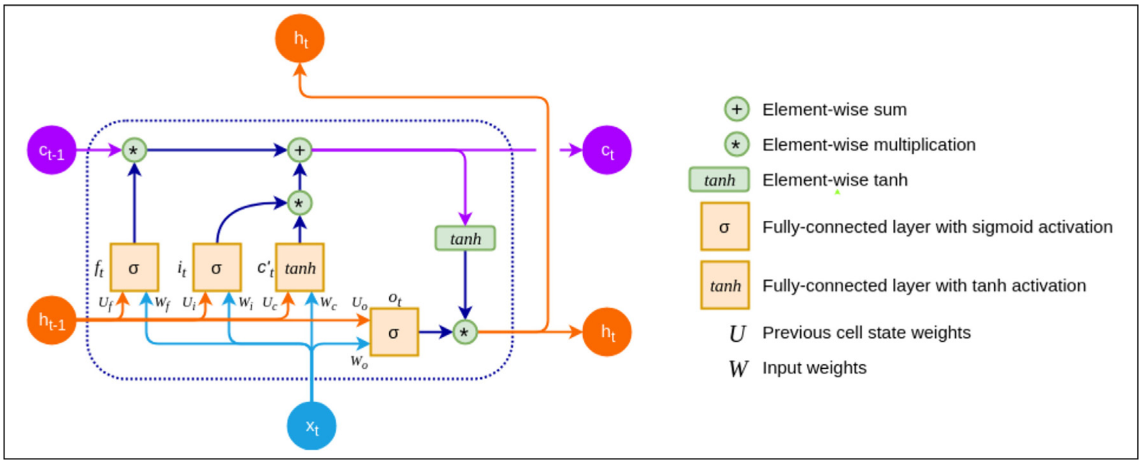

Element-wise sum
Element-wise multiplication
Element-wise tanh
Fully-connected layer with sigmoid activation
Fully-connected layer with tanh activation
Previous cell state weights
Input weights

Fig. 3. The LSTM Memory Cell Basic Architecture (Vasilev et al., 2019).

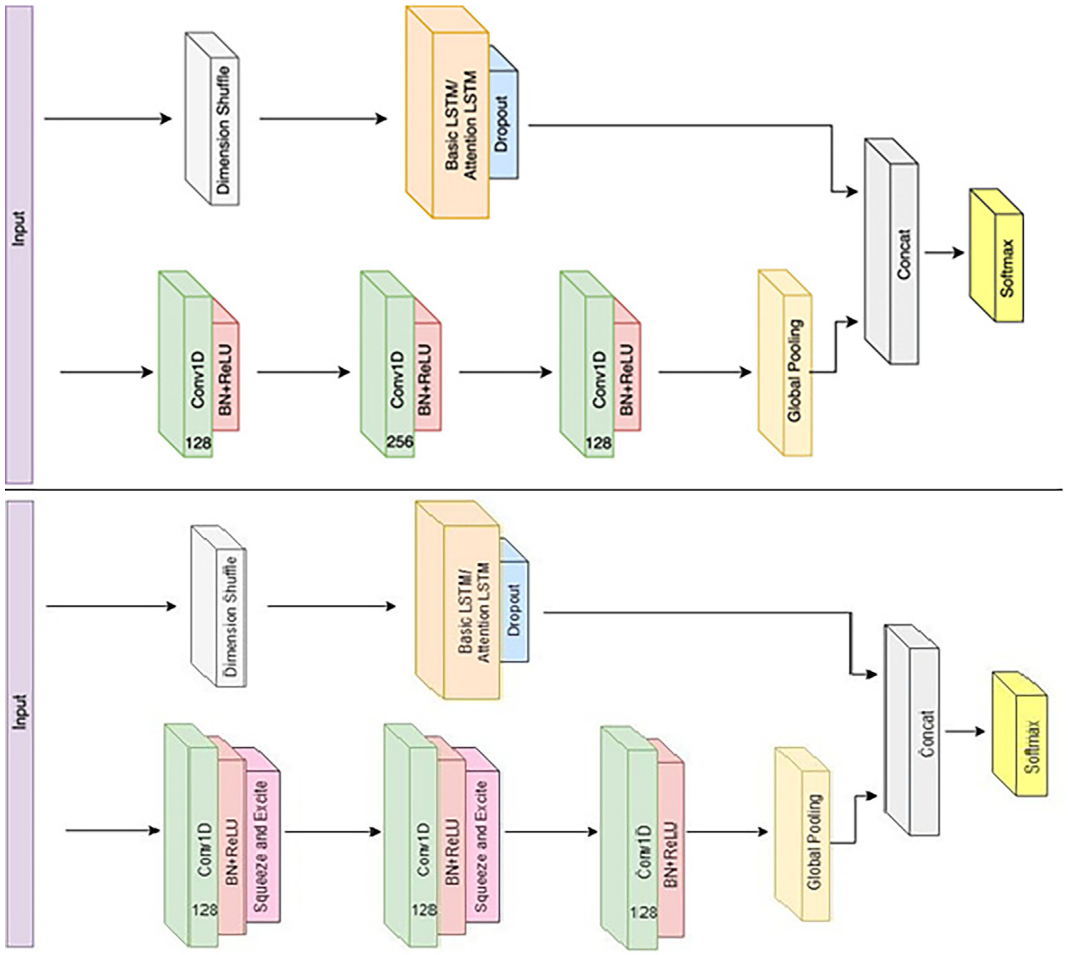


Fig. 4. At top the LSTM FCN (Karim et al, 2017) and at the bottom the MLSTM FCN (Karim et al, 2019).

3.2. Deep learning models applied in the "Don't Know" response task

3.2.1. LSTM Autoencoder

The Autoencoder (AE) (Kramer, 1992), originally called associative network, is a specific type of ANN, which is not exactly an architecture in itself, but rather, a way of designing and training a MLP, CNN or RNN model so that the model learns to reproduce the input data as the output data. Therefore, a LSTM Autoencoder (LSTM AE) (Sutskever et al., 2014) is an autoencoder with recurrent architecture, which is able to model and learn to reproduce sequential input data, such as time series.

The LSTM AE model is formed by an encoding and decoding structure, formed with LSTM layers. As shown in Fig. 5, the encoding LSTM layer receives an input sequence $(x_1, x_2, \ldots, x_n)$

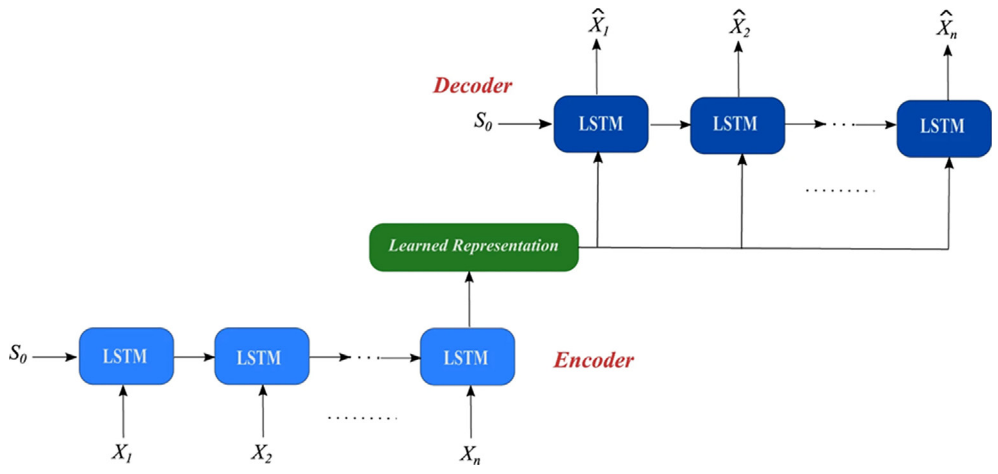


Fig. 5. The LSTM Autoencoder (Sagheer and Kotb, 2019).

and encodes it into a fixed size representation vector. Then, the decoding LSTM layer receives the vector as input and tries to reconstruct the input sequence. Thus, the objective of the LSTM AE is to reduce the reconstruction error, in order to learn to reconstruct the input sequence as the output sequence. Nowadays, the deep autoencoder is the most popular model of AE, and unlike the classic AE that has only one coding and decoding structure, deep autoencoders are formed by several encoding and decoding stages, with these stages being represented by a sequence of encoding layers, followed by a stack of decoding layers (Zhou et al., 2014). By employing multiple encoding and decoding layers, a deep autoencoder has the ability to discriminate and represent complex characteristics of datasets in a more effective way (Zhou and Paffenroth, 2017).

Given the ability of AEs to learn to reconstruct the original trained data with a minimum reconstruction error, when an AE model, after training, is faced with an input that deviates in some way from the data used during training, a high reconstruction error will be generated by the model. As a result, AEs are widely used in novelty detection tasks, as the "don't know" response generation task. Since, by defining a threshold value for the reconstruction error, the AE can be used to indicate whether the input received is unknown, that is, if the input has a different pattern from that observed in the training set.

3.2.1.1. The one class Autoencoder approach. There are different approaches to using AEs for novelty detection tasks, for example, an AE can be trained with a set of classes to learn how to reconstruct them. In the nuclear area, this approach was used by Yang et al. (2019) and Yang and Kim (2020), training an AE model, with recurrent architecture, with a set of nuclear accidents, in order to use the model to indicate when the input received by a classification model, was not part of the accidents known by the model. Another approach is to use AE as an OC-NN, that is, train an AE to learn how to rebuild only a single class. This approach has also been analyzed in the nuclear area (Pinheiro et al., 2020; Kim et al., 2021), in this research multiple one-class autoencoders were used to attribute the "don't know" response capacity to a NPP's accident identification system. In this case, each one-class autoencoder was trained only with data from one of the accidents covered by the system. Although both approaches are valid, the use of OC-NNs for this type of task has achieved better results (Chalapathy et al., 2018; Gutoski et al., 2017; Oza and Patel, 2019; Perera et al., 2019; Wei et al., 2018).

4. Implementation of the deep learning based nuclear accident identification system

4.1. System Description

The system proposed by this work has a modular structure, formed by three main modules, where each module is responsible for performing a specific task. As shown in Fig. 6, the first module of the system is the Anomaly Detection Module (ADM), which has the objective of monitoring the state variables of the NPP and detecting when the behavior of these variables differs from the normal operating pattern. The next module is the Design Basis Accidents Classification Module (DBACM), which focuses on the task of classifying the previously detected anomaly, as one of the DBAs of the NPP. The final module of the system is the Validation Module (VM), which has the function of validating whether the anomaly identified, in the DBACM, really represents the DBA to which it was attributed. Therefore, if the anomaly identified does not meet the validation criteria of the VM, this module is responsible for generating the answer "don't know", meaning that the anomaly identified is unknown and cannot be attributed to any of DBAs of the NPP.

A main characteristic of the proposed system is that the tasks performed by each module are made with the use of DNN models. Therefore, each module has one or more DNN models as its operation core. Besides the accuracy, the use of DNNs will grant the system a fast execution, since, after their training, DNNs can be executed in less than one second. In addition, the modular structure guarantees the system flexibility in its execution, for instance, the only module that is constantly active is the ADM, the other modules are only activated when an anomaly is detected.

4.2. Dataset modeling

In this study, a dataset originated generated by Alvarenga (1997), containing operational scenarios simulated for a Brazilian PWR reactor, the Angra 2 reactor, was used to evaluate the system's performance and train the deep learning models that compose it. The dataset is composed of 16 operational scenarios, which are the normal operation and 15 DBAs, as shown in Table 1. In addition, each operational scenario is represented by a 61-second time evolution of 16 state variables. Table 2 lists these state variables that characterize each operational scenario. Moreover, from this description, the dataset used in this study can be defined

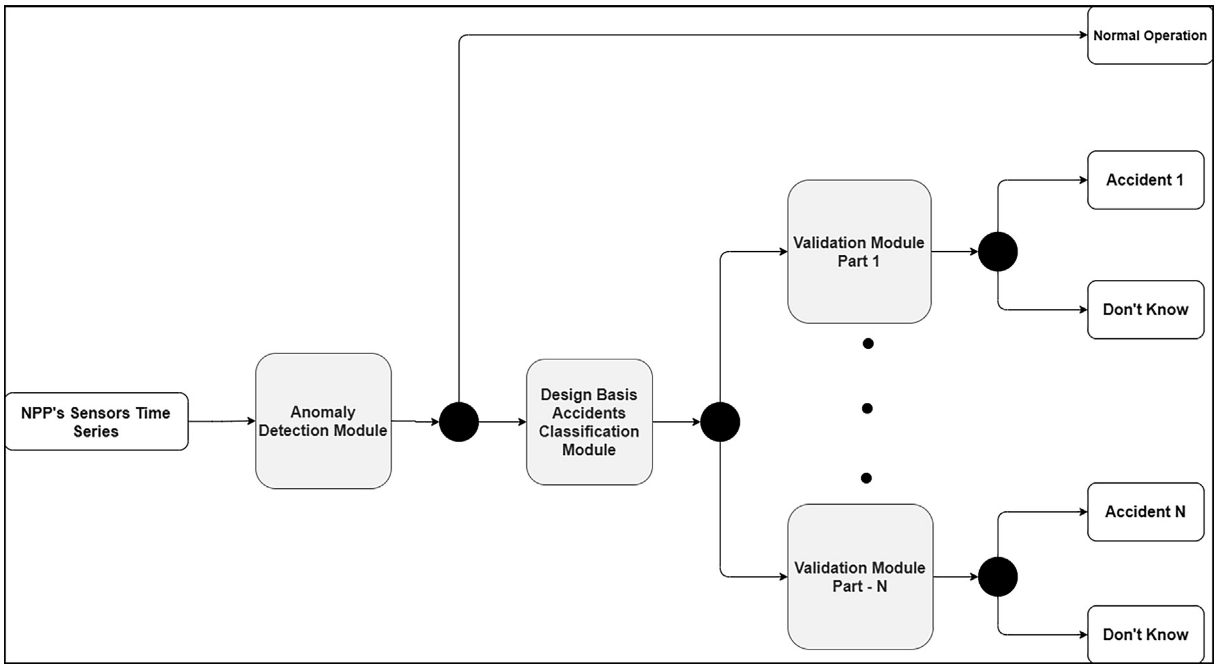


Fig. 6. System architecture diagram.

Table 1
Operational Scenarios.

ID	Scenario	Description
1	BLACKOUT	Loss of electrical power
2	BLACKSEM	Loss of electrical power without reactor shutdown
3	LOCA	Loss of coolant of the primary system
4	MEFWISEM	Main and auxiliary feed water isolation without shutdown of the reactor
5	MEFWISO	Main and auxiliary feed water isolation
6	MFWBR	Main feed water break
7	MFWBRSEM	Main feed water break without reactor shutdown
8	MFWISEM	Main feed water isolation without reactor shutdown
9	MFWISO	Main feed water isolation
10	MSTMISEM	Main steam line isolation without reactor shutdown
11	MSTMISO	Main steam line isolation
12	NORMAL	Normal operation at 100% power level
13	SGTR	Steam generator tube rupture
14	STMLIBR	Steam line break
15	TRIPREA	Shutdown of the turbine without reactor shutdown
16	TRIP	Shutdown of the reactor and turbine

Table 2
State Variables.

ID	Description	Unit
1	Core flow rate	%
2	Hot-leg temperature	°C
3	Cold-leg temperature	°C
4	Core flow rate	kg/s
5	Steam generator level (wide range)	%
6	Steam generator level (narrow range)	%
7	Steam generator level	MPa
8	Feedwater flow	kg/s
9	Steam flow	kg/s
10	Primary flow	kg/s
11	Pressurizer pressure	MPa
12	Thermal power	%
13	Nuclear power	%
14	Subcooling margin	°C
15	Pressurizer level	%
16	Average temperature	°C

as a set of 16 multivariate time series (operational scenarios), where each of these multivariate series is composed of 16 univariate time series (state variables) with size 61.

4.2.1. The data augmentation method

In the context of ANNs, especially in DNNs, the number of examples used to train models is an aspect of great importance. Considering that, the superior performance of DNNs depends on the number of examples available for training, in order to avoid overfitting, that is when the model fits exactly well to the training set, but it is not able to generalize to new data. As a result, a widely implemented strategy when working with deep learning models is the use of data augmentation methods, which are methods that significantly increase the diversity of data available for training the models, without actually collecting new data.

The use of data augmentation is widespread in the field of image processing with DNNs, with several methods available (Perez and Wang, 2017). In comparison, in the field of time series processing with DNNs, the development and application of data augmentation methods is a less widespread practice. Although, when a data augmentation strategy is applied on time series datasets, the performance of DNN models improves significantly (Wen et al., 2020).

Therefore, in order to improve the robustness of the proposed method, and at the same time, provide deep learning models with an adequate number of training examples. In this work, it is proposed the use the instrumentation noise, which is the uncertainties of measuring instruments (typically considered 1% in NPPs), as a method of data augmentation. In order to implement this data augmentation strategy, each variable, of each multivariate time series, was overlapped with white noise with normal distribution and a 1% standard deviation.

4.2.2. The sliding window technique

The sliding window technique (SWT) is a method used to facilitate the modeling and analysis of a time series. The SWT consists of using a fixed-size temporal window to divide a time series into multiple subsequences that have the same size as the window used. Thus, considering a time series T with size n and a temporal window with size w, a matrix D is constructed by sliding the window along T, placing the subsequence S_p in the p-nth line of D. So, the size of the matrix D = [(n - w + 1), w]. In order to exemplify, Fig. 7 shows the application of the SWT, using a temporal window size of 5, in a temporal sequence with size 10. As a result of the SWT application, 6 subsequences are generated.

The SWT is widely applied when modeling time series for convolutional and recurrent networks. Therefore, in this work, for the recurrent and convolutional DNN models, the SWT is also used to model the nuclear accidents time series dataset.

4.2.3. The dataset Preparation and distribution

Originally, each class (operational scenario) of the dataset was represented by one multivariate time series with size 61, meaning that each univariate time series (state variables) had 61 observations. However, it was noticed that the events really start at the third observation (2 s on the dataset). Therefore, the first step in the data modeling process was the removal of the first two observations from each multivariate time series.

After that, the second step of the modeling process was the application of the instrumental noise data augmentation method. Thus, for each operational scenario, 299 new multivariate time series were generated. At the end of this modeling step, each class of the dataset was now made up of 300 multivariate time series (the original series, plus 299 noisy series) of size 59.

The third modeling step was the implementation of sliding window technique on the augmented dataset. In this process, temporal windows of different sizes were applied to the augmented dataset. As a result, for each temporal window size applied, a modified version of the augmented dataset was generated.

Table 3 shows how the original dataset was being modified in each step of the modeling process. The first line of Table 3 shows the characteristics of the original dataset after the removal of the first two observations from each multivariate time series. The second line presents the characteristics of the dataset after the augmentation. Therefore, the value of 17,700 samples per class is obtained by multiplying the number of multivariate time series in each class, which is 300, by the size of the time series, which is 59. Similarly, the value of 283,200 for the total number of samples is obtained, by multiplying the number of classes (16), by the number of samples in each class (17700). The remaining lines of Table 3, show the characteristics of the augmented dataset after the implementation of the STW with temporal windows of different sizes. The size of the temporal window is presented at the column "Timesteps", this name is utilized because, generally, when the STW is applied in the context of ANNs, the temporal window size is called timesteps. Another characteristic, relevant to be mentioned, is that when using the STW in the context of ANNs, the dataset assumes a 3D format [number of examples, timesteps, number of variables].

Afterward, in order to prepare the datasets generated for the experiments with the DNN models, each dataset was subdivided into a training, validation, and test set. Thus, Table 4 presents the characteristics and distribution of samples for each dataset.

As the final step of the modeling phase of this study, in order to improve the training efficiency of the DNN models, the data of the datasets have been normalized according to Eq. (4).

$$X_N = \frac{\left(X - \overset{\hat{A}'}{X}\right)}{S}$$

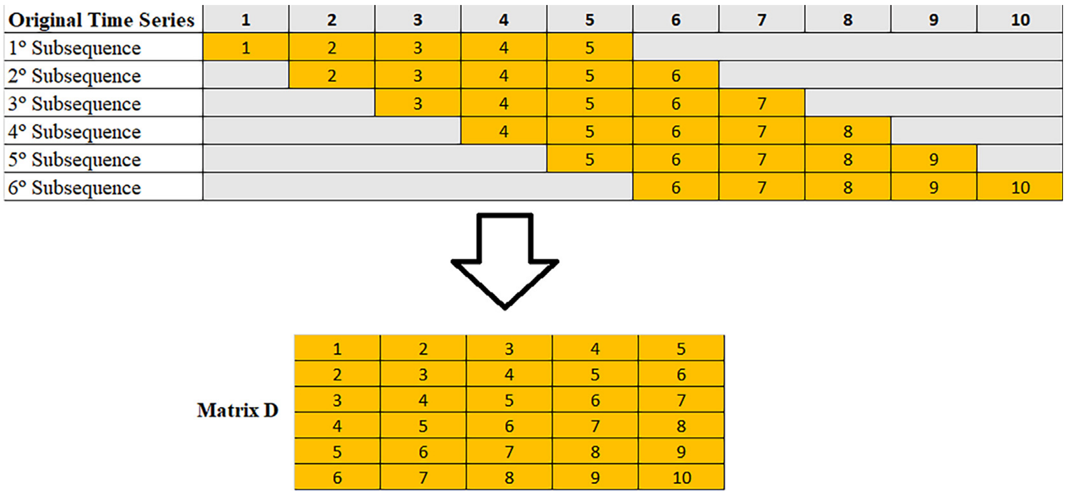
where X_N is the normalized value; X is the original value; $\overset{\hat{A}'}{X}$ is the average and S is the standard deviation.

5. Experiments and analysis of results

5.1. The experiments for the design basis accidents classification module

The choice of DNN models for experiments with the DBACM considered two criteria, the first criterion being the choice of models that in the literature have already demonstrated the ability to achieve state-of-the-art results in several TSC problems. The second criterion was the selection of DNN models, which, in the literature, demonstrated good performance in the NAC task. Thus, the following the four DNN models, cited on section "*3.1. Deep Learning Models Applied in The Nuclear Accident Classification Task*", were selected for the experiments.

Furthermore, for the experiments, all DNN models were trained, validated, and tested with the datasets presented in section "*4.2.3. The Dataset Preparation and Distribution*". However, one modification made in the datasets for the experiments was the removal of the normal operational scenario. Since, as the system diagram (Fig. 6) showed, the DBACM is responsible for the classification of DBAs, while the normal operation is identified by the ADM.



Original Time Series	1	2	3	4	5	6	7	8	9	10
1° Subsequence	1	2	3	4	5					
2° Subsequence		2	3	4	5	6				
3° Subsequence			3	4	5	6	7			
4° Subsequence				4	5	6	7	8		
5° Subsequence					5	6	7	8	9	
6° Subsequence						6	7	8	9	10

Matrix D

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9
6	7	8	9	10

Fig. 7. The sliding window technique operation.

Table 3
The datasets characteristics.

Dataset	Timesteps	N° of Classes	N° of Samples per Class	N° Total of Samples	N° of Variables	Dataset Format
Original Set	–	16	59	944	16	[944, 16]
Data Augmentation Set	–	16	17,700	283,200	16	[283200, 16]
Sliding Window – 5 Timesteps Set	5	16	16,500	264,000	16	[264000, 5, 16]
Sliding Window –10 Timesteps Set	10	16	15,000	240,000	16	[240000, 10, 16]
Sliding Window – 15 Timesteps Set	15	16	13,500	216,000	16	[216000, 15, 16]
Sliding Window – 20 Timesteps Set	20	16	12,000	192,000	16	[192000, 20, 16]
Sliding Window – 25 Timesteps Set	25	16	10,500	168,000	16	[168000, 25, 16]

Table 4
Characteristics and distribution of the training, validation and test sets.

Dataset	Subset	Timesteps	N° of Classes	N° of Samples per Class	N° Total of Samples	N° of Variables	Dataset Format
Data Augemantation Set	Training	–	16	5900	94,400	16	[94400, 16]
	Validation	–	16	5900	94,400	16	[94400, 16]
	Test	–	16	5900	94,400	16	[94400, 16]
Sliding Window – 5 Timesteps Set	Training	5	16	5500	88,000	16	[88000, 5, 16]
	Validation	5	16	5500	88,000	16	[88000, 5, 16]
	Test	5	16	5500	88,000	16	[88000, 5, 16]
Sliding Window – 10 Timesteps Set	Training	10	16	5000	80,000	16	[80000, 10, 16]
	Validation	10	16	5000	80,000	16	[80000, 10, 16]
	Test	10	16	5000	80,000	16	[80000, 10, 16]
Sliding Window – 15 Timesteps Set	Training	15	16	4500	72,000	16	[72000, 15, 16]
	Validation	15	16	4500	72,000	16	[72000, 15, 16]
	Test	15	16	4500	72,000	16	[72000, 15, 16]
Sliding Window – 20 Timesteps Set	Training	20	16	4000	64,000	16	[64000, 20, 16]
	Validation	20	16	4000	64,000	16	[64000, 20, 16]
	Test	20	16	4000	64,000	16	[64000, 20, 16]
Sliding Window – 25 Timesteps Set	Training	25	16	3500	56,000	16	[56000, 25, 16]
	Validation	25	16	3500	56,000	16	[56000, 25, 16]
	Test	25	16	3500	56,000	16	[56000, 25, 16]

Moreover, among the models, only for DRNN the sliding window technique was not used to model the dataset. For the other DNN models, the technique was used and the performance of these models was evaluated considering different numbers of timesteps.

5.1.1. Analysis of the results obtained for the design basis accidents classification module

Table 5 shows the accuracy obtained by each of the DNN models for the test set. Analysing the result, the DRNN, which has a MLP architecture, and doesn't use the sliding window technique, was the one that achieved the worst result. However, for the other DNN models, as the number of timesteps increased, the accuracy of these models also improved. In addition, Fig. 8 highlights the relationship between the number of timesteps and the performance of the models.

Although the LSTM, FCN and LSTM FCM models achieved similar results, reaching 100% accuracy with only 25 timesteps, when analysing the average accuracy in Table 5, it is possible to observe that the FCN and LSTM-FCM exceeded the LSTM, with the FCN reaching the highest average accuracy. These results are in line with those presented in the TSC literature, where the FCN also been achieving the best performance in different TSC problems.

In order to assess in more detail, the performance of the FCN, Figs. 9–11, show the confusion matrix generated by the FCN for different numbers of timesteps. Analysing the confusion matrices, starting from Fig. 9, it is possible to observe that the greatest difficulty of FCN lies in the correct classification of the accidents MEFWISEM, MEFWISO, MFWBR, MFWBRSEM, MFWISEM, and MFWISO. This is an expected result, given that some of these accidents have extremely similar behavior. However, as the number of timesteps increases, the amount of time information available for the FCN to extract from each example also increases. So, with 25 timesteps, FCN no longer misclassify these accidents and reaches 100% accuracy in the classification of nuclear accidents.

Table 5
Accuracy achieved by the DNNs models on the test set.

Timesteps	DRNN	LSTM	FCN	LSTM-FCN
1	94.18%	–	–	–
5	–	95.35%	95.56%	95.63%
10	–	96.60%	97.34%	97.24%
15	–	98.10%	98.54%	98.41%
20	–	99.28%	99.49%	99.26%
25	–	100%	100%	100%
30	–	100%	100%	100%
35	–	100%	100%	100%
40	–	100%	100%	100%
45	–	100%	100%	100%
50	–	100%	100%	100%
55	–	100%	100%	100%
60	–	100%	100%	100%
Average Accuracy	–	99.1111%	99.2443%	99.2121%

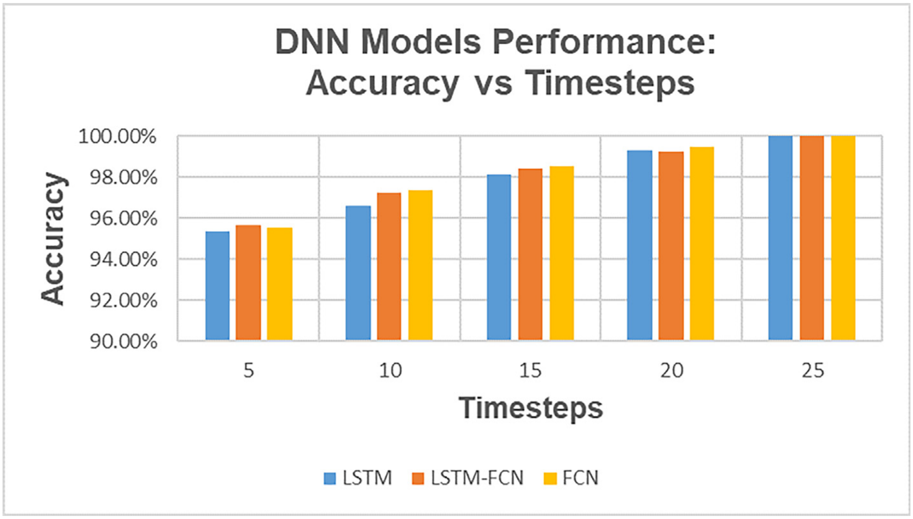
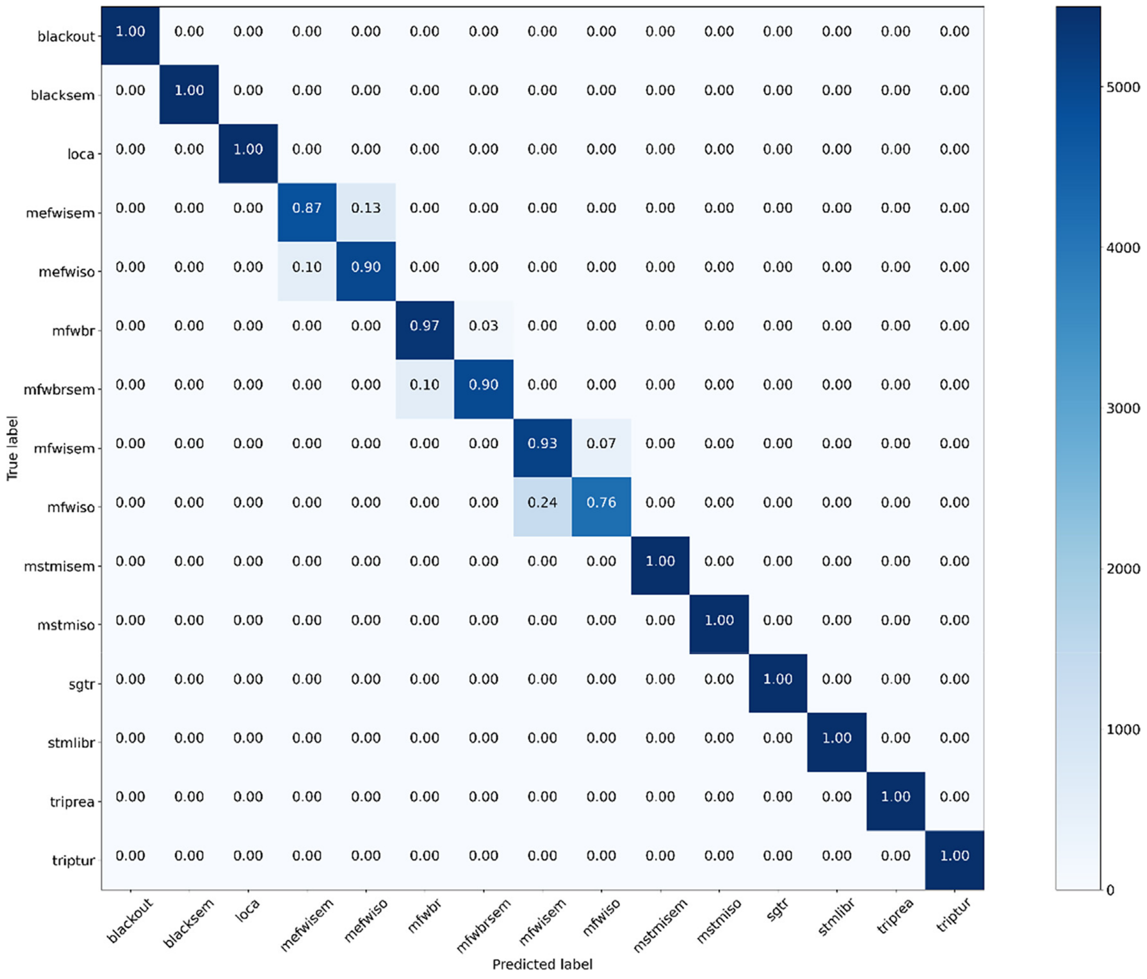


Fig. 8. Performance Evolution of LSTM, LSTM-FCN and FCN in relation to the number of timesteps.



True label \ Predicted label	blackout	blacksem	loca	mefwisem	mefwiso	mfwbr	mfwbrsem	mfwisem	mfwiso	mstmisem	mstmiso	sgtr	stmlibr	triprea	triptur
blackout	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
blacksem	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
loca	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mefwisem	0.00	0.00	0.00	0.87	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mefwiso	0.00	0.00	0.00	0.10	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mfwbr	0.00	0.00	0.00	0.00	0.00	0.97	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mfwbrsem	0.00	0.00	0.00	0.00	0.00	0.10	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mfwisem	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.93	0.07	0.00	0.00	0.00	0.00	0.00	0.00
mfwiso	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.76	0.00	0.00	0.00	0.00	0.00	0.00
mstmisem	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
mstmiso	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
sgtr	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
stmlibr	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
triprea	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
triptur	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Fig. 9. FCN confusion matrix for the test set modelled with 5 timesteps.

Considering the complexity of the assessed classification scenario, having 15 accidents and some being very similar. The ability to achieve 100% accuracy in just 25 timesteps demonstrates a good result.

5.2. The experiments for the Validation module

For the experiments for the VM, two approaches, using LSTM AEs, were evaluated to generate a “don’t know” response. The

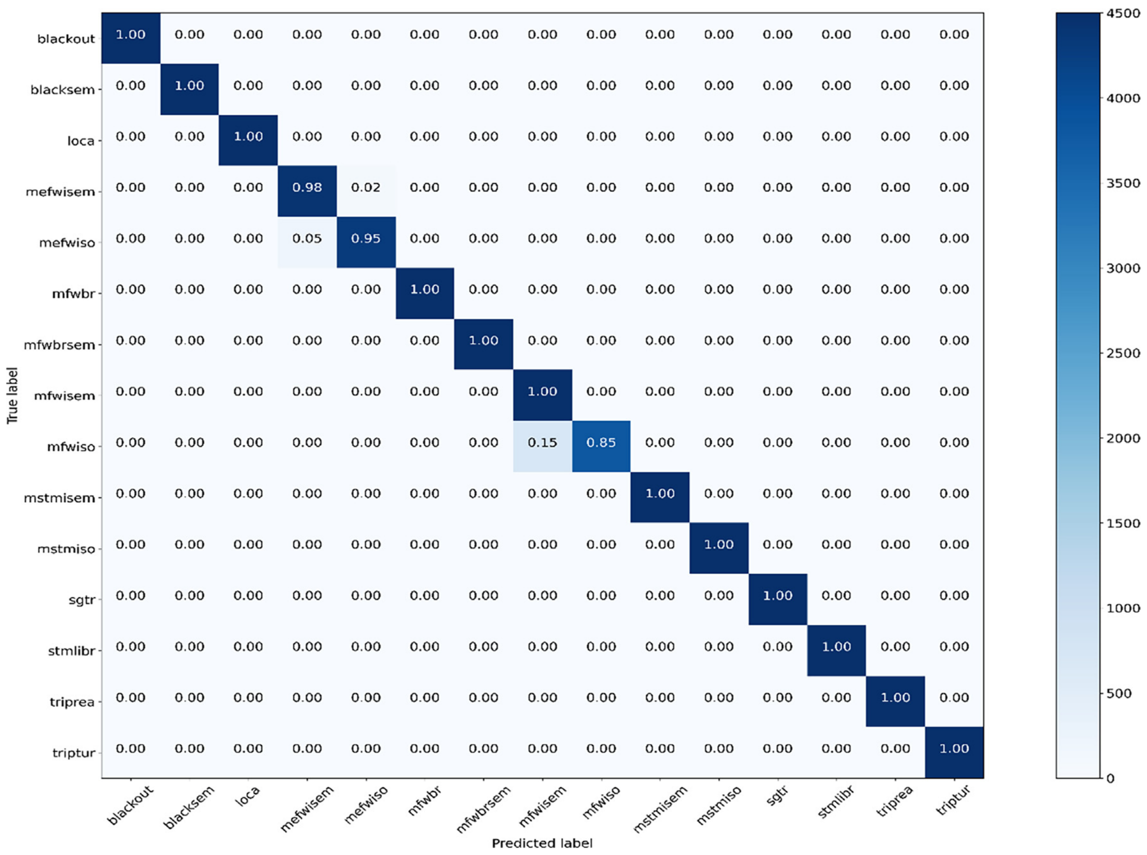


Fig. 10. FCN confusion matrix for the test set modelled with 15 timesteps.

LSTM AE model was selected for this module since its capacity for this type of task has already been demonstrated in the literature (Yang and Kim, 2020a, 2020b). Therefore, the two approaches using LSTM AE for the experiments were defined in the following way:

- **All-Classes LSTM AE Approach (AC-LSTM AE)**: It consists of training an LSTM AE model with all known classes (accidents).
- **One-Class LSTM AE Approach (OC-LSTM AE)**: It consists of training multiple LSTM AE models one for each class (accident) known.

In both approaches, the LSTM AE models were configured with: two encoding LSTM layers having respectively 128 and 64 cells, a fixed vector layer with size equals the number of timesteps defined for the input dataset, and two decoding LSTM layers having respectively 64 and 128 cells. Furthermore, as was done for the DBACM, the LSTM AEs were trained, validated, and tested with the datasets presented in section "*4.2.3. The Dataset Preparation and Distribution*", with the sliding window technique being used to model the data, and the NORMAL operational scenario being also removed.

The first test was focused in compare the performance of the two approaches. Therefore, the dataset, modeled with the sliding window technique, using 5 timesteps was used. Furthermore, since there are 15 accidents, a total of 15 experiments were carried out for each approach.

For the AC-LSTM AE approach, in each experiment, an accident was removed from the dataset and the training data from the remaining 14 accidents were then used to train an LSTM AE model. After the training, the maximum reconstruction error for the training set was used to define a threshold value. Then, to assess its capacity, the model was confronted with samples from the test set of the 14 known (trained) classes and samples from the unknown (untrained) class. With the premise that for samples of known classes, the model should generate a reconstruction error lower than the threshold, and for examples of the unknown class, it should produce an error greater than the threshold.

On the other hand, for the OC-LSTM AE approach, in each experiment, an OC-LSTM AE model was trained with the training data from only one accident of the dataset. After the training, the maximum reconstruction error for the training set was also used to define a threshold value. Then, to assess the capacity of the OC-LSTM AE model, it was confronted with samples from the test set of the known (trained) class and with samples from the others 14 accidents, with those being used as unknown (untrained) classes. The premise was the same as in the previous approach, for the samples of the known class the model should generate a reconstruction error smaller than the threshold, and for samples of the unknown classes, it should produce an error greater than the threshold.

5.3. Analysis of the results obtained for the validation module

The result, shown in Table 6, obtained from the experiments using the AC-LSTM AE approach, reveals that this approach has a great capacity to identify the known events. This fact is represented by the "Average Know" cell, which shows that an average accuracy of about 99.99% was achieved in the identification of known accidents. On the other hand, in the identification of unknown events, the performance of this approach was extremely limited, reaching an average accuracy of only 57.93%, this value is shown in the "Average Don't Know" cell.

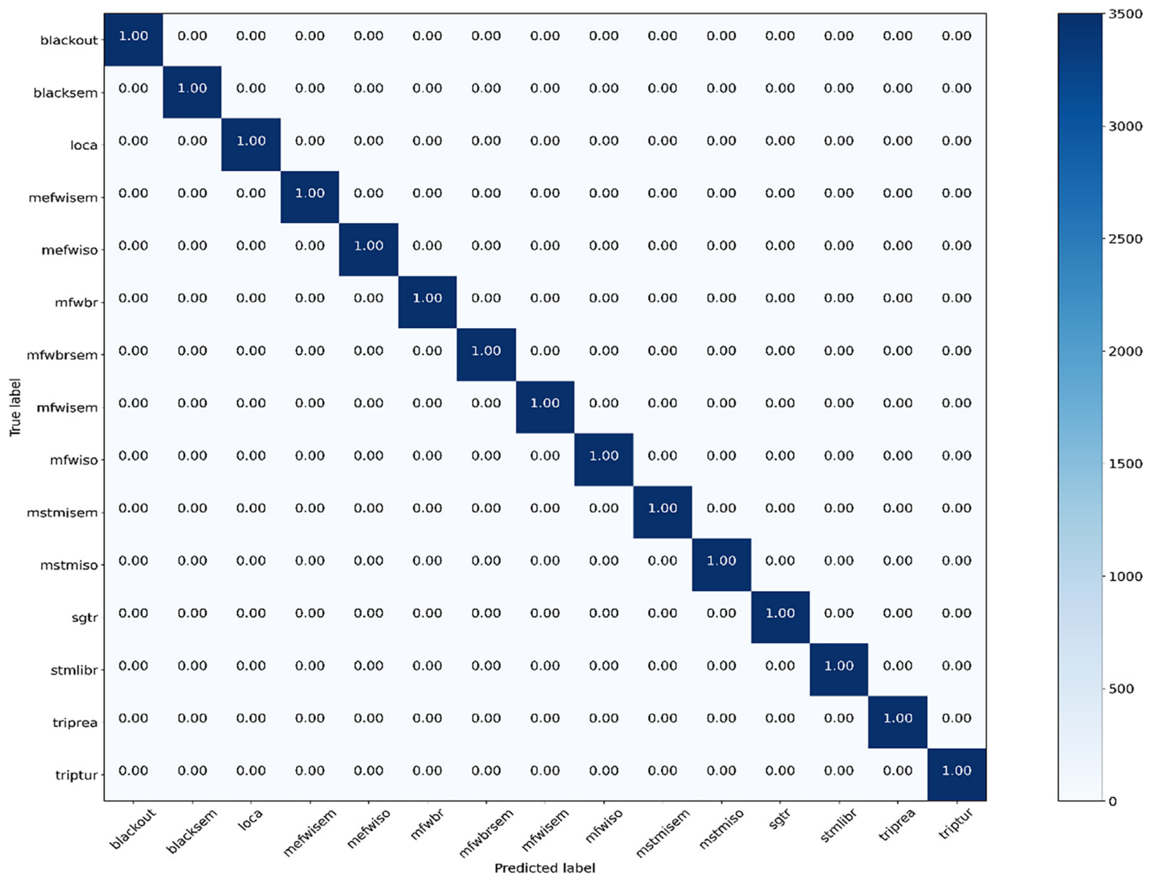


Fig. 11. FCN confusion matrix for the test set modelled with 25 timesteps.

On the other hand, Table 7 shows the results obtained from the experiments using the OC-LSTM AE approach. Unlike the AC-LSTM AE approach, which only achieved good performance in the identification of known events, the OC-LSTM AE approach obtained a good result in the identification of known events (average accuracy of 99.76%) and in the identification of events unknown (average accuracy of 99.25%).

Given the superior results achieved by the OC-LSTM AE approach, additional experiments were carried out to assess whether it would be possible to further refine its ability to identify unknown events, without negatively affecting its ability to identify known events. Therefore, for these experiments, the datasets modeled, with the sliding window technique, using 15 and 25 timesteps were used.

Tables 8 and 9 show the results obtained in the experiments for each number of timesteps. As with the DNN models used for the classification of nuclear accidents, the performance of the OC-LSTM AE approach tended to improve according to the increase in the number of timesteps. So, with 25 timesteps, the OC-LSTM-AE approach was able to achieve an average accuracy in the identification of unknown events of around 100%, and in addition, the ability to identify known events were not negatively impacted, in fact, the average accuracy in identifying known events reached 99.82%.

When comparing the results obtained using the OC-LSTM AE approach with similar ones in the literature (Kim et al., 2021; Pinheiro et al., 2020), it can be stated that the OC-LSTM AE approach achieved a state-of-the-art performance, in this problem of generating "don't know" response in the context of a NAIS. This statement is even more grounded when considering the complexity of the scenario presented, 15 nuclear accidents, with some of them being extremely similar.

5.4. The experiments for the anomaly detection module

Since the OC-LSTM AE approach demonstrated the best results in the experiments for the VM, the same approach was evaluated for the ADM, with the distinction that, since the objective of ADM is detecting when the input deviates from normality. The experiments for the ADM consisted of training an OC LSTM-AE model with training examples of the normal operation only, and after the training, the model was confronted with test samples of the normal operation and with test samples from the 15 accidents, in order to assess its capacity to detect normality and abnormality. Furthermore, for these experiments, the datasets modeled with the sliding window technique using 5, 15, and 25 timesteps and the same LSTM AE model configuration utilized for the VM was also used.

5.4.1. Analysis of the results obtained for the anomaly detection module

As Table 10 presents the OC-LSTM AE, in all the experiments conducted the OC LSTM-AE approach was able to detect the accidents as anomalies with 100% accuracy, as well as, normality detection accuracy that was also close to 100%.

5.5. Analysis of the results in relation to similar approaches in the literature

Although not yet implemented at the level of classic artificial intelligence approaches, the use of methods, based on deep learning models, to design a nuclear accident identification system to support the operators, has been gaining strength in the scientific literature of the area, therefore, it is important to highlight, in terms of results obtained, the works similar to the one proposed

Table 6
Results of experiments with the AC-LSTM AE approach using a 5 timesteps dataset.

	No blackout	No blacksem	No loca	No mefwisem	No mefwiso	No mfwbr	No mfwbrsem	No mfwisem	No mfwiso	No mstmisem	No mstmiso	No sgtr	No stmlibr	No triprea	No triptur
blackout	**86.18%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
blacksem	100%	**100%**	100%	100%	100%	99.68%	99.96%	99.96%	100%	100%	100%	100%	100%	100%	100%
loca	100%	100%	**67.69%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mefwisem	100%	100%	100%	**15.11%**	100%	100%	100%	99.98%	100%	100%	100%	100%	100%	100%	100%
mefwiso	100%	100%	100%	100%	**10.64%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbr	100%	100%	100%	100%	100%	**86.89%**	100%	99.98%	100%	100%	100%	100%	100%	100%	100%
mfwbrsem	100%	100%	100%	100%	100%	100%	**82.33%**	99.98%	100%	100%	100%	100%	100%	100%	100%
mfwisem	100%	100%	100%	100%	100%	100%	100%	**67.27%**	100%	100%	100%	100%	100%	100%	100%
mfwiso	100%	100%	100%	100%	100%	100%	100%	100%	**24.55%**	100%	99.96%	100%	100%	100%	100%
mstmisem	100%	100%	100%	100%	100%	100%	100%	99.96%	100%	**88.25%**	100%	100%	100%	100%	100%
mstmiso	100%	100%	100%	100%	100%	100%	100%	99.98%	100%	100%	**44.55%**	100%	100%	100%	100%
sgtr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**69.45%**	100%	100%	100%
stmlibr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**0.00%**	100%	100%
triprea	100%	100%	100%	100%	100%	100%	99.98%	100%	100%	100%	100%	100%	100%	**47.56%**	100%
triptur	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	99.98%	100%	100%	**78.55%**
Average Don't Know	**57.935%**														
Average Know	**99.998%**														

Table 7
Results of experiments with the OC-LSTM AE approach using the 5 timesteps dataset.

	No blackout	No blacksem	No loca	No mefwisem	No mefwiso	No mfwbr	No mfwbrsem	No mfwisem	No mfwiso	No mstmisem	No mstmiso	No sgtr	No stmlibr	No triprea	No triptur
blackout	**99.98%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
blacksem	100%	**99.93%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
loca	100%	100%	**100%**	100%	100%	100%	100%	100%	100%	100%	100%	88.51%	100%	100%	100%
mefwisem	100%	100%	100%	**99.84%**	67.24%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mefwiso	100%	100%	100%	67.31%	**99.56%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbr	100%	100%	100%	100%	100%	**99.49%**	87.33%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbrsem	100%	100%	100%	100%	100%	87.47%	**99.44%**	100%	100%	100%	100%	100%	100%	100%	100%
mfwisem	100%	100%	100%	99.07%	99.96%	100%	100%	**99.44%**	68.09%	100%	100%	100%	100%	100%	100%
mfwiso	100%	100%	100%	99.40%	99.98%	100%	100%	69.13%	**99.96%**	100%	100%	100%	100%	100%	100%
mstmisem	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.38%**	100%	100%	100%	94.35%	100%
mstmiso	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.78%**	100%	100%	100%	100%
sgtr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**100%**	100%	100%	100%
stmlibr	100%	100%	99.52%	100%	100%	100%	100%	100%	100%	100%	100%	96.71%	**99.84%**	100%	100%
triprea	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	96.76%	100%	100%	**99.85%**	100%
triptur	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.75%**
Average Don't Know	**99.25%**														
Average Know	**99.76%**														

Table 8
Results of experiments with the OC-LSTM AE approach using the 15 timesteps dataset.

	No blackout	No blacksem	No loca	No mefwisem	No mefwiso	No mfwbr	No mfwbrsem	No mfwisem	No mfwiso	No mstmisem	No mstmiso	No sgtr	No stmlibr	No triprea	No triptur
blackout	**99.76%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
blacksem	100%	**99.93%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
loca	100%	100%	**100%**	100%	100%	100%	100%	100%	100%	100%	100%	98.16%	100%	100%	100%
mefwisem	100%	100%	100%	**100%**	80.18%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mefwiso	100%	100%	100%	82.22%	**99.89%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbr	100%	100%	100%	100%	100%	**99.91%**	96.00%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbrsem	100%	100%	100%	100%	100%	100%	**99.44%**	100%	100%	100%	100%	100%	100%	100%	100%
mfwisem	100%	100%	100%	87.22%	96.67%	100%	100%	**99.96%**	82.00%	100%	100%	100%	100%	100%	100%
mfwiso	100%	100%	100%	90.67%	95.47%	100%	100%	82.87%	**99.93%**	100%	100%	100%	100%	100%	100%
mstmisem	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.98%**	100%	100%	100%	100%	100%
mstmiso	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.76%**	100%	100%	100%	100%
sgtr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.96%**	100%	100%	100%
stmlibr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	99.94%	**99.87%**	100%	100%
triprea	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	99.82%	100%	100%	**99.94%**	100%
triptur	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**100%**
Average Don't Know	**99.48%**														
Average Know	**99.89%**														

Table 9
Results of experiments with the OC-LSTM AE approach using the 25 timesteps dataset.

	No blackout	No blacksem	No loca	No mefwisem	No mefwiso	No mfwbr	No mfwbrsem	No mfwisem	No mfwiso	No mstmisem	No mstmiso	No sgtr	No stmlibr	No triprea	No triptur
blackout	**99.77%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
blacksem	100%	**100%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
loca	100%	100%	**99.94%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mefwisem	100%	100%	100%	**98.91%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mefwiso	100%	100%	100%	100%	**99.63%**	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbr	100%	100%	100%	100%	100%	**99.83%**	100%	100%	100%	100%	100%	100%	100%	100%	100%
mfwbrsem	100%	100%	100%	100%	100%	100%	**99.69%**	100%	100%	100%	100%	100%	100%	100%	100%
mfwisem	100%	100%	100%	100%	100%	100%	100%	**100%**	100%	100%	100%	100%	100%	100%	100%
mfwiso	100%	100%	100%	100%	100%	100%	100%	100%	**100%**	100%	100%	100%	100%	100%	100%
mstmisem	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.97%**	100%	100%	100%	100%	100%
mstmiso	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.94%**	100%	100%	100%	100%
sgtr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.94%**	100%	100%	100%
stmlibr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.86%**	100%	100%
triprea	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**100%**	100%
triptur	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.77%**
Average Don't Know	**100%**														
Average Know	**99.82%**														

Table 10
Results of experiments using the 5, 10, 15, 20 and 25 timesteps datasets with with the OC-LSTM AE approach.

	AE Normal – 5 Timesteps	AE Normal – 10 Timesteps	AE Normal – 15 Timesteps	AE Normal – 20 Timesteps	AE Normal – 25 Timesteps
normal	**99.98%**	**100.00%**	**99.97%**	**99.92%**	**99.91%**
blackout	100%	100%	100%	100%	100%
blacksem	100%	100%	100%	100%	100%
loca	100%	100%	100%	100%	100%
mefwisem	100%	100%	100%	100%	100%
mefwiso	100%	100%	100%	100%	100%
mfwbr	100%	100%	100%	100%	100%
mfwbrsem	100%	100%	100%	100%	100%
mfwisem	100%	100%	100%	100%	100%
mfwiso	100%	100%	100%	100%	100%
mstmisem	100%	100%	100%	100%	100%
mstmiso	100%	100%	100%	100%	100%
sgtr	100%	100%	100%	100%	100%
stmlibr	100%	100%	100%	100%	100%
triprea	100%	100%	100%	100%	100%
triptur	100%	100%	100%	100%	100%

in this study, so that similarities and differences between the approaches can be better pointed, and this way further improve the discussion of the use of deep learning models for the construction of this type of system.

In this context, Saeed et al. (2020), constructed a modular nuclear accident identification system, in which the main feature that can be highlighted was the use of two DNN models in parallel, a CNN and a LSTM, a module that was responsible for the classification of the accidents and unknown situation identification based on the comparison of the output of the two DNNs. The training accuracy of the CNN of this module was 100%, using the sliding window technique with 100 timesteps. In addition, similar to the system proposed in this work, the Saeed et al. (2020) system also had an anomaly detection module, however, this one used principal component analysis for the detections.

Despite the fact that the system proposed in this work and the work implemented by Saeed et al. (2020) are similar in terms of the use of a modular structure for the construction of the systems, there are differences in the approaches, mainly in relation to the use of only a single module to classify accidents and generate the "don't know" response, and also the fact that not all system modules use deep learning models. Therefore, approaches more in line with the system proposed in this work are that of Kim et al. (2021), Pinheiro et al. (2020), Yang et al. (2020) and Yang and Kim (2020).

The works of Yang et al. (2020) and Yang and Kim (2020), proposed a modular deep learning system, that implemented LSTM for the module of classification of accidents and a LSTM Autoencoder for the generation of the "don't know" response. In Yang and Kim (2020), in a case study with three operational scenarios, where one was used as an unknown scenario to evaluate the "don't know" response capability, an accuracy of 94.38% was achieved by the LSTM, using 30 timesteps in the sliding window technique, in the classification of the know scenarios. In addition, the LSTM Autoencoder-based approach for the "don't know" response module, although results have not been specified numerically in terms of accuracy, was demonstrated as a potential good approach to be used in this module. The work of Yang et al. (2020) followed the same configuration of modules and deep learning models for their system, the main difference was the number of operational scenarios contemplated in the study case, which were four, and the LSTM was able to achieve an accuracy of 99.84% in the classification, also using 30 timesteps in the sliding window technique.

Although Yang et al. (2020) and Yang and Kim, (2020) works demonstrated a specific module for the "don't know" response, based on an autoencoder type model as an interesting alternative, the approach was not evaluated in-depth, and, moreover, the one-class network approach, which usually demonstrates better results in solving novelty detection problems using autoencoder, has not been used. From this angle, the works Pinheiro et al., 2020 and Kim et al, (2021), analyse more in depth the capacity of multiple one class autoencoders models as alternatives for the "don't know" response generation module.

The Kim et al, (2021) modular system approach uses a LSTM for classification of the accidents, and multiple one class LSTM variational autoencoders (VAE) for the "don't know" generation. In the study case, 15 accidents were used as known scenarios and a dataset with 5 other accidents was used to evaluate the "don't know" response. For the classification of the known scenarios, the best LSTM configuration achieved an accuracy of 97.68%, using 10 timesteps in the sliding window technique. On the other hand, the one class LSTM VAE approach was able to correctly classify the known (trained) data with an accuracy of 97.57%, while the accuracy for the classification of the unknown (untrained) data was 92.37%.

The results of Kim et al, (2021) confirm the capacity of one class approach using autoencoders, however, it can be noted that in this case, the system tends to misclassify around 7.62% of the unknown data as known data, however, is preferable, in cases like this, to have a greater tendency to generate "don't know" response, because it is preferable that the system to be slightly more inclined to generate "don't know" answers for the operator than generate wrong answers. Considering this aspect, the work of Pinheiro et al, (2020) also used multiple one-class autoencoders for the "don't know" module of their system, in this case, deep autoencoders with MLP architecture, and for the module of classification of accidents, a DRNN was implemented. In the case study using 13 operational scenarios, the system was able to achieve an overall accuracy of 99.88%, in terms of correct "don't know" classifications, while retaining a substantial amount of 94.56% correct event identifications.

Taking into account the results obtained by these nuclear accident identification systems, in different study cases, it can be assessed that the construction of this type of system based on a modular structure, especially when each system module is coupled with a deep learning model, can lead to overall better performance of the system. However, an important aspect noted and approached by this study is the evaluation of the appropriated deep learning model to compose each system module. Considering, that even implementing a similar modular structure of some of these works, better results were achieved. Since, via the experiments testing different deep learning models and approaches for the system modules, a configuration formed by a fully convolutional network for classification of accidents module and one class LSTM autoencoders for the anomaly detection and for the "don't know" response modules, was able to achieve an overall accuracy of 100%, in terms of correct "don't know" identification, maintaining a high-level standard (99.82%) in the correct event identifications.

6. Conclusion

This paper presents a modular structured, deep learning-based, nuclear power plant accident identification system with anomaly detection and "don't know" response capability. In the proposed system, three modules were developed, with each module responsible to execute, through one or multiple deep learning models, one specific task. The tasks solved by the system are nuclear accident classification, "don't know" response generation, and anomaly detection, which are the main tasks that an automated NPP's accident identification system must perform, in order to efficiently fulfill its main objective, which is to provide a quick and reliable response to the NPP operators team, assisting the team in its decision-making process, in case of anomalous situations at the NPP.

Therefore, to evaluate the performance of the proposed system, the experiments conducted in this study made use of 16 simulated operational scenarios (15 accidents plus the normal operation) of one of the main operational Brazilian PWR NPPs. Furthermore, for each module of the system, comparative experiments were conducted using different contemporaneous state-of-the-art deep neural networks models and approaches.

The results achieved from the experiments were considered promising, exceeding in performance and/or complexity what had been obtained in previous works in the literature (Kim et al., 2021; Peng et al., 2018; Pinheiro et al., 2020; Saeed et al., 2020; Santos et al., 2019; Yang and Kim, 2020a, 2020b). Considering that, for the nuclear accident classification task, the best DNN model tested, a fully convolutional network, that is a state of art DNN model for time series classification, was able to achieve a classification accuracy of 100% with just 25 timesteps, in a scenario of 15 accidents, with some accidents being very similar. Moreover, in the "don't know" response task, the proposed one class LSTM Autoencoder approach, also with 25 timesteps yielded an average accuracy of 100% in terms of correct "don't know" identification, while still retaining the accuracy of 99.82% on correct event identifications. Further, in the anomaly detection task, the one class LSTM Autoencoder approach, also achieved a good result, obtaining an average accuracy of 100% in terms of correct anomaly identification, in all experiments, while maintaining an accuracy level over 99% on normal operation identification.

These experiments' results demonstrated that the proposed modular deep learning-based system is able to solve its main tasks with a high level of reliability, efficiency, and response speed. Thus, due to the performance achieved, the viability of such a system can now be evaluated in a realistic scenario, within the NPP. Therefore, the next step in this study is the assessment, *in loco*, with the NPP operators, of the system capacity and implementation viability.

CRediT authorship contribution statement

Marcelo C. Santos: Methodology, Software, Visualization, Writing - original draft, Data curation, Investigation, Formal analysis. **Claudio M.N.A. Pereira:** Writing - review & editing, Validation. **Roberto Schirru:** Conceptualization, Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to acknowledge: CAPES (Coordination for the Improvement of Higher Education Personnel), CNPq (National Council for Scientific and Technological Development) and Petrobras for financial support.

References

Alvarenga, M.A.B., 1997. Diagnóstico do desligamento de um reator nuclear através de técnicas avançadas de inteligência artificial Doctoral thesis. COPPE/UFRJ, Rio de Janeiro, Brazil.
Bagnall, A., Dau, H.A., Lines, J., Flynn, M., Large, J., Bostrom, A., Southam P., Keogh, E. 2018. The uea multivariate time series classification archive. arXiv:1811.00075.
Bartlett, E.B., Uhrig, R.E., 1992. Nuclear power plant status diagnostics using an artificial neural network. Nucl. Technol. 97 (3), 272–281.
Benuwa, B.B., Zhan, Y.Z., Ghansah, B., Wornyo, D.K., Banaseka, K.F., 2016. A review of deep machine learning. Int. J. Eng. Res. Afr. 24, 124–136.
Chalapathy, R., Chawla, S. 2019. Deep Learning for Anomaly Detection: A Survey. arXiv: 1901.03407.
Chalapathy, R., Menon, A.K., Chawla, S. 2018. Anomaly detection using one-class neural networks, arXiv:1802.06360.
Dam, R.S.F., Santos, M.C., Desterro, F.S.M., Salgado, W.L., Schirru, R., Salgado, C.M., 2021. A novel radioactive particle tracking algorithm based on deep rectifier neural network. Nucl. Eng. Technol., 1738–5733
Dau, H.A., Bagnall, A., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Keogh, E., 2019. The UCR time series archive. J. Autom. Sinica 6 (6), 1293–1305.
Desterro, F.S.M., Santos, M.C., Gomes, K.J., Heimlich, A., Schirru, R., Pereira, C.M.N.A., 2020. Development of a Deep Rectifier Neural Network for dose prediction in nuclear emergencies with radioactive material releases. Prog. Nucl. Energy 118, 103110.
Fawaz, I.H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.-A., 2019. Deep learning for time series classification: a review. Data Min Knowl Disc 33, 917–963.
Glorot, X., Bordes, A., Bengio, Y. 2011. Deep Sparse Rectifier Neural Networks. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, PMLR 15:315-323.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
Gutoski, M., Ribeiro, M., Aquino, N.M.R., Lazzaretti, A.E., Lopes, H.S. 2017. A clustering-based deep autoencoder for one-class image classification. In: 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI), pp. 1-6, IEEE.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.
Hu, J.; Shen, L.; Sun, G. 2018. Squeeze-and-Excitation Networks. arXiv:1709.01507
Karim, F., Majumdar, S., Darabi, H., Chen, S., 2017. LSTM fully convolutional networks for time series classification. IEEE Access 6, 1662–1669.
Karim, F., Majumdar, S., Darabi, H., Harford, S., 2019. Multivariate lstm-fcns for time series classification. Neural Netw 116, 237–245.
Kim, H., Arigi, A.M., Kim, J. 2021. Development of a diagnostic algorithm for abnormal situations using long short-term memory and variational autoencoder, 153, 108077.
Kramer, M.A., 1992. Autoassociative neural networks. Comput. Chem. Eng. 16, 313–328.
Kwon, D., Kim, H., Kim, J., Suh, S.C., Kim, I., Kim, K.J., 2019. A survey of deep learning-based network anomaly detection. Cluster Computing, 1–13.
Kwon, K.C., Kim, J.H., 1999. Accident identification in nuclear power plants using hidden Markov models. Eng. Appl. Artif. Intell. 12, 491–501.
LeCun, Y., 1989. Generalization and network design strategies. Technical Report CRG-TR-89-4. University of Toronto.
LeCun, Y., Bengio, Y., Hinton, G., Deep learning. 2015. Nature, 521(7553), pp 436-444.
Long, J., Shelhamer, E., Darrell, T. 2015 Fully convolutional networks for semantic segmentation. arXiv:1411.4038.
Markou, M., Singh, S., 2003. Novelty detection: A review - Part 1: Statistical approaches. Signal Process. 83 (12), 2481–2497.
Mol, A.C.A., Martinez, A.S., Schirru, R. 2002. A New Approach for Transient Identification with "Don't Know" Response Using Neural Networks. In: Ruan, D., Fantoni, P.F. (eds.). Power Plant Surveillance and Diagnostics: Applied Research with Artificial Intelligence, 253–272.
Nicolau, A.S., Schirru, R., 2017. A new methodology for diagnosis system with "don't know" response for nuclear power plants. Ann. Nucl. Energy 100, 91–97.
Pedamonti, D. 2018. Comparison of Non-linear Activation Functions for Deep Neural Networks on MNIST Classification Task. arXiv:1804.02763.
Peng, B. S., Xia, H., Liu, Y. K., Yang, B., Guo, D., Zhu, S. M. 2018. Research on Intelligent Fault Diagnosis Method for Nuclear Power Plant Based on Correlation Analysis and Deep Belief Network. Progress in Nuclear Energy, 108, 419-427.
Oza, P., Patel, V.M., 2019. Deep CNN-based Multi-task Learning for Open-Set Recognition. arXiv 1903.03161.
Pereira, C.M.N.A., Schirru, R., Lapa, C.M.F., Canedo, J.A.C., Waintraub, M., Meneses, A. A.M., Baptista, R.P., Siqueira, N.N., 2007. Particle swarm optimisation applied to nuclear engineering problems. Int. J. Nucl. Knowledge Manage. (IJNKM) 2 (3).

Perera, P., Nallapati, R., Xiang, B. 2019. OCGAN: One-class novelty detection using gans with constrained latent representations. In: Conference on Computer Vision and Pattern Recognition (2019).

Perez, L., Wang, J., 2017. The effectiveness of data augmentation in image classification using deep. Learning. arXiv, 1712.04621.

Pimentel, M.A.F., Clifton, D.A., Clifton, L., Tarassenko, L., 2014. Review: A review of novelty detection. Signal Process. 99.

Pinheiro, V.H.C., Santos, M.C., Desterro, F.S.M., Schirru, R., Pereira, C.M.N.A., 2020. Nuclear Power Plant accident identification system with "don't know" response capability: Novel deep learning-based approaches. Ann. Nucl. Energy 137.

Pinheiro, V.H.C., Schirru, R., 2019. Genetic Programming Applied to the Identification of Accidents of a PWR Nuclear Power Plant. Ann. Nucl. Energy 124, 335–341.

Rocco, S.C.M., Zio, E., 2007. A support vector machine integrated system for the classification of operation anomalies in nuclear components and systems. Reliab. Eng. Syst. Saf. 92 (5), 593–600.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M. 2018. Deep One-Class Classification. In: Proceedings of the 35th International Conference on Machine Learning, PMLR 80:4393-4402.

Saeed, H.A., Peng, M.-J., Wang, H., Zhang, B.-W., 2020. Novel fault diagnosis scheme utilizing deep learning networks. Prog. Nucl. Energy 118, 103066.

Sagheer, A., Kotb, M., 2019. Unsupervised pre-training of a deep LSTM-based Stacked autoencoder for multivariate time series forecasting problems. Sci. Rep. 9 (1), 1–16.

Santos, M.C., Pinheiro, V.H.C., Desterro, F.S.M., Avellar, R.K., Schirru, R., Nicolau, A.S., Lima, A.M.M., 2019. Deep rectifier neural network applied to the accident identification problem in a PWR nuclear power plant. Ann. Nucl. Energy 133, 400–408.

Schmidhuber, J., 2015. Deep learning in neural networks: an overview. Neural Networks 61, 85–117.

Srivastava, R.K., Greff, K., Schmidhuber, J. 2015. Training Very Deep Networks. Advances in Neural Information Processing Systems 28 (NIPS 2015), p. 1–9.

Susto, G.A., Cenedese, A., Terzi, M. 2018. Chapter 9 - Time-Series Classification Methods: Review and Applications to Power Systems Data. Big Data Application in Power Systems. p. 179–220.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. Adv. Neural Inf. Process. Syst. 4, 3104–3112.

Teixeira, T.P., et al. 2020. Determination of eccentric deposition thickness on offshore horizontal pipes by gamma-ray densitometry and artificial intelligence technique. v. 165, p. 109221.

Vasilev, I., Slater, D., Spacagna, G. 2019. Python Deep Learning -Second Edition. 2nd Revised edition ed.

Wei, Q., Ren, Y., Hou, R., Shi, B., Lo, J.Y., Carin, L. 2018. Anomaly detection for medical images based on a one-class classification. In Medical Imaging 2018: Computer-Aided Diagnosis, volume 10575, 105751M. International Society for Optics and Photonics.

Wen, Q., Sun, L., Song X., Gao, J., Wang, X., Xu, H. 2020. Time Series Data Augmentation for Deep Learning: A Survey. arXiv:2002.12478.

Wang, Z., Yan, W., Oates, T. 2017. Time series classification from scratch with deep neural networks: A strong baseline. Institute of Electrical and Electronics Engineers Inc., p.p 1578–1585.

Yang, J., Lee, S., Kim, J. 2020. Nuclear Power Plant Accident Diagnosis Algorithm Including Novelty Detection Function Using LSTM. In: Ahram T. (eds) Advances in Artificial Intelligence, Software and Systems Engineering. AHFE 2019. Advances in Intelligent Systems and Computing, vol 965. Springer, Cham.

Yang, J., Kim, J., 2020abababab. Accident diagnosis algorithm with untrained accident identification during power-increasing operation. Reliab. Eng. Syst. Saf. 202, 107032.

Zhou, C., Paffenroth, R.C. 2017. Anomaly Detection with Robust Deep Autoencoders. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 13-17, Halifax, NS, Canada.

Zhou, Y., Arpit, D., Nwogu, I., Govindaraju, V. 2014. Is Joint Training Better for Deep Auto-Encoders?, arXiv:1405.1380.