CORTEX

Research and Innovation Action (RIA)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 754316.

Start date : 2017-09-01 Duration : 48 Months
http://cortex-h2020.eu

Development of machine learning techniques and evaluation of analysis results

Authors : Pr. Kollias STEFANOS (University of Lincoln), Andreas Stafylopatis (ICCS-NTUA) Georgios Leontidis (UoL) George Alexandridis (ICCS-NTUA) Tatiana Tabouratzis (UoP, ICCS-NTUA) Aiden Durrant (UoL)

CORTEX - Contract Number: 754316

Project officer: Foivos MARIAS

Document title	Development of machine learning techniques and evaluation of analysis results
Author(s)	Pr. Kollias STEFANOS, Andreas Stafylopatis (ICCS-NTUA) Georgios Leontidis (UoL) George Alexandridis (ICCS-NTUA) Tatiana Tabouratzis (UoP, ICCS-NTUA) Aiden Durrant (UoL)
Number of pages	40
Document type	Deliverable
Work Package	WP03
Document number	D3.4
Issued by	University of Lincoln
Date of completion	2019-08-12 19:28:06
Dissemination level	Public

Summary

This document outlines and describes the development of deep neural network architectures and other machine learning techniques for the unfolding of reactor transfer functions from in-core and ex-core neutron detectors, developed in CORTEX Workpackage 3, mainly in Task 3.3. The techniques developed utilise simulated modelling of the induced neutron flux of perturbations to classify and localise perturbation types and their sources.

Approval

Date	By
2019-08-12 19:28:56	Pr. Kollias STEFANOS (University of Lincoln)
2019-08-13 08:17:30	Pr. Christophe DEMAZIERE (Chalmers)

Table of Contents

1 Introduction 5

2 Summary of Developments 6
2.1 Extraction of model parameter values from data 6
2.2 Efficient data analysis by training deep neural networks or using pre-trained networks through transfer learning 6
2.2.1 Frequency Domain 6
2.2.2 Time Domain 9
2.2.3 Time and Frequency Domain 10
2.2.4 Using CNNs and Wavelet Scaleograms 10
2.2.5 Using parametric approaches and neural networks 11
2.3 Preliminary analysis of plant data and adaptation 11

3 Key Developments/Contributions 12
3.1 A Deep Learning Approach to Anomaly Detection in Nuclear Reactors 12
3.1.1 Data Pre-Processing 12
3.1.2 2-D Convolutional Neural Network 14
3.1.3 Localisation through Clustering 15
3.1.4 Denoising Autoencoder 17
3.2 Towards a Deep Unified Framework for Nuclear Reactor Perturbation Analysis 18
3.2.1 Data Pre-Processing 18
3.2.2 3-D CNN 20
3.2.3 LSTM-RNN 21
3.2.4 Combined Approach 23
3.3 3D Convolutional and Recurrent Neural Networks for Reactor Perturbation Unfolding and Anomaly Detection 24
3.3.1 Large-Scale Data Handling 24
3.3.2 Densely Connected 3-D CNN 24
3.3.3 Time Domain Extension 25
3.4 Convolutional Networks based on Scaleograms for Reactor Perturbation Unfolding and Anomaly Detection 26
3.5 Machine learning techniques for predicting detector malfunctioning 31

4 Conclusions 36
4.1 Future Work 36

5 References 37

6 Annexes 39
6.1 A Deep Learning Approach to Anomaly Detection in Nuclear Reactors. Supporting Material 39
6.1.1 Normalised Cross-Correlation (NCC) 39
6.1.2 DAE Visualisations 39
6.2 Towards a Deep Unified Framework for Nuclear Reactor Perturbation Analysis: Supporting Material 40
6.2.1 3-D Convolutions 40
6.2.2 Grid Search for Loss Weight Coefficients 40

Index of Tables

Table 1 Results of the CNN Inception-v3 Unfolding 15
Table 2. Settings and results of the DAE experiments 18
Table 3. Scenarios provided in deliverables 2018.04.12-Version v2 and 2018.06.04-Version v2 with their assigned Binary Class ID for multi-label classification 19
Table 4. Results of the Frequency Domain 3D CNN experiment for perturbation classification and localisation regression. (*) Marks combined perturbation scenarios 21
Table 5. All Scenarios with their assigned Binary Class ID for multi-label classification 26
Table 6. The mean fold accuracy for each type of sensor per number of splits 30
Table 7. Weighted accuracy for both in-core and ex-core sensors for various signal-to-noise ratios 30

Table of Figures

Figure 1. 2-D array output of the unrolling procedure. (a) Signal Phase (b) Signal Amplitude 13
Figure 2. Visualisation of the volumetric splitting. Left: Twelve Voxels. Right: Forty-eight Voxels 13
Figure 3. t-SNE visualisation of k-means (k = 4) of the seventh block; the obtained training set clusters are (a-b) and the test set predictions are shown in (c-d) 16
Figure 4. Depiction of DAE architecture used to solve the corrupted signal reconstruction problem 17
Figure 5. Description of detector locations for the signals used in training, validation and test of deep LSTM network model in classifying different types and combinations of time domain perturbations 22
Figure 6. Unified framework for time and frequency domain perturbation type classification and location regression, using a LSTM network at the top for time domain signals and a 3D CNN below for frequency domain signals; both networks output 512 dimensional latent variable representations of their inputs, with their flow being controlled by XOR gate logic and a switch (which is activated for perturbation coordinate regression in the frequency domain) 23
Figure 7. Adapted Densely Connected 3-D CNN for Classification and Regression of Perturbation Types and Source from a Simulated Core Volume; the Dense Block is shown below 25
Figure 8. A preprocessed (i.e. detrended and resampled) signal, partitioned into k=8 parts 28
Figure 9. The CNN architecture employed for classifying the scaleograms 28
Figure 10. Input signal for different signal-to-noise ratios 29
Figure 11. Performance dropoff for different signal-to-noise ratios 31
Figure 12. The original three *indet1-2-3* signals 32
Figure 13. The pairwise differences of the three signals, *indet1-2-3* (after scaling/normalization) 33
Figure 14. Examples of drift 33
Figure 15. Examples of fluctuations 34
Figure 16. Examples of intermittencies 34
Figure 17. Examples (a-c) of reconstruction provided by DAE when: 75%, 50%, 25% of sensors were used; in each case: left: Original signal, middle: Obscured signal, right: Reconstructed signal 39

Figure 18. Weight coefficient grid search for the 3D-CNN classification and regression losses; coefficient 0.3 for classification and 0.7 for regression yielded the best performance 40

Abbreviations

APSD	Auto Power-Spectral Density
CPSD	Cross Power-Spectral Density
CNN	Convolutional Neural Network
DAE	Denoising Autoencoder
DNN	Deep Neural Network
FA	Fuel Assembly
GAP	Global Average Pooling
GRNN	General Regression Neural Network
LSTM	Long Short-Term Memory
MP	Missing Part
NCC	Normalized Cross-Correlation
ND	Neutron Detector
NF	Neutron Flux
NN	Neutron Noise
PA	Polynomial Approximation
PSI	Paul Scherer Institute
PWR	Pressurized Water Reactor
RNN	Recurrent Neural Network
OL	Oscillation Large
OS	Oscillation Small
OT	Oscillation & Trend
SPS	Parametric Splines
TL	Trend Large
TS	Trend Small

Summary

This document outlines and describes the development of deep neural network architectures and other machine learning techniques for the unfolding of reactor transfer functions from in-core and ex-core neutron detectors, developed in CORTEX Workpackage 3, mainly in Task 3.3. The techniques developed utilise simulated modelling of the induced neutron flux of perturbations to classify and localise perturbation types and their sources.

1 Introduction

The EU project CORTEX involves the development of monitoring techniques and experimental validation that allows detecting anomalies in nuclear reactor cores, such as abnormal vibrations of fuel and core internals, flow blockage, coolant inlet perturbations, etc. The monitoring of the so-called neutron noise (fluctuations in neutron flux recorded by in-core and ex-core neutron instrumentation), on which the methods in this project are based, is a non-intrusive technique.

The current Deliverable presents the CORTEX WP3 contribution towards the following objective:

"To develop and use machine learning data analysis methodologies for inverting the reactor transfer function and recovering the anomaly responsible for the observed fluctuations. Emphasis is put on situations where the in-core and ex-core instrumentation is very scarce."

The aim of the deliverable is to utilize machine learning techniques for the unfolding of reactor transfer functions from limited number of neutron flux detectors located throughout the core [1] - [2]. This is a challenging task, examining the potential for machine learning to learn the underlying functions within the reactor core from limited information. Machine learning requires a large amount of training data to 'teach' the algorithm to recognize the necessary features constructing the input signals, therefore, simulations of the neutron flux readings induced by core perturbations provide the required amounts of training data. Two simulations have been developed for this purpose providing data in both the frequency and time domain, CORE SIM [3] produced by Chalmers and SIMULATE-3K [4] by PSI respectively.

The machine learning methodologies presented are based on recent state-of-the-art developments in deep learning and deep neural networks (DNNs). Specific progress was made for the estimation of the type and position of abnormal fluctuations and perturbations, using classification and regression analysis with 2-D and 3-D deep neural networks. Very promising results have been obtained when dealing with data simulated in the frequency, as well as in the time domain, which will be extended towards real plant data. Moreover, the use of artificial neural networks and other machine learning techniques has been also examined for detection of abnormalities in data simulated in the time domain.

Three papers have been presented to the IEEE WCCI/IJCNN and SSCI Conferences in 2018 [13], two of them co-authored by UoL and Chalmers/PSI groups and one co-authored by ICCS-NTUA and PSI. Additionally, in 2019, a poster by UoL has been presented and awarded in the FISA conference and a paper by ICCS-NTUA was submitted to ICAAI Conference in 2019.

2 Summary of Developments

2.1 Extraction of model parameter values from data

The work that has been accomplished focuses on designing and developing Deep Neural Network (DNN) models that are able to analyse the generated simulated data so as to classify and localise anomaly sources from induced neutron noise readings. Identification of such DNN architectures and parameters will be examined and used next for the analysis of real plant data in WP4 of the CORTEX project.

Analysis of the developed DNNs will continue to be accomplished in the last few months of WP3, so as to define the best architectures to be used for real plant data analysis. As more real plant data will become available, their analysis will permit us to transfer the acquired knowledge from simulated data case to real plant data analysis in the most appropriate way.

2.2 Efficient data analysis by training deep neural networks or using pre-trained networks through transfer learning

The approaches we developed can be decomposed into two distinct categories dependent on the domain of the data generated, for a variety of scenarios, from the two simulators described in Deliverables D3.1 and D3.2. The first refers to data generated in the frequency domain by Chalmers University of Technology. The second refers to data generated in the time domain by Paul Scherer Institute. Distinctly different DNNs have been developed and tested for analysing these types of data, as described in the following.

2.2.1 *Frequency Domain*

The frequency domain datasets provide large amounts of data, with different perturbation types coming from every possible source location. Furthermore, since the generated signals cover the whole three-dimensional (3-D) volume of the simulated core, it was possible to also analyse them in the form of two-dimensional (2-D) image slices. This allowed the employment of a variety of powerful feature extraction techniques, resulting in effective DNN learning and anomaly detection. The provided data have, in most cases, a size of 32x32x26 (a size of *32x32x34* resulted sometimes). Since frequency domain data take complex values, each signal is decomposed into its amplitude and phase information, which are then analysed using deep learning techniques.

2.2.1.1 2-D Pretrained Convolutional Neural Networks

The first of the approaches that we developed was based on deep Convolutional Neural Networks (CNNs), aiming to perform spatial feature analysis of the 3-D volumes of the induced neutron noise. A variety of successful CNNs has been developed the last few years for analysis of 2-D images. To take advantage of these developments, a conversion procedure was devised to provide the 3-D volumes in a more conventional 2-D format. More specifically, the 3-D volume was unrolled into a series of 2-D ‘slices’ to manipulate and utilise more common-place deep learning approaches, allowing the learning of spatial feature representations. Using pre-trained DNNs, successfully developed for other object analysis tasks, as prior knowledge, and transferring this knowledge to initialise our DNN architectures was the basis of our approach.

This 2-D unrolling procedure was performed for both the amplitude and phase of the complex signal. This resulted in two 2-D matrices, each containing slices of the whole *32x32x26* volume (as shown in Figure 1). The two matrices were concatenated to form a three-channel input, as is required in the pre-trained CNNs. We selected to duplicate the amplitude information, thus, the first two channels were identical, containing the amplitude information; the third channel contained the phase information.

Firstly, a pre-trained, Inception-v3, CNN architecture has been utilised with its weights transferred to a respective CNN architecture which was then fine-tuned with the CORTEX frequency domain data. The obtained results are shown in Table 1, as will be described in the following Section. This approach classified the perturbation type into the three considered classes and localised the perturbation source to both twelve and forty-eight coarse voxel subsection representations of the core, with very high accuracy. In particular, this experiment showed the capability of machine learning methods to unfold the reactor transfer function from noise information provided by CORE SIM, achieving 99.9% accuracy in classification.

To complicate the problem and make the input more realistic, the input was corrupted in two ways. The first involved the addition of white Gaussian noise at signal-to-noise-ratios (SNR) equal to 1 and 3; the second obscured the signals by setting 25%, 50%, and 75% of the signal values equal to zero, i.e., reducing the amount of provided information. Furthermore, the amount of training data was lowered from 75% to 25%, reducing the amount of available training data in this case. Despite of these corruptions to the training procedure, the network still achieved very good classification accuracy and noise localization ability.

A second experiment was then performed, utilizing the information learnt from the CNNs to cluster the extracted representations, obtaining a finer resolution for noise localization within the voxel representation of the core. A combination of K-means clustering and K-Nearest Neighbor approaches were implemented, using the feature representation vector from the final layer of the CNN, rather than whole data input. This approach resulted in excellent performance with the perturbation source being localized to a single point accuracy within the finer resolution.

Next, a third experiment was based on the use of a denoising autoencoder; this type of CNN was trained to reconstruct the signals from their corrupted forms, thus producing a complete representation of the input. The reconstruction was evaluated by measuring the normalized cross correlation (NCC) metric between the target (original signal) and the reconstructed one. The ncc metric ranges between -1.0 (completely differing) to +1.0 (perfectly matching). The reconstruction of the corrupted signal achieved a 0.991 NCC value, showing the very good performance of the proposed CNN denoising autoencoder.

2.2.1.2 3-D Convolutional Neural Networks

We then trained 2-D CNNs from scratch, without using a pre-trained CNN, using the CORTEX data. The results were also very good as shown in the following Section. As a consequence, we considered the possibility to use and train 3-D CNNs from scratch as well. Utilising 3-D CNNs allows learning of spatial relationships across all dimensions of the core, with the potential to provide more accurate localisation across the z-axial dimension. Given the nature of the signal being generated in large 3-D volumes and the addition of hypothetical perturbation scenarios originating anywhere within the core, using 3-D DNNs could unfold the transfer function of the generated core mesh.

Signals were provided in the same form as above, i.e., as *32x32x26*, complex values. These were decomposed into their amplitude and phase components, then concatenated together to form a volume of size 64x64x26; zero padding was used in the third dimension to obtain these signal dimensions. Considering the high performance achieved in the previous study, we focused on the experiment with obscured signal values. To implement this and obscure the signals, a mask of size 32x32 was generated maintaining only 20% of the volume's information, with the remaining elements being set to a zero value. This mask was applied axially along the third dimension of the original signal ensuring that the same elements were removed across all samples.

The proposed solution addressed two tasks:

- classification of perturbation type (three classification tasks)
- regression, in which the signal is unfolded, identifying (*i, j, k*) coordinates from where the perturbation source originated in the simulated core mesh.

The approach outperformed the previous unfolding approach which used a coarser mesh combined with clustering, resulting in lower mean squared error (MSE) and mean absolute error (MAE) between the target and predicted coordinates. The classification of perturbation type also remained very high, whilst considering only 20% of the measurements of the original *32x32x26* volume.

Following this approach, a new, vastly larger dataset was provided to us, consisting of nine perturbation types (including differing modes of vibration) and a finer *32x32x34* core mesh modelling boundary sources, providing perturbations at every possible location within the core. This resulted in a greater challenge for the network to classify and localise more classes to more locations.

To achieve this, we extended the proposed 3-D CNN architecture outlined above. As was described previously, the volumetric complex signal was first decomposed into its amplitude and phase components. However, in this approach the two volumes were concatenated along a fourth dimension resulting in a volume of *2x32x32x34*. Furthermore, instead of corrupting the signals through setting a percentage of them to zero, we focused solely on 48 in-core locations and 8 ex-core locations; these correspond to the 56 neutron flux detectors usually utilized in real plant cores. This results in far less readings than previous approaches; in fact we used, instead of 5,325, just 56 simulated detectors. Additionally, the complex signals have been further pre-processed before being decomposed, with the auto-power spectral densities (APSD) and cross-power spectral densities (CPSD) being calculated and used, so as to align with real plant readings.

This sparse volumetric representation was then fed as input to an extended 3-D, densely connected, CNN, where the classification and regression tasks were performed through the previously described architecture, with extensions made for the greater number of classification tasks. Densely connected 3-D CNNs were implemented and used in this case, due to their ability to allow for the greater flow of information / gradients throughout the model architecture, resulting in a greater transportation of knowledge of the high and low-level features extracted by the network.

Overall, this approach achieved highly accurate classification results, performing similarly to the previously described ones, while classifying in more perturbation types, and regressing to a MAE of ≈0.3 coordinate positions or ≈4cm within the *≈4m x 4m x 4m* core volume. Please note that all of these results were achieved with just 56 core detectors strategically spatially placed within the core, solidifying the capabilities of machine learning techniques to unfold reactor transfer functions from limited core information.

2.2.2 *Time Domain*

Signals produced in the time domain were provided in a successive way, by increasing the number of scenarios and the complexity of the scenarios. The first set of data included 100 seconds of simulated neutron flux readings from 48 in-core detectors and 8 ex-core detectors. The readings were measured at a sampling rate of 10 Hz producing 56 vectors (one per detector) of length 10001 time-steps for a given scenario. The format of these signals was replicated across all time series data. It provided a total of fifteen scenarios each of which with the above mentioned *56x10001* format; these scenarios contained noise induced from 5x5 central fuel assembly vibrations, coolant flow fluctuations, and coolant temperature fluctuations. The fifteen scenarios included vibrations with differing modes, amplitudes, and then combinations of scenarios. A table of these combinations is shown in Table 3.

The signals first underwent signal re-sampling to reduce their length into more appropriate sizes, while augmenting the signals to produce more training data. The procedure involved the application of sliding windows of 100 time-steps in width and step strides of 5 time-steps. Furthermore, these signals were corrupted with the addition of white Gaussian noise at signal-to-noise-ratios 5 and 10; the intuition behind this was to test the sensitivity of our model to noisy signals.

Given the sequential nature of the data, recurrent neural networks (RNN) were chosen with the premise to extract meaningful temporal features from the signals. RNNs are a special kind of neural network designed specifically for sequential information, using learnable memory gates to extract information with relevance to previously seen time-steps. More specifically, long short-term memory (LSTM) RNNs have been implemented for their known ability to model long-term dependencies within the lengthy 100 time-step signals.

To extract meaningful representations from the signals, two stacked LSTM layers had been implemented, each comprising of 512 neurons. The LSTM took as input one 100 time-step signal from one detector at a time, aiming to classify the type of perturbation. Given that the scenarios contained combinations of perturbations, the model aimed to classify each perturbation individually, resulting in a multi-class, multi-label problem.

In the performed experiments, the LSTM model demonstrated a high classification accuracy of 97% on the clean signals, 81% on signals corrupted with SNR=10 and 77% on signals corrupted with SNR=5. Please note, these classification results are for one second of detector readings at one given detector location. These results illustrate the potential of machine learning to unfold reactor transfer functions within the time domain.

Following this, we were provided with a greater number of simulation scenarios, increasing the number to a total of twenty-seven different scenarios, including those from the previous case. The new scenarios included a vibration of the central cluster of fuel assemblies in the Y-direction, in addition to further combinations of scenarios. The same network as aforementioned, has been employed resulting in an accuracy of 93%, 82% and 80% for clean signals, and for signals corrupted by noise at SNR=10, and SNR=5 respectively.

The latest dataset provided the ability of localisation of fuel assembly vibrations; 5 scenarios have been generated with X-direction fuel assembly vibrations of differing modes and amplitudes, including combination with coolant flow and temperature fluctuations. Each scenario contained 193

different source locations – one for every (*i, j*) location within the simulated core mesh – each of which provided 100 second measurement readings as in previous scenarios. However, to localise the source, spatial relationships between detectors needed to be considered. Therefore, the LSTM was modified to receive as input a matrix of *56x100 inputs;* all detectors for a 100 time-step window were fed into the LSTM-RNN for classification and regression of the *i, j* location.

The output of the LSTM-RNN resembled that of the 3-D CNN for classification of the perturbation type and the regression of the coordinates for the perturbation source. The results show that on clean signals the classification accuracy is very high at 99%; this seems consistent with the fact that all detectors were utilized for fewer classification types. More importantly, the MAE of the localisation of the source is ≈1.3 coordinate positions between the prediction and the target, further showing the capability of unfolding within the time domain.

2.2.3 *Time and Frequency Domain*

A combined approach fusing both data types has also been implemented. It is known that, due to the nature and differences of the simulators, the respective signals cannot be used completely interchangeably. Therefore, a shared feature representation has been utilized. Frequency domain signals were fed to the 3-D CNN, and time signals fed to a stacked LSTM-RNN, as was previously described. However, both outputs of these networks produce a 512-dimensional vector representing the initial signals. These output layers were then combined, fusing the representations of the two data domains. This layer was consequently fed to a layer responsible for classification of all possible perturbation types across both domains. If a perturbation type was identified from the frequency domain, the network passed the 512-dimensional representation to a regression layer for the unfolding and subsequent localisation of the perturbation coordinates (*i, j, k*).

Two papers and one poster presenting in more detail the above developments have been presented at the 2018 IEEE International Joint Conference on Neural networks (IJCNN/WCCI 2018, Brazil, July 2018), at the 2018 IEEE Symposium Series in Computational Intelligence (SSCI 2018, India, November 2018) and FISA 2019 (FISA / EURADWASTE 2019, Romania, June 2019). More technical details and further rationale on the above developments are elaborated in the *Key Developments,* in Section 3 of this Deliverable.

2.2.4 *Using CNNs and Wavelet Scaleograms*

Experiments have been also performed using features provided by the application of advanced signal processing methods to the neutron flux signals. A method has been developed which implements this through the application of the discrete wavelet transform (DWT) to the signals and the generation of corresponding wavelet scaleograms [14]. Subsequently, those scaleograms are provided to a CNN that learns to classify them to the different perturbation types. The method is described in detail in Section 3 below. A related paper has been recently submitted to the ICAAI Conference.

The proposed methodology was evaluated on data generated by the SIMULATE-3K tool to model some basic types of perturbations occurring in nuclear reactors. The tool was built to simulate the steady state operation of a Pressurized Water Reactor under specific scenarios. These included phenomena such as fuel assembly vibrations (both alone and in groups), as well as coolant flow and temperature oscillations. Each of these was simulated for different frequencies and amplitudes.

A CNN architecture with 6 layers and around 2.5 million parameters was selected (as shown in Figure 11 of Section 3). Various experiments have been performed with clean and noisy signals. Results are presented which indicate the ability of the method, based on in-core (ex-core) detectors to classify with high accuracy, up to about 99 (83) % at SNR equal to 10, or 96 (78) % at SNR equal to 1.

2.2.5 *Using parametric approaches and neural networks*

Three methods have been also used and compared for predicting anomalous neutron signals obtained by malfunctioning detectors, when dealing with time-domain generated signals. These were the parametric Polynomial Approximation, the semi-parametric Splines and the non-parametric General Regression Neural Networks.

The main target was to achieve a high level of detection accuracy, while minimizing the computational complexity of the used validation procedure. To this end, the minimum possible number of neutron noise signals was sought which – combined with the shortest time window – allowed swift, reliable and accurate anomaly source identification (classification and localization).

Results are presented in Section 3 (shown in Tables 8-13). A paper presenting this method has been presented at the 2018 IEEE Symposium Series in Computational Intelligence (SSCI 2018, India, November 2018).

2.3 Preliminary analysis of plant data and adaptation

The work on preliminary analysis of plant data started in September 2018. The current research has focused on first attempts to use the above developments for the monitoring of real plant data provided via GRS. Within the next few months we will focus on our developments both in the time and frequency domain to adapt and transfer the acquired knowledge to real plant data.

3 Key Developments/Contributions

This section outlines the key developments and contributions achieved within CORTEX WP3 Task 3.3, following the descriptions made in Section 2. The key developments outline the technical details and the rationale behind the techniques that we developed and used in CORTEX.

3.1 A Deep Learning Approach to Anomaly Detection in Nuclear Reactors

The proposed developments provide a novel 2-D convolutional neural network for the classification of perturbation types and unfolding of the reactor transfer function for the localisation of the perturbation source. The work utilises simulated data provided by Chalmers in the frequency domain and proposes a transfer learning approach, clustering, and denoising methods for effective data analysis. These contributions were presented at the 2018 IEEE International Joint Conference on Neural networks (IJCNN/WCCI 2018, Brazil, July 2018) [5].

The contributions mainly focus on the use of 2-D convolutional neural networks for the unfolding of reactor transfer functions, allowing the classification of perturbation type and localisation of its source from generated simulations of the induced neutron noise (fluctuations of neutron flux).

3.1.1 *Data Pre-Processing*

The data provided by CORE SIM is a 3-D representation of the induced neutron noise, considered as the ideal case where all voxels in the core volume contain readings, simulating a detector at every location. Moreover, the output is a clean signal only carrying the information provided by Dirac's like perturbation. The simulation consists of the fast and thermal neutron responses of the same scenario, each in the form of a 3-D complex signal arrays of size *32x32x26*, with each pair of volumes containing a perturbation of different types originating at differing coordinate points *i, j, k.* The dataset is comprised of 19552 instances per frequency (0.1, 1.0 and 10Hz).

To effectively learn meaningful representations within the volume a conversion procedure was devised to unroll the volumes into a two-dimensional form, allowing for a more conventional input to neural networks and one that can utilize pre-trained networks. Furthermore, due to the nature of the complex signals and the limitations of deep neural networks to process these signals, each volume was first decomposed into its amplitude and phase. Next, each of these volumes were independently 'unrolled' using the proposed conversion procedure. The unrolling involved the concatenation of axial slices into the same plane, resulting in a 2-D array containing each z-axial slice of the core volume, where the amplitude of each point refers to the amplitude of the induced neutron noise, the output of this procedure can be seen in Figure 1.

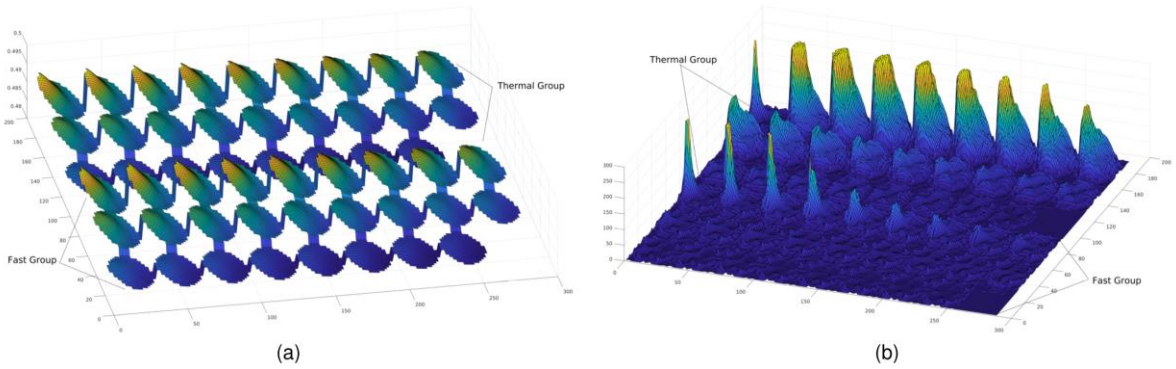

Figure 1. 2-D array output of the unrolling procedure. (a) Signal Phase (b) Signal Amplitude

To add a further level of realism and complication to the signals, each had been corrupted in two manners, the first through the addition of noise and the second by the removal of data. The intuition behind the latter is to mimic that in reality only a small number of detectors were present within the core volume, therefore removing information is key to simulate real-world conditions.

For the former method of corruption, white Gaussian noise (WGN) had been added to each signal at two distinct signal-to-noise-ratios (SNR), SNR=1 and SNR=3. To ensure that the perturbation had been influenced, the noise had been added depth-wise to each of the 26 slices of the core volume. Obscuring the signals through removal had also been achieved in a similar depth-wise manner, this involved setting a percentage of random data points to zero, removing it as input. Three different percentages of data were kept for training, 25%, 50%, and 75%, noting this had been applied before the training procedure ensuring that the same data points were consistently removed through training.

Finally, each of the inputs underwent a volumetric splitting procedure to obtain labels of a well-defined region rather than the original finer voxel representation. This was achieved by compartmentalizing the *32x32x26* volume into either twelve or forty-eight subsections by a factor of *2x2x3* or *4x4x3* respectively. These newly produced subsections were then utilized as the labels for the *i, j, k* localisation of the perturbation source. This can be illustrated in Figure 2.

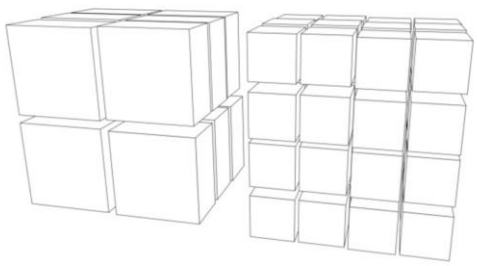

Figure 2. Visualisation of the volumetric splitting. Left: Twelve Voxels. Right: Forty-eight Voxels

3.1.2 *2-D Convolutional Neural Network*

Convolutional neural networks are a specialised type of DNN designed to perform automatic feature extraction on known grid-like topologies. This is achieved through a series of volume-wise convolutions and multiplications, followed by pooling to reduce the computational complexity of the increased number of volumetric channels produced by CNNs. Typically, CNNs receive as input image-like structures of *width x height x depth,* where depth is usually 3 for each of the RGB channels of an image. To mimic this the unfolding procedure was applied as aforementioned, this adds the further advantage of utilising large-scale image datasets to pretrain a network to learn high-level features and transfer this knowledge to the CORTEX datasets.

To input into the CNN, the fast and thermal components of amplitude and phase were concatenated together to preserve the integrity of the data. The three-channel input consequently consisted of the concatenated amplitude as the first and second channel (both identical) with the third being the concatenated phase. Finally, the image was zero-padded to the target dimensions of *299x299x3* to correctly allow transferable features from the CNN model.

The CNN architecture was chosen to be that of the Inception-v3 [7], due to its high capability to transferable parameters ratio, including its known effectiveness to perform well on a variety of image-based tasks. Furthermore, a transfer learning approach was implemented where the Inception-v3 architecture had been pre-trained on a large-scale image dataset, ImageNet. This is of specific interest as the features learnt from pretraining can be transferred to the CORTEX data, potentially allowing for faster training with greater performance due to the transfer of learnt general features (weights). The fine-tuning procedure consisted of feeding the three-channel signal through the Inception-v3 model until the last pooling layer, where a 2048-dimensional vector representation was extracted. This representation was fed to a fully connected network outputting to a SoftMax classification layer of twelve or forty-eight nodes representing each of the volumetric splitting resolutions previously described.

The best performance was achieved with a two layer fully-connected network fed from the pre-trained Inception-v3 model, the results of which can be seen in Table 1. These hidden layers consisted of 2048 units each with Rectified Linear Units (ReLU) activations:

$$ReLU: \rightarrow f(x) = \max(0, x) \tag{1}$$

Furthermore, the regularizing technique Dropout had been implemented to mitigate the effect of overfitting in the network. Dropout involves the randomly setting of neurons in the hidden layer of the fully-connected network to 0.0, this reduces the learning capacity of the network. In the case of the proposed network, the probability of keeping neurons in each hidden layer ($n: \neq 0$) was set to $P(n) = 0.8$. Finally, to deal with the imbalance of classes when splitting the volume into subsections, it was chosen to employ the use of a weighted categorical cross entropy loss function:

$$\mathcal{L}(x, \hat{x}) = -\sum_{j=1}^{J} \omega_j x \log(\hat{x}) \tag{2}$$

where:

$$\omega_j = \frac{\max(\{N_i\}_{i=[1:J]})}{N_j} \tag{3}$$

This loss encourages the model to focus on under-represented classes improving performance. The term ω_j is a weight coefficient for the j^{th} of all classes J as a function of the proportion of instances N_j compared to the most densely populated class.

Table 1 Results of the CNN Inception-v3 Unfolding

Classes	Sensors (%)	Signal	Train/Val/Test (%)	Pretrained Accuracy (%)	Scratch Accuracy (%)
12	100	Clean	75-10-15	97.0	99.9
	100	SNR=3	75-10-15	88.7	99.9
	100	SNR=1	75-10-15	84.2	98.0
	25	Clean	50-20-30	93.7	99.9
	25	Clean	25-15-60	93.4	98.4
	25	SNR=1	50-20-30	76.6	94.1
48	100	Clean	75-10-15	92.3	99.9
	100	SNR=1	75-10-15	72.9	92.5
	25	Clean	50-20-30	90.3	97.8
	25	Clean	25-15-60	85.1	91.1
	25	SNR=1	50-20-30	65.2	82.3

The results of the CNN unfolding to volumetric subsections for different combinations of signal obscuring and dataset size can be seen in Table 1. The highest performance was achieved on the twelve class subsections training the CNN from scratch reaching 99.9% accuracy, the pretrained CNN performed worse with 97.0% accuracy, this pattern was also the case for the forty-eight subsections. These results may be caused by the features being extracted in pre-training, where they could be seen as irrelevant to the tasks at hand, potentially 'confusing' the network with non-task dependent information. Furthermore, it is seen that the CNN based unfolding performs well with only a 1.5% drop in performance despite reducing the training set size from 75% to 25% of the total dataset size for the twelve classes of clean signals. Furthermore, in the forty-eight class problem the addition of noise proved the network to be robust, with the clean signals and 25% of sensors achieving 97.8%, with the SNR=1 resulting in 82.3% accuracy.

3.1.3 *Localisation through Clustering*

The generation of labels from the dataset involved the volumetric splitting as described previously which decreased the granularity of the volume, providing more instances per class resulting in a more effective training set. However, it is important for fine localisation to increase granularity, but at the trade-off of less instances per class. This trade-off resulted in a methodology to artificially increase the resolution of the volume, clustering the predicted subsection to further increase the granularity of the prediction.

The methodology takes advantage of latent variable representations of the input volume already extracted from the previously proposed 2-D CNN. The latent variable is the output of the CNN architecture after the last average pooling layer (a 2048-dimensional vector), and has the advantage of using learnt feature representations, which conveniently also are far smaller in size in comparison to the much larger 299x299x3 raw input.

To elaborate, consider the case of increasing the resolution from twelve volumetric subsections to forty-eight using clustering. Each training image was fed to the CNN aforementioned to compute the 2048-dimensional representation. Each representation vector referring to a corresponding voxel location of the twelve subsections is then clustered into one of four through the use of k-means++ clustering algorithm, increasing the granularity from twelve to forty-eight sub-clusters. Finally, the centroids of these forty-eight sub-clusters were calculated. To classify during testing, all data were fed to the trained CNN and their respective representations were classified using a nearest-neighbour method to one of the forty-eight centroids.

Formally, given the extracted $N^{[L-1]}$-dimensional activations $(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n)$, from the last fully-connected layer $L-1$ (of L layers), as latent variable representations of n total input images, the objective function

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \quad (4)$$

clusters them into k sets $C = \{C_1, C_2, \ldots, C_k\}$ as to minimise within-cluster L^2 norms. In the above case the $N^{[L-1]}$-dimensional activations come from the output of the last pooling layer of the CNN.

Finally, the k-means++ [8] seeding strategy was used, rather than randomly sampling the initial centroids as in the standard k-means approach, leading to faster convergence and generally better results [8]. Furthermore, visualisations of the clustering can be seen in Figure 3 showing the final clusters from the 2048-dimensional feature representation. These visualisations were achieved using t-Stochastic Neighbour Embedding (t-SNE) as it provides accurate structure revealing maps of high-dimensional data in lower dimensions, such as in 2D or 3D.

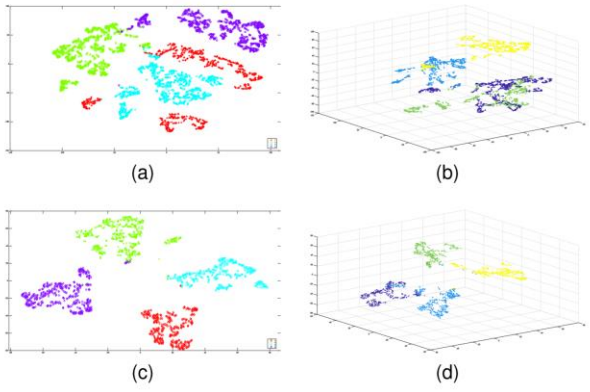

Figure 3. t-SNE visualisation of k-means (k = 4) of the seventh block; the obtained training set clusters are (a-b) and the test set predictions are shown in (c-d)

The results of clustering to a finer granularity for sub-clusters per volumetric subsection were promising achieving 95.3% test accuracy when increasing the resolution from twelve to forty-eight classes. This indicates the promise for finer prediction, with K-NN approach being implemented

further increasing granularity to the original *32x32x26* resolution. This approach achieved an impressive localisation error of 1.05 MSE or average Euclidian distance (L^2 norm) at $k = 6$. This means an average error of just over 1.0 coordinate point in the reactor when localising the source (*i, j, k*).

3.1.4 *Denoising Autoencoder*

Autoencoders are a type of neural network designed to copy its input to an output instead of to a particular label. Autoencoders learn an identity function of the input to help learn additional features within the network, denoising autoencoders (DAE) on the other hand are forced to learn a denoising function of the corrupted input to a target output. This property is especially useful for the denoising and reconstruction of signal that have been corrupted, providing clean signals to model of the types previously described.

The DAE is constructed of an encoder, to compress and encode the corrupted input $f(\hat{x})$, and a decoder for up-sampling and reconstructing the input, forcing the network to learn useful properties of the input. The DAE is made of five convolutional layers, four of which have 32 (*3x3*) filters and ReLU activations. The final convolutional layer includes 3 filters of size *3x3* and a sigmoid (σ) activation function to reconstruct an image of identical dimensions to the input. Moreover, two max pooling layers of filter size *2x2* were used to reduce the representation and finally produce a *75x75x32* encoding layer. The decoder follows the same pattern but in reverse, as to up-sample the encoding volume back to the input size. A diagram of the proposed DAE network can be seen in Figure 4.

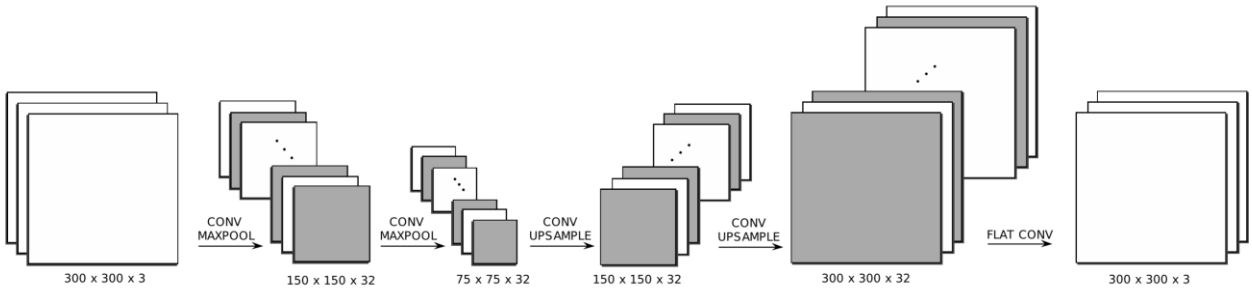

Figure 4. Depiction of DAE architecture used to solve the corrupted signal reconstruction problem

The network was trained to minimise the mean squared error penalising the network relative to how similar the reconstructed input $g(f(\hat{x}))$ is to the original input x.

$$mse = \frac{1}{n}\sum_{i=1}^{n}(x_i - g(f(\hat{x}_i))^2 \quad (5)$$

Experiments on the DAE were performed on 25%, 50% and 75% of available sensors corrupted with SNR=1 and performance measured using normalised cross correlation (NCC), providing a sub-pixel image matching evaluation precision (See 6.1.1 for details). The average NCC of the reconstruction was 0.991 in the worst case, where -1.0 NCC means the input and reconstruction are completely different, with +1.0 NCC representing a complete match. Results of all experiments of the DAE can be seen in Table 2 and visualisations in 6.1.2.

Table 2. Settings and results of the DAE experiments

Deep-CNN Autoencoder				
Sensors	Signal	Train/Test	Normalised Cross-Correlation (NCC)	
			Clean vs Corrupted	Clean vs Reconstructed
75%	Clean	25% / 75%	0.77	0.995
50%	Clean	25% / 75%	0.57	0.995
25%	Clean	25% / 75%	0.37	0.993
25%	SNR=1	25% / 75%	0.36	0.991

3.2 Towards a Deep Unified Framework for Nuclear Reactor Perturbation Analysis

The developments made outline a 3-D convolutional neural network approach for the unfolding of the reactor transfer function in the frequency domain, a recurrent neural network for classification of perturbation type in the time domain, and a unified approach aiming to combine the two domains to form a common shared representation. The work in [6] was presented at the 2018 IEEE Symposium Series in Computational Intelligence (SSCI 2018, India, November 2018).

3.2.1 *Data Pre-Processing*

3.2.1.1 Frequency Domain

The data generated in the frequency domain comes in the form of 3-D volumes, produced by Chalmers using the CORE SIM simulation software. The complex *32x32x26* signal volumes, are comprised of fast and thermal groups of the same scenario, each containing one perturbation type (absorber of variable strength, propagating perturbation originating from the lowest axial position – referred to as type 1 hereafter, or randomly sampled from all axial positions – referred to as type 2 in the following) originating at one voxel location (*i, j, k*), identically as previously described. Similarly to previously developments, the complex signals were decomposed into to their amplitude and phase components, due to complex signals not being easily implemented into DNNs.

However, for the proposed method, the 3-D volumes were to be kept in their volumetric state where the four volumes of *32x32x26* (fast, thermal, phase, and amplitude) are concatenated together to form a *64x64x26* 3-D volume for input. Furthermore, to add a level of realism to the input, the volume was first corrupted through the removal of information to corrupt the signal. This corruption was achieved through the application of a *32x32* mask applied in a depth-wise manner along the 26 slices. The mask removes information (by multiplying by zero) at two ratios, keeping 5% and 20% of the initial measurements of the volume. Finally, as a last step the *64x64x26* volume was zero padded to *64x64x32* for manipulation convenience.

3.2.1.2 Time Domain

Data generated in the time domain through the use of SIMULATE-3K by PSI were given in the form of fifteen scenarios. Each scenario is presented as 100 second readings sampled at 10Hz for forty-eight in-core detectors and eight ex-core detectors, resulting in a *56x10001* array of neutron flux readings of a given perturbation. The perturbation types and their given properties can be seen in Table 3.

Given the limited amount of data presented, it was appropriate to perform data augmentation to increase data samples. This augmentation was achieved through re-sampling via a sliding window approach, where each window had a width of 100 time-steps (one second) and stride 5 time-steps producing 1980 vectors of 100 time-steps per detector, per scenario. Furthermore, to explore the effect of noise on the proposed network, white Gaussian noise has been added at two signal-to-noise-ratios, SNR=10 and SNR=5.

Finally, the simulations are provided with the inclusion of combinations of different scenarios, therefore to identify the individual perturbations a one-hot encoding approach has been implemented. This encoding seen in Table 3 as 'ID' identifies the classes via a separate binary digit, noting amplitude variations of perturbations are considered of the same classification.

Table 3. Scenarios provided in deliverables 2018.04.12-Version v2 and 2018.06.04-Version v2 with their assigned Binary Class ID for multi-label classification

Scenario	Perturbation Type	Frequency	Amplitude	Binary Class ID
1 2	5x5 Cluster FAs (X) 5x5 Cluster FAs (X)	White Noise White Noise	1mm 0.5mm	1 0 0 0 1 0 0 0
3 4	5x5 Cluster FAs (X) 5x5 Cluster FAs (X)	1.5Hz 1.5Hz	1mm 0.5mm	0 1 0 0 0 1 0 0
5	Inlet Coolant Temp	Random	±1℃	0 0 1 0
6	Inlet Coolant Flow	Random	±1%	0 0 0 1
7	Inlet Coolant Temp & Inlet Coolant Flow	Random	±1℃ & ±1%	0 0 1 1
8 9	5x5 Cluster FAs (X) & Inlet Coolant Temp	White Noise & Random	1mm & ±1℃ 0.5mm & ±1℃	1 0 1 0
10 11	5x5 Cluster FAs (X) & Inlet Coolant Temp	1.5Hz & Random	1mm & ±1℃ 0.5mm & ±1℃	0 1 1 0
12 13	5x5 Cluster FAs (X) & Inlet Coolant Flow	White Noise & Random	1mm & ±1% 0.5mm & ±1%	1 0 0 1
14 15	5x5 Cluster FAs (X) & Inlet Coolant Flow	1.5Hz & Random	1mm & ±1% 0.5mm & ±1%	0 1 0 1

3.2.2 *3-D CNN*

Three-dimensional CNNs operate to extract spatial information across a 3-D space using the same principles as aforementioned in the 2-D case. 3-D CNNs utilise volume-wise convolutions and multiplications with 3-D kernels resulting in a set of 3-D filters learnt to capture spatial patterns, a formal description is given in 6.2.1. The choice to implement a 3-D network was the next step to achieve localisation of the perturbation source to the resolution of the simulated core. Additionally, it makes intuitive sense to utilise the volume as a whole for unfolding the reactor transfer function rather than per-slice.

The proposed network is comprised of seven 3-D CNN layers, each CNN layer is followed by a batch normalisation layer to normalise the output for faster training, and a ReLU activation to limit the values between 0 and 1. The network itself follows a conventional VGG structure in [9] with the addition of additional tricks to increase capability. One of which was the implementation of *1x1x1* convolutions, these convolutions known as Bottleneck layers are introduced between the conventional *3x3x3* layers. This reduces the number of parameters in the network incurred by the 3-D convolutions and increases the complexity, maintaining an appropriately sized model for faster training. Moreover, Max Pooling with *2x2x2* kernels down-samples the input to smaller dimensions for computational efficiency. Finally, a Global Average Pooling (GAP) layer produces a 512-dimensional vector representation as output, averaging over each feature map per channel. A depiction of this model can be seen in Figure 6.

The 512-dimensional representations produced by the above CNN were fed to two separate fully-connected layers, one for the multi-label classification of the perturbation type with three sigmoid non-linear units. The second for the regression of perturbation source coordinates containing three linear units one for each of the *i, j, k* coordinates. For the combined perturbation case, the three sigmoid units represent the seven possible classes denoted by one-hot encoded binary units as

$$C = \{001, 010, 100, 101, 011, 110, 111\} \tag{6}$$

where C contained all combinations of localised (100), travelling type 1(010), and travelling type 2(001) previously described. In the case of implementation, the three linear units for regression become nine to account for the possibility that all three perturbations may be present at once.

The 3-D convolutional network was trained by the means of minimising the error of multiple objectives or tasks (classification and localisation), known as a multi-task problem. A linear weighted sum of the individual losses of the tasks is performed, weighting each loss by a coefficient λ_i to control the dominance of each task loss over the gradient. This multi-task optimisation objective is minimised with respect to $\boldsymbol{W}$ where the loss for classification is the negative log-likelihood, and for coordinate regression is the L2 loss. P and C denote the number of perturbation types and location coordinates respectively, with λ_1, λ_2 being tuned weight coefficients for each loss.

$$\mathcal{L}(\mathcal{D}; \boldsymbol{W}, \lambda_1, \lambda_2) = -\frac{1}{N}\sum_{i=1}^{N}\left[\frac{\lambda_1}{P}\sum_{j=1}^{P}\left[y_1^j \log\left(\hat{y}_1^j\right) + \left(1 - y_1^j\right)\log\left(1 - \hat{y}_1^j\right)\right] + -\frac{\lambda_2}{C}\sum_{c=1}^{C}\|y_2^c - \hat{y}_2^c\|^2\right]_i \tag{7}$$

The results obtained in Table 4 were achieved with $\lambda_1 = 0.3, \lambda_2 = 0.7$ for classification and regression respectively. These values were obtained through a hyper parameter grid search which can be seen in 6.2.2, the network was trained as previously described only adjusting the values of λ_1, λ_2. The

training procedure was optimised using the Adam optimizer with default parameters and with a batch size of 32.

Table 4. Results of the Frequency Domain 3D CNN experiment for perturbation classification and localisation regression. (*) Marks combined perturbation scenarios

Sensors (%)	Train/Valid/Test (%)	Classification Accuracy (%)	(i, *j*, *k*) Regression	
			MAE	MSE
20	60 / 15 / 25	**99.75±0.09**	**0.2528±0.03**	**0.1347±0.02**
20	25 / 15 / 60	99.12±0.17	0.4221±0.05	0.4152±0.07
20	15 / 25 / 60	98.62±0.22	0.5886±0.05	0.8174±0.12
5	60 / 15 / 25	99.32±0.18	0.326±0.05	0.2086±0.04
5	25 / 15 / 60	98.34±0.22	0.4818±0.05	0.6044±0.08
5	15 / 25 / 60	97.27±0.54	0.689±0.1	1.0749±0.25
20*	60 / 15 / 25	99.82±0.05	0.5602±0.04	1.6036±0.15
20*	25 / 15 / 60	99.56±0.07	0.8942±0.04	3.5739±0.16
20*	15 / 25 / 60	99.44±0.08	0.9635±0.06	4.2814±0.19
5*	60 / 15 / 25	99.47±0.03	0.8809±0.04	3.4424±0.16
5*	25 / 15 / 60	98.33±0.24	0.5001±0.04	0.6381±0.08
5*	15 / 25 / 60	**97.15±0.15**	**1.9528±0.11**	**11.902±0.66**

As shown in Table 4 the network achieved high classification performance with 99.75%±0.09 and 97.15%±0.15 accuracy in the best case and worst case respectively. For regression the difference between the best case of 0.2528 ± 0.03 (MAE), 0.1347±0.02 (MSE) and the worst case of 1.95±0.11 (MAE), 11.90±0.66 (MSE) shows that the introduction of the combined perturbations impacted the performance of the network. Whereas, the classification task was seen to be robust to reduction in available sensors and introduction of combinations, only differing by ≈2.60%. However, in the best case scenario the coordinate prediction was only ≈0.25 coordinate positions from the target on average, showing excellent regression at the full core resolution. Overall, the results show that deep learning approaches are again able to accurately unfold the reactor transfer function to a high-resolution volume whilst still achieving good results in the presence of multiple perturbations per input.

3.2.3 *LSTM-RNN*

Recurrent neural networks (RNNs) are different specialised type of DNN, designed to learn temporal relationships within data. Given the sequential nature of the time-series data provided by PSI, the use of RNNs are particularly useful for learning feature of the pre-processed 100 time-step signals. In particular, long short-term memory (LSTM) [10] – a type of RNN unit – were chosen to be implemented in this proposed work due to its high-capability to learn long range dependencies within data, ideal for the lengthy 100 time-step signals. Specifics on implementation and technical details of LSTM-RNNs can be found in the original paper [10] or [6].

The LSTM network were constructed of two stacked layers with each LSTM cell containing 512 neurons, this can be visually depicted in Figure 6. The stacking of LSTM layers allows for an increased representational capacity, allowing for a greater potential of learning meaningful representations from the signals. The network outputs from the LSTM in a similar fashion to the previously described 3-D CNN where a 512-dimensional representational vector of the input signal is fed to a four non-linear sigmoid layer for classification. Each of the four sigmoid units represents one of the possible classification classes seen in Table 3 in the ID section for binary classification.

The time series data generated and processed as in section 3.2.1.2 Time Domain, contains scenarios which are comprised of multiple perturbations per signal, these scenarios as with the 3-D instance above are to be classified for all perturbation types per input. This task of multiple classifications per input is known as multi-label classification. The proposed approach implements a binary cross entropy loss / negative log-likelihood criterion to predict a multiple class per given input. The following criterion was minimised to achieve such a result

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{PN}\sum_{j=1}^{P}\sum_{i=1}^{N}\left[y_j \log\left(\hat{y}_j\right) + \left(1 - y_j\right)\log\left(1 - \hat{y}_j\right)\right]_i \quad (8)$$

where P is the number of sigmoid units used as output for the multi-label classification task, and N is the number of samples in a batch.

Using the above LSTM-RNN architecture, individual detector measurements of 100 time-steps in length, or one second (*1x100*) were utilised to detect the presence of each of the fifteen scenarios. Moreover, the split of train, validation and test sets were constructed by systematically splitting specific detectors. The intuition behind this choice was to ensure that the same scenario from the same detector was not present across sets, therefore not invalidating the results. The layout of which detectors were chosen to reside in each set is shown in Figure 5 below. Hyperparameters of the network were tuned experimentally, optimised with RMSprop, and batch size of 32.

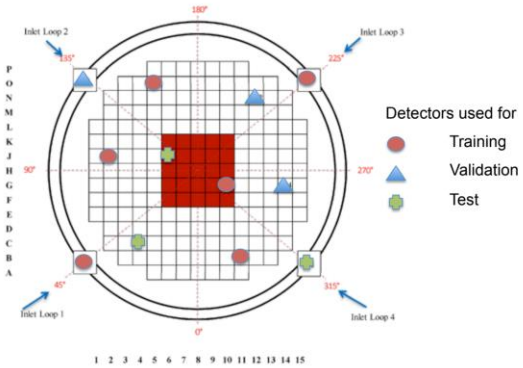

Figure 5. Description of detector locations for the signals used in training, validation and test of deep LSTM network model in classifying different types and combinations of time domain perturbations

The results of this time series classification were excellent, achieving 97% classification accuracy for clean signals with no corruption for just one detector reading per input. Continuing, the addition of noise at SNR=10 and SNR=5 resulted in good results, with 81% and 77% classification accuracy

respectively. These results further show the effectiveness of machine learning and deep learning approaches for the effective unfolding of reactor functions, even in the time domain per detector.

3.2.4 *Combined Approach*

The combination of differing data domains is of great interest to learn as much as possible from a limited amount of data, moreover, inputting in different domains each has their specific benefits contributing to a better unfolding procedure. The work proposed outlines a unified approach to merge learning representations from each domain in a manner illustrated in Figure 6. It is essential to first understand and clarify that the data generated from the two different simulation software; CORE SIM and SIMULATE-3K, both model different reactor cores in the work reported in this document. This results in an incompatibility between the data samples, meaning they cannot be fused early in the learning process. Nevertheless, the developed methodology will be capable of handling simulation data from different domains but corresponding to the same reactor core, when such simulation data are available.

In order to cope with simulation data from different domains, it was chosen to concatenate the outputs of each of the previously described models to form a single representation vector containing the representations from both domains. This approach was made simpler for the fact that both networks output their feature vectors as 512-dimensional representations, thus making the process simpler. Additionally, the classification layer of the network fused both the RNN and CNN classification layers together forming a seven-unit non-linear sigmoid classification layer to identify the occurrence of a given class of both domains. The time-domain data also did not have the ability to be localised, meaning that a procedure was implemented to pass the 512-dimensional representation to a nine-unit linear output layer to regress coordinate when a perturbation type was identified from the frequency domain. This approach can be seen by the XOR gate in Figure 6.

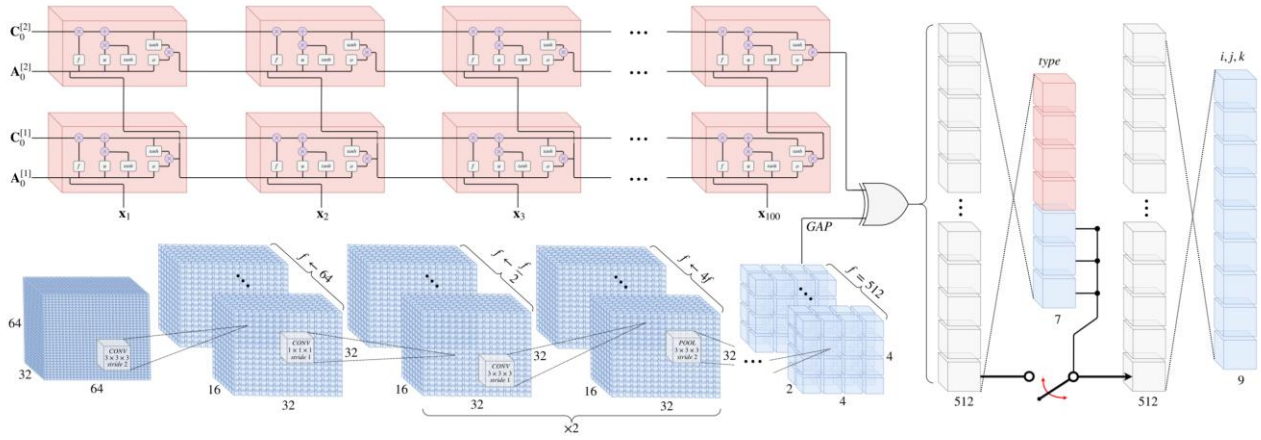

Figure 6. Unified framework for time and frequency domain perturbation type classification and location regression, using a LSTM network at the top for time domain signals and a 3D CNN below for frequency domain signals; both networks output 512 dimensional latent variable representations of their inputs, with their flow being controlled by XOR gate logic and a switch (which is activated for perturbation coordinate regression in the frequency domain)

The resulting implementation utilises shared latent variables extracted from each of the given architectures for the individual domains. Although not entirely unifying, this approach provides the grounding for the cases when core simulations or plant data in different domains contains readings from the same core physics.

3.3 3D Convolutional and Recurrent Neural Networks for Reactor Perturbation Unfolding and Anomaly Detection

This work proposes a DNN for the classification and localisation of core perturbations in the frequency domain, in addition to an extended dataset in the time-domain**.** The proposed method also utilises the large-scale dataset with an increased core resolution and number of perturbation types, resulting in a larger challenge to unfold the reactor functions. The work presented in this section was presented at the FISA 2019 conference as part of the FISA/EURADWASTE PhD poster competition (FISA 2019, 9th European Commission Conference on EURATOM Research and Training in Safety of Nuclear Reactors, Romania, June 2019) [12].

3.3.1 *Large-Scale Data Handling*

We have received a new set of outputs in the frequency domain produced in CORE SIM+ by Chalmers, resulting in a largely increased number of perturbation types (nine types, including modes of vibration, compared to former three types) and source locations. Additionally, the data generated is of a finer resolution compared to the previously deliverable, presented in *32x32x34* volumes of complex signals. As with the previous developments the complex signals have been decomposed into their amplitude and phase components with each of these volumes being concatenated on a new fourth dimension to produce a *2x32x32x34* input for the DNN.

The initial challenge of this dataset is the overall size and computation resources to process the large amounts of generated data (upwards of 3TB). To ensure that developments could be made in a timely manner and that storage of processed data was not of imminent issue, ‘on-line’ scripts were developed based on those delivered by Chalmers. These scripts utilise the computational components and power available for machine learning, more specifically, the multiple-processor and graphics processing unit implementation for faster processing. These developed features allow for the generated signals to be pre-processed ‘on-the-fly’ rather than being processed and then stored.

Pre-processing of the generated data follows similar principles as was described above. Firstly, the signals were decomposed into their amplitude and phase components; however, only the thermal neutron group has been utilised, so as to reduce computational load. The exclusion of the fast group was chosen, as it was seen to provide no advantage to the training of the DNN compared to the thermal group alone. In addition, neutron detectors are mostly sensitive to the thermal neutron flux only. Furthermore, to provide a level of realism to the experimentations, only forty-eight in-core detectors and eight ex-core detectors have been used, represented by their voxel locations within the core volume. These locations have been systematically chosen to represent the positions of where the detectors reside in within the core simulated in CORE SIM+. Lastly, the APSD and CPSD of these fifty-six detector readings were calculated using the modified scripts; this aligns the readings to be closer to those of real plant readings. The now processed APSD and CPSD of the fifty-six detectors per amplitude and phase is then fed to the network, as an *2x32x32x34* volume of data.

3.3.2 *Densely Connected 3-D CNN*

Dense networks [11] are a powerful convolutional architecture, utilising a number of machine learning methods to more effectively learn and transport input representations through the network. The main advantage being the increased flow of information / gradients through the use of jump

connections between layers, implemented by concatenating feature maps from one layer to all preceding layers. More formally,

$$X_\ell = H_\ell([X_1, X_2, \dots, X_{\ell-1}]) \tag{9}$$

where a ℓ^{th} hidden layer H_ℓ, receives as input the feature-maps of all preceding hidden layers, resulting in X_ℓ. In the context of this contribution, this allows for high and low-level spatial features to be more appropriately shared between layers contributing more to the learning of different features.

The proposed network adapts the idea of dense connections to be utilised with 3-D convolutions, creating a network construction illustrated by Figure 7 below. Each convolutional layer was followed by Batch normalisation and ReLU activation with the last layer implemented being Global Average Pooling (GAP), outputting a 256-dimensional representation of the input, two separate fully-connected layers are used as output of the network. The first is a nine-unit non-linear sigmoid layer for the classification of the nine perturbation types, the latter is a three-unit linear layer for the regression of the (*i, j, k*) coordinates of the perturbation source.

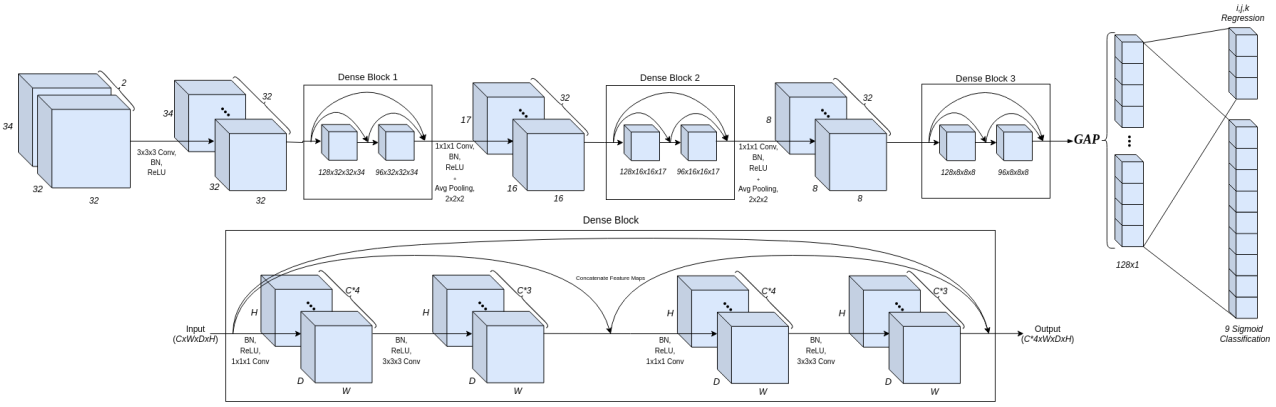

Figure 7. Adapted densely connected 3-D CNN for classification and regression of perturbation types and source from a simulated core volume; the dense block is shown below

This proposed network was trained to minimise the same multi-task loss function described in 3.2.2, with the Adam optimiser using default parameters. Furthermore, the data had been split into 60%, 15% and 25%, source location wise per frequency for the train, validation, and test sets respectively. Finally, a batch size of 32 was used with the same weight coefficients for the multi-task loss function of $\lambda_1 = 0.3, \lambda_2 = 0.7$. The experimentation results show that the increased number of perturbation types can be identified effectively with 99.9% and 99.85% accuracy for the full volume and the forty-eight in-core detectors. Additionally, the performance of the dense connections can be seen in the regression tasks, providing a low error, given the increased granularity of the core model. For just forty-eight detector readings, the mean absolute error (MAE) and mean squared error (MSE) for the localisation of the perturbation source were 0.2954 and 0.3171 respectively. This equates to roughly ≈4cm error in a (*4m x 4m x 4m)* volume.

3.3.3 *Time Domain Extension*

Following the success of the time-series approach developed in 3.2.3, an extension was derived for the application on the new provided dataset. This dataset referred to twelve additional scenarios, compared to the previous cases, creating a total of twenty-seven scenarios. These new scenarios

contain a new Y-direction fuel assembly vibration and further combinations of scenarios. This can be seen, including the associated values, in Table 5.

Table 5. All Scenarios with their assigned Binary Class ID for multi-label classification

Scenario	Perturbation Type	Frequency	Amplitude	Binary Class ID
1 2	5x5 Cluster FAs (X) & Inlet Coolant Temp & Inlet Coolant Flow	White Noise & Random & Random	1mm & ±1℃ & ±1% 0.5mm & ±1℃ & ±1%	1 0 1 1 0 0
3 4	5x5 Cluster FAs (X) & Inlet Coolant Temp & Inlet Coolant Flow	1.5Hz & Random & Random	1mm & ±1℃ & ±1% 0.5mm & ±1℃ & ±1%	1 0 1 1 0 0
5 6	5x5 Cluster FAs (Y) 5x5 Cluster FAs (Y)	White Noise White Noise	1mm 0.5mm	0 0 0 0 1 0 0 0 0 0 1 0
7 8	5x5 Cluster FAs (Y) 5x5 Cluster FAs (Y)	1.5Hz 1.5Hz	1mm 0.5mm	0 0 0 0 0 1 0 0 0 0 0 1
9 10	5x5 Cluster FAs (X) & 5x5 Cluster FAs (Y)	White Noise & White Noise	1mm & 1mm 0.5mm & 0.5mm	1 0 0 0 0 1
11 12	5x5 Cluster FAs (X) & 5x5 Cluster FAs (Y)	1.5Hz & 1.5Hz	1mm & 1mm 0.5mm & 0.5mm	0 1 0 0 0 1

The data provided is of the same form as previously described. They were processed in the same manner, augmented via sliding window and separating detectors from one-another. The train, validation and test split has also been constructed via the scheme shown in Figure 5 and corruption of the signals by white Gaussian noise took place at SNR=10 and SNR=5. Finally, the same one-hot encoding technique has been applied with two additional units added for the increased number of scenarios. This in-turn means that the LSTM-RNN network outputs to six sigmoid units for multi-label classification of perturbation types.

The network utilised had been trained as the previously described LSTM-RNN with adjustments made for the additional output units. The results remained high with 96.41% accuracy on the clean signals, 82.25% for SNR=10, 80.34% for SNR=5. These results although based on extending our former developments, help solidify the use of deep learning methodologies for perturbation classification. Moreover, it is impressive to obtain such a high accuracy, for so many overlapping classifications, from a one second reading at one given detector location only.

3.4 Convolutional Networks based on Scaleograms for Reactor Perturbation Unfolding and Anomaly Detection

The main objective of the proposed methodology is to be able to identify the driving perturbation from neutron flux signals captured by the in-core and ex-core sensors of a nuclear reactor. This is achieved through the application of the discrete wavelet transform (DWT) to the signals and the

generation of the corresponding wavelet scaleograms [14]. Subsequently, those scaleograms are provided to a CNN that learns to classify them to the different perturbation types.

The experiments performed had two distinct directions: to investigate if scaleograms could be used for anomaly detection in nuclear reactors and to identify the optimal time window that should be used when performing this sort of analysis. Intuitively, smaller windows would segment the input signal into more parts and consequently produce a larger training set for the CNN. However, segmenting the signal too much could lead to the perturbation not being captured in the time windows and reduce performance. The proposed methodology was evaluated on data generated by the SIMULATE-3K tool to model some basic types of perturbations occurring in nuclear reactors.

The scenarios were grouped under four main categories:

1. Fuel assembly vibrations: This category describes a vibration of a single fuel assembly in one direction. Four different fuel assemblies were simulated for different conditions (i.e. different type of vibration and amplitude). A total of 32 scenarios were generated through this process.
2. Cluster of fuel assemblies vibrating together: Here a cluster of fuel assemblies located in the center of the reactor, vibrate synchronously. 12 scenarios with different combinations of vibration frequencies and amplitudes fall into this category.
3. Coolant flow oscillations: This scenario simulates random oscillations in the flow of the coolant by up to $\pm 1\%$.
4. Coolant temperature oscillations: This scenario simulates random oscillations in the temperature of the coolant by up to $\pm 1^{o}C$.

Each scenario consisted of a series of signals from the reactor: 48 from sensors inside the reactor (denoted as “in-core”) and 8 from outside (denoted as “ex-core”). Depending on the scenario, the signals have a length of either 35 sec or 100 sec and a sampling rate of 0.01s. The in-core sensors are in 8 different axial locations, each taking measurements at 6 different heights [6]. The ex-core sensors, on the other hand, are placed in 4 locations at 2 different levels. It has been reported that due to their spatial symmetry (the 4 ex-core sensors are placed 90^{o} from one another) some vibrations may not be detected at all [6]. Due to this, measurements from inside the core should also be considered.

A few preprocessing steps were performed on each signal. The first was to re-sample the signals using bilinear interpolation, so that they all have the same length, since in some cases the sampling rate differed from scenario to scenario. A length of 8192 was selected, as it is the closest power of 2 to the original length, which helps with the DWT computation. After bringing all signals to the same length they were detrended, removing the linear component that best fits the data. Afterwards, the mother wavelet that best describes the data was identified. This was done by computing the optimal mother wavelet for each scenario and selecting their mode.

In order to evaluate the generalization capability of our methodology, a cross-validation scheme was devised where the input signal was partitioned into k parts. One of these was used for testing and the rest $k-1$ were used for training the CNN. This procedure was repeated k times with a different part used for testing. The selection of which signal should be used for testing was consistent for all signals in a given run; for example if $k=4$, for a given run all models were trained of parts 1, 2, 4 and evaluated on part 3, so that the third part of the input signals wasn't seen by any model.

To investigate the minimum signal length that the CNN can be trained on, three values for k were selected: $k = \{2, 4, 8\}$. The latter produced input signals consisting of $8192 / 8 = 1024$ points (which roughly corresponds to 1.25 seconds). A preprocessed signal that has been split into 8 parts is depicted in Figure 8.

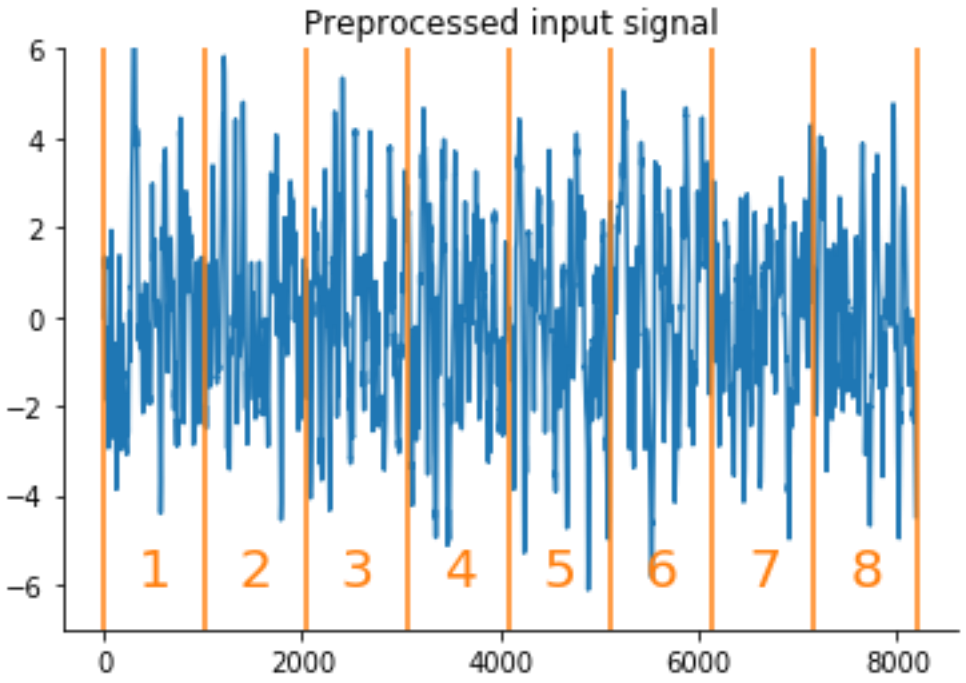

Figure 8. A preprocessed (i.e. detrended and resampled) signal, partitioned into k=8 parts

After partitioning the signal into k parts, a scaleogram was extracted from each one. Due to the imbalance of samples between the classes, both the training and evaluation of the model were weighted. The weighted accuracies on the validation sets of the k folds were averaged.

A CNN architecture with 6 layers and 2.5 million parameters was selected (as shown in Figure 9).

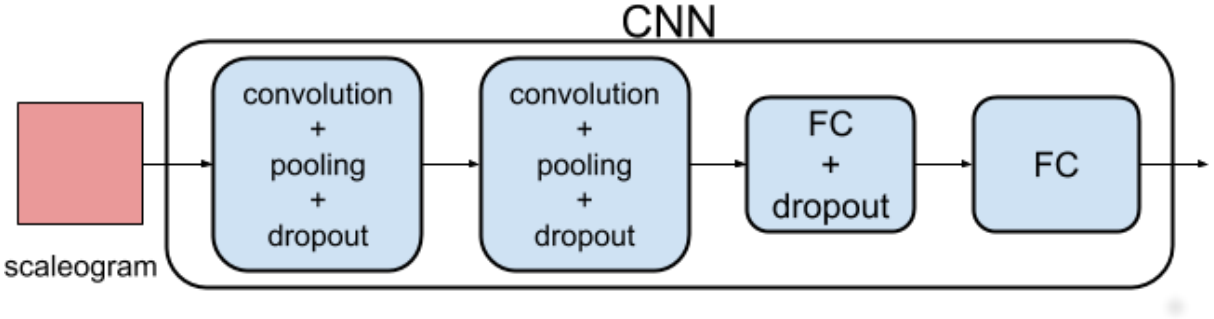

Figure 9. The CNN architecture employed for classifying the scaleograms

The first convolution layer has 32 filters, a kernel of $11x11$ and strides of $4x4$, while the second one 64 filters and a kernel size of $5x5$. After both convolutions, max pooling operations are applied with kernels of $3x3$ and strides of $2x2$. Finally, dropout is applied after these pooling layers, with a probability of 35%. The architecture is concluded with two fully connected layers, with 256 and 4

neurons respectively; dropout (with 30% probability) is applied after the first. The network was trained on a cross-entropy loss with an Adam optimizer with an initial learning rate of 10^{-4}. As mentioned previously, the samples were weighted during training to alleviate the class imbalance.

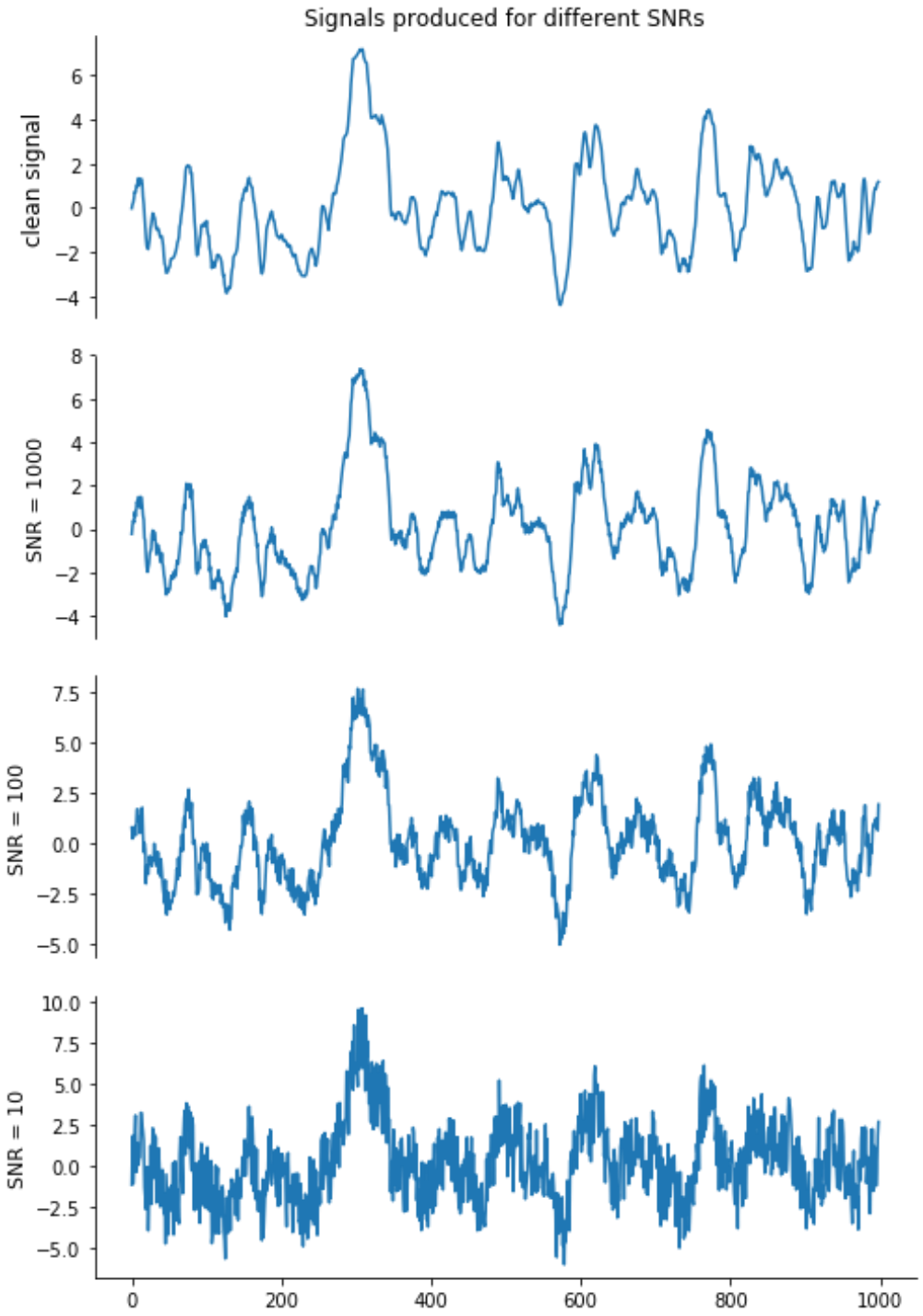

Figure 10. Input signal for different signal-to-noise ratios

After assessing the ability of the model to discriminate between the four classes, further experiments were performed to examine robustness of the technique to external noise. Each input signal was imputed with white noise of different SNRs; the signals were processed with the presented methods (i.e. detrended, resampled, split into k parts, decomposed by DWT, converted to scaleograms).

A total of 10 cases were examined, with different SNR, in particular, $10^6, 10^5, 10^4, 10^3, 10^2\ 10, 1, 10^{-1},\ 10^{-2}$ and 10^{-3}; the higher the value of the SNR, the "cleaner" the signal. A sample input signal with noise added to it is depicted in Figure 10. Values higher than 〖10〗^3 have little visual effect to the signal, while for values lower than 10 the presence of noise in the signal is significant.

The first experiment involved using "clean" signals from the reactor. For each type of sensor (i.e. in-core/ex-core), 3 values of k were examined. The mean accuracy of k folds is presented in Table 6.

Table 6. The mean fold accuracy for each type of sensor per number of splits

k	*In-core*	*Ex-core*
2	98.04%	83.33%
4	99.63%	83.44%
8	99.85%	93.88%

As expected, partitioning the signal into more parts leads to better overall accuracy, because this procedure produced more input signals and in turn more scaleograms. Furthermore, even though in-core sensors clearly outperform the ex-core ones, the latter remain reliable and can detect the anomalies in most of the cases.

Table 7. Weighted accuracy for both in-core and ex-core sensors for various signal-to-noise ratios

	k					
	2		4		8	
SNR	*In-core*	*Ex-core*	*In-core*	*Ex-core*	*In-core*	*Ex-core*
clean	92.80%	70.52%	99.10%	80.47%	99.83%	82.40%
10^6	92.48%	71.15%	98.99%	81.46%	99.80%	83.26%
10^5	92.19%	68.02%	99.13%	80.73%	99.84%	82.94%
10^4	91.46%	73.33%	98.95%	81.15%	99.84%	82.97%
10^3	92.88%	66.56%	98.90%	80.78%	99.81%	82.94%
10^2	93.18%	67.71%	98.74%	80.99%	99.70%	85.05%
10	90.68%	73.02%	98.06%	80.78%	99.41%	82.84%
1	85.40%	68.44%	89.75%	77.24%	95.96%	78.52%

10^{-1}	72.36%	23.12%	73.13%	53.85%	70.39%	57.86%
10^{-2}	13.59%	4.06%	18.94%	4.27%	16.24%	4.51%
10^{-3}	4.08%	3.33%	3.64%	4.06%	3.64%	3.98%

The input signals were imputed with white noise of various strengths, indicated by the signal-to-noise ratio (SNR). The results of the noise experiments can be seen in Table 7.

The framework's overall performance remains unchanged for noisy signals with high values of SNR (i.e. low noise). For signals with an SNR of below 10 the performance starts deteriorating. From this point on, the noise has a strong presence in the overall signal, making it tough for the network to classify. At very low values of SNR (i.e., below 10^{-1}), the signal is dominated by the noise, making it virtually impossible to classify without denoising. The trend of the in-core sensors outperforming the exterior ones continues throughout this experiment. Figure 11 illustrates this deterioration.

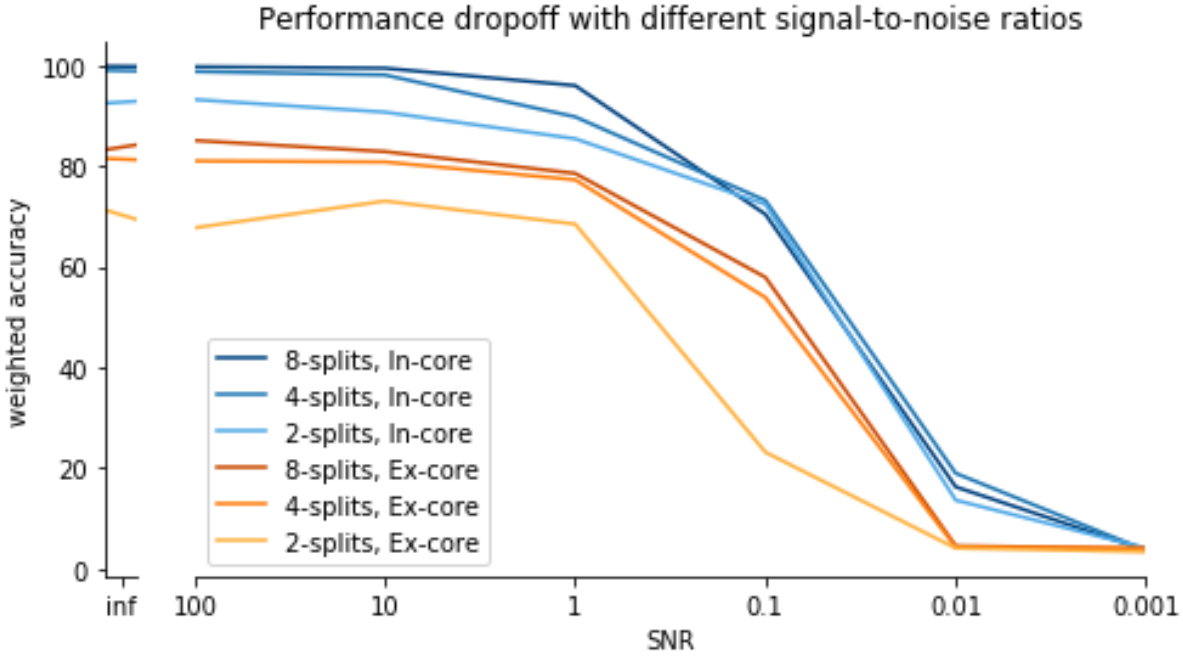

Figure 11. Performance dropoff for different signal-to-noise ratios

3.5 Machine learning techniques for predicting detector malfunctioning

In this Section, a method is presented for identifying anomalous detector signals [13]. Simulated neutron detectors (NDs) model the time-dependent neutron flux (NF) fluctuations at various radial and axial locations in the core. For the purpose of efficiency and timeliness of response, rather than utilising all the available ND signals in concert for verifying neutron noise (NN) signal correctness, the use of 3-tuples of NDs is proposed. The rationale behind combining the information of sets of appropriately selected 3-tuples of NDs for reaching consensus is that, while two NDs are not adequate for the task-at-hand (in case of disagreement between a pair of ND signals, it is not possible to establish which ND is misbehaving and/or receiving abnormal information), three NDs

can be used for competently, as well as confidently deriving signal validity. This number of NDs not only simplifies the decision-making process and minimises computational complexity, but also boosts robustness of the final decision in cases of failing NDs and/or erroneous/unexpected NN signals, thus also accommodating for the far-from-infrequent situation of scarce in- and ex-core instrumentation.

In the following, two signals derived from the fuel assembly vibration scenario simulation performed by PSI (10-7 vibrating fuel assembly, white noise, 1.1 mm amplitude), named *indet1* and *indet3* (collected by neutron detector I1 at Levels 1 and 3, respectively) are used for validating signal *indet2* (collected by neutron detector I1 at Level 2). These signals are shown in Figure 12, with Figure 13 further demonstrating the pairwise relationships between them, derived by independently normalizing each signal in the $[0.1, 0.9]$ interval and subtracting each signal from the other. In all Figures in this Section, i.e., Figures 12-16, the x-axis represents time (e.g., 3501 samples of each detector, sampled every 0.01 sec for a total duration of 35 seconds), while the y-axis depicts the measured neutron noise. The high-frequency fluctuations, which are largely due to the inherent noise in the NN signals, trends, as well as transients can be observed at different timestamps and scales. Despite their differences (shown in Figure 13), the signals employed for this investigation are highly correlated (with their pair-wise correlation coefficients ranging in $[0.85, 0.99]$).

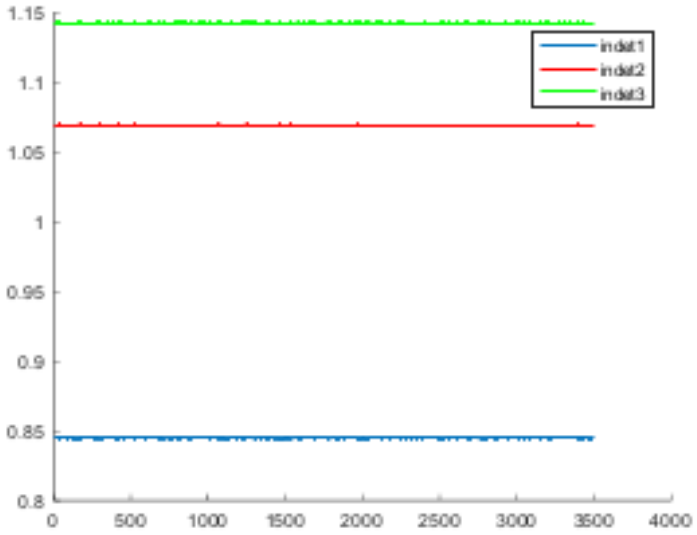

Figure 12. The original three *indet1-2-3* signals

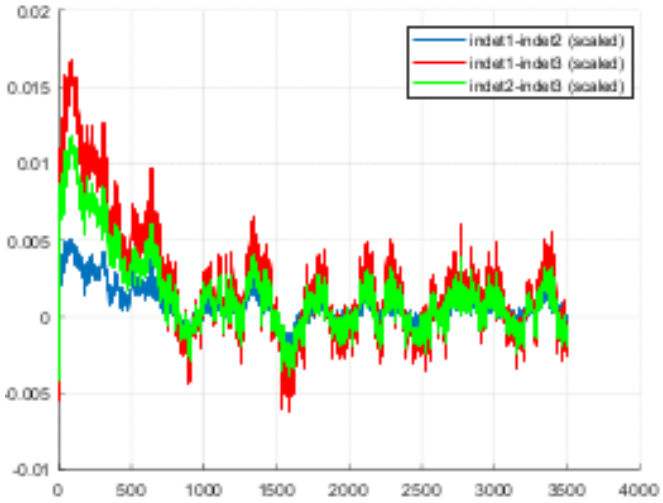

Figure 13. The pairwise differences of the three signals, *indet1-2-3* (after scaling/normalization)

Further to the selection of triplets of signals, the absolute minimal signal length is used for signal verification: the $\{indet1(t), indet2(t)\}$, $\{indet3(t), indet2(t)\}$ and $\{[indet1(t), indet2(t)], indet2(t)\}$ input-output pairs of signals captured at times t are employed – in turn – for setting up the targeted prediction. During testing, signals $indet1(t'), indet3(t')$ captured at time $t'(\neq t)$, are used for predicting $indet2(t')$. In case of malfunction, the change (drop) in correlation between one or more pairs of signals can act as an early sign of decreasing agreement between them and be further exploited for identifying the erroneous signals and/or malfunctioning NDs. It is also possible to aggregate the decisions of different combinations of NDs (based, for instance, on signal correlation, ND location and distance) for reaching a consensus-driven decision that takes into account each ND derived decision while further exploiting the confidence in, as well as the complementarity of, the individual decisions.

The following (perturbed, deviating from normal) signals have been derived from $indet1, indet2$ and $indet3$ via injection/addition/juxtaposition of drifts, implemented by adding linear trends to the mean value of the original signal; different amplitudes (e.g. small/large trends in Figure 14) are used for investigating the capability, as well as the "limit", of successfully retrieving the original signal.

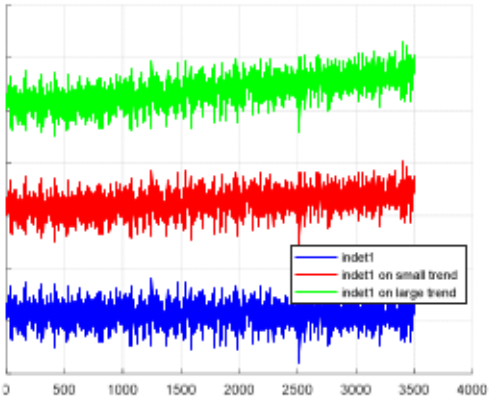

Figure 14. Examples of drift

Fluctuations simulated by adding sinusoids of different amplitudes and periods to the original signal (e.g. small and large oscillations in Figure 15) are also considered.

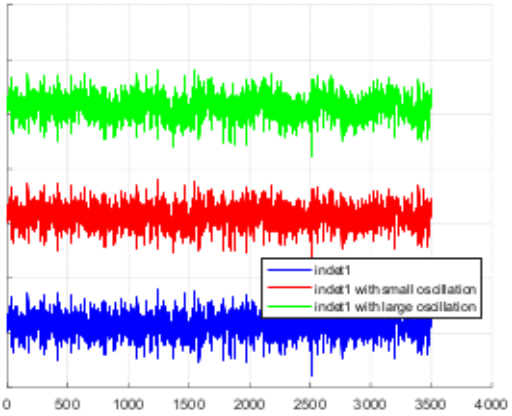

Figure 15. Examples of fluctuations

Combinations of drift and oscillations (e.g. middle signal in Figure 16) are also considered. Intermittencies, where parts of the original signal are missing and are substituted by the (local) mean value of the signal overlaid with white noise of varying standard deviation (top signal in Figure 16) are also investigated.

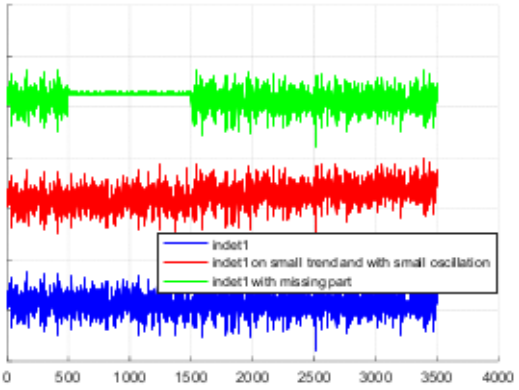

Figure 16. Examples of intermittencies

Three prediction methods, covering the entire parametric through to non-parametric spectrum, are used next, namely parametric Polynomial Approximation (PA), Semi-Parametric Splines (SPSs) and non-parametric General Regression Neural Networks (GRNNs).

While PA analytically determines the optimal polynomial coefficients that allow input variables optimally approximate output variables (by minimising the distance between actual and predicted outputs), SPSs employ a set of predefined forms which are, subsequently, combined in a piece-wise manner so as to optimally approximate the output variables. The nonparametric GRNN, on the other hand, through single-epoch training generates optimal separating hyperplanes between the pattern classes.

The GRNN constitutes a two-layer artificial neural network architecture of straightforward as well as transparent construction, which makes use of a single tunable parameter (the spread, σ). The value

of σ determines the range of influence of each training pattern, consequently shaping the GRNN approximating hyperspace in terms of the available (training) data and the desired continuity of the separating hyperplanes. A small value of σ creates a localized influence of each training pattern on the separating hypersurface. Conversely, a large value of σ makes separating hypersurface more general and impervious to outliers and other extreme training patterns.

The nodes of the two GRNN layers represent the input features, with each feature being encoded in a single node of the lower layer of the GRNN; as far as the training patterns are concerned, each pattern is encoded in a single node of the upper layer of the GRNN. The connections between nodes are only possible between nodes of different layers and are, furthermore, limited to pairs of nodes (one node from each layer) which are related in a positive or negative manner; expressing whether the appearance of the input feature represented by the node of the lower layer is promoted or suppressed by the training pattern encoded in the node of the upper layer. The connection weights are determined independently for each node of the upper layer; the magnitude of all non-zero connections from each node of the upper layer is the same, with the sign of each connection being determined by whether the pair of connected nodes expresses a positive, or negative relationship.

GRNN construction is swift, flexible and can be adjusted in an on-line manner. A single presentation of the training-set patterns is enough for setting the optimal form of a non-parametric class-separating hyper-surface such that accurate GRNN responses are returned not only to known inputs, but also to novel inputs derived from the same pattern space. By changing σ, the generalization potential and robustness to noise of the GRNN is adjusted.

The three prediction methods are implemented independently – yet under identical conditions of data normalization and partitioning, using 10-fold cross-validation – for monitoring and concurrently identifying ND deviations from steady-state operation and malfunctions. The methods have been subsequently compared in terms of accuracy, in two distinct cases: distorted signals; malfunctioning NDs.

As reported in [13], the SPS operation has been found slightly – yet not consistently – superior to that of PA, demonstrating the advantages of using more degrees of freedom, as these are allowed by the specific piece-wise separating hyper-surface construction methodology over PA. The GRNN approach has been found superior to both PA and SPS approaches, demonstrating swift training and perfect recall of the entire dataset; robustness to noise and reliability to missing or otherwise corrupted data (when substituted by their average or weighted average values) during testing; significantly improved prediction accuracy when the GRNN predictions of the different input encodings are combined.

The GRNN demonstrates the smallest - by at least one order of magnitude - thresholds for detecting erroneous signals (TS, TL, OS, OL, OT, MR), where: TS/TL stands for Trend Small/Large; OS/OL stands for Oscillation Small/Large; OT stands for Oscillation & Trend; MP stands for Missing Part. Furthermore, the GRNN is comparably more robust to normal operation, with higher thresholds by far over PA and SP).

4 Conclusions

Contributions made in WP3 Task 3 have been advantageous to test and demonstrate the capability of machine learning methods for the unfolding of reactor transfer functions from limited readings of the induced neutron noise. The large number of developments provides a grounding for the application to real plant data in the future, helping to gain a greater understanding of the challenges and potential approaches to overcome these challenges. The results from all contributions have shown that high performance is achieved, iteratively improving with the introduction of more, higher resolution, and more challenging data sets. It has been shown that high performance is achieved in the frequency, as well as in the time domain, with high robustness to corruption by noise.

4.1 Future Work

As work continues and contributions evolve, localisation of even larger numbers of perturbations at a given time will be possible. In addition, uncertainty values can be provided for the predictions, giving a high level of trust to the deep learning systems. Adaptation of these models to provide validation on real plant data will be the main focus of our future work, aiming to transfer the knowledge and developments performed on simulation scenarios to the real plant readings.

5 References

[1] Christophe Demazière, Paolo Vinai, Matthew Hursin, Stefanos Kollias and Joachim Herb. Noise-based core monitoring and diagnostics – overview of the Cortex project. Advances in Reactor Physics, Mumbai, December 2017.

[2] Christophe Demazière, Paolo Vinai, Matthew Hursin, Stefanos Kollias and Joachim Herb. Overview of the Cortex project. PHYSOR, Mérida, April 2018.

[3] Christophe Demaziere. Core sim: a multi-purpose neutronic tool for research and education. Annals of Nuclear Energy, 38(12): 2698–2718, 2011.

[4] Gerardo Grandi, Jeffrey A. Borkowsky, and Kord S. Smith. Simulate-3k models and methodology. SSP-98013, Revision, 6, 2006.

[5] Francesco Caliva, Fabio De Sousa Ribeiro, Antonios Mylonakis, Christophe Demaziere, Paolo Vinai, Georgios Leontidis, Stefanos Kollias, et al. A deep learning approach to anomaly detection in nuclear reactors. IEEE International Joint Conference on Neural Networks (IJCNN), Brazil, July 2018.

[6] Fabio De Sousa Ribeiro, Francesco Caliva, Dionysios Chionis, Abdelhamid Dokhane, Antonios Mylonakis, Christophe Demaziere, Georgios Leontidis and Stefanos Kollias. Towards a Deep Unified Framework for Nuclear Reactor Perturbation Analysis. IEEE Symposium Series on Computational Intelligence (SSCI), India, November 2018.

[7] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. IEEE Conference on Computer Vision and Pattern Recognition, Nevada, USA, June 2016.

[8] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. 18th annual ACM/SIAM Symposium on Discrete Algorithms, pp. 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. IEEE International Conference on Computer Vision and Pattern Recognition, Boston, June 2015.

[10] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.

[11] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. IEEE Conference on Computer Vision and Pattern Recognition, Hawaii, USA, July 2017.

[12] Aiden Durrant, Georgios Leontidis and Stefanos Kollias. 3D Convolutional and Recurrent Neural Networks for Reactor Perturbation Unfolding and Anomaly Detection. 9th European Commission Conference on EURATOM Research and Training in Safety of Reactor Systems and Radioactive Waste Management (FISA 2019 - Euradwaste' 19), Romania, 2019.

[13] Tatiana Tambouratzis, Dionysis Chionis and Abdelhamid Dokhane. General Regression Neural Networks for the Concurrent, Timely and Reliable Identification of Detector Malfunctions and/or Nuclear Reactor Deviations from Steady-State Operation. IEEE Symposium Series on Computational Intelligence (SSCI), India, November 2018.

[14] Ervin Sejdic, Igor Djurovic and Ljubica Stankovic. Quantitative Performance Analysis of Scalogram as Instantaneous Frequency Estimator. IEEE Transactions on Signal Processing, 56(8):3837-3845, 2008.

6 Annexes

6.1 A Deep Learning Approach to Anomaly Detection in Nuclear Reactors. Supporting Material

6.1.1 *Normalised Cross-Correlation (NCC)*

NCC is an evaluation metric employed to measure the difference between two matrices. In the case of these developments, the precision between the corrupted image reconstruction and the original image.

Given two three-channelled images A and B, we can quantify their similarity per channel as

$$ncc = \frac{\sum_{i,j}(a_{i,j}-\mu_{\boldsymbol{A}})(b_{i,j}-\mu_{\boldsymbol{B}})}{\left[\sum_{i,j}(a_{i,j}-\mu_{\boldsymbol{A}})^2 \sum_{i,j}(b_{i,j}-\mu_{\boldsymbol{B}})^2\right]^{0.5}} \quad (9)$$

where $a_{i,j}$ and $b_{i,j}$ refer to each pixel in A and B with $\mu_{\boldsymbol{A}}$ and $\mu_{\boldsymbol{B}}$ as their mean pixel intensities per channel. The final ncc is the average of the three channels, given as $\mathbb{R} \cap [-1,1]$.

6.1.2 *DAE Visualisations*

The figure below shows the reconstruction of the corrupted signals from the convolutional denoising autoencoder proposed in section 3.1.4. The original and corrupted signal are shown below, with the denoising autoencoder given the corrupted signal as input and aiming to map the function that takes such inputs, reconstructing to form an output as close to the original as possible. The output of the denoising is shown in (c).

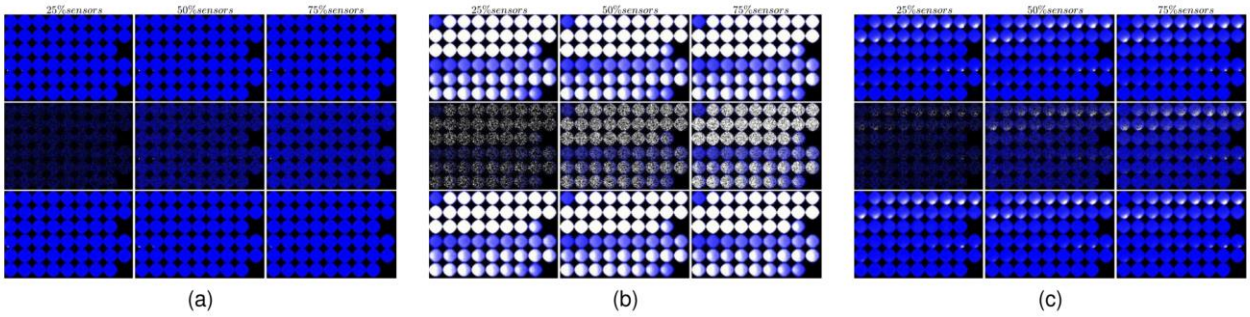

Figure 17. Examples (a-c) of reconstruction provided by DAE when: 75%, 50%, 25% of sensors were used; in each case: left: Original signal, middle: Obscured signal, right: Reconstructed signal

6.2 Towards a Deep Unified Framework for Nuclear Reactor Perturbation Analysis: Supporting Material

6.2.1 *3-D Convolutions*

Formally, in 3D CNNs one would compute a pre-activated value of a given unit $n_{i,j,k}^{[\ell]}$ at (i,j,k) position in a 3-D feature map of layer ℓ, by the summation of the weighted kernel contributions from the previous hidden layer's units $\boldsymbol{A}^{[\ell-1]}$

$$n_{i,j,k}^{[\ell]} = \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} \sum_{z=0}^{Z-1} \boldsymbol{W}_{x,y,z}^{[\ell]} \boldsymbol{A}_{i+x,j+y,k+z}^{[\ell-1]} \quad (10)$$

where $\boldsymbol{W}_{x,y,z}^{[\ell]}$ is a learnt weight of kernel $\boldsymbol{W}^{[\ell]}$ of dimensions $X \times Y \times Z$ in layer $\boldsymbol{\ell}$, which is convolved with the previous layer ($\boldsymbol{W}^{[\ell]} * \boldsymbol{A}^{[\ell-1]}$) .

6.2.2 *Grid Search for Loss Weight Coefficients*

A grid search of hyper-parameters was conducted to find the most appropriate values for the weight coefficients for both classification and regression for the weighted multi-task loss function. The figure below shows the error for the classification error, and regression errors (MAE and MSE) for all values 0.1 – 0.9 at 0.1 increments, this results in values for 0.3 and 0.7 for the classification and regression loss weights respectively. These values are then substituted into the multi-task loss function used in section 3.2.2 to train the 3-D CNN network proposed.

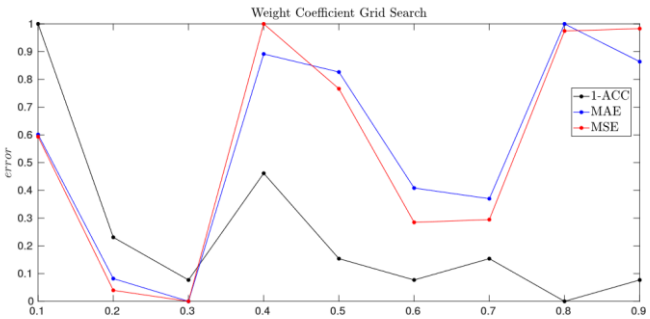

Figure 18. Weight coefficient grid search for the 3D-CNN classification and regression losses; coefficient 0.3 for classification and 0.7 for regression yielded the best performance