HKIE THE HONG KONG INSTITUTION OF ENGINEERS 香港工程師學會

HRSG early tube leak detection with a transfer learning neural network and Gramian Angular Difference Field

Hon Fung Chow

CLP Power Hong Kong Limited, Hong Kong, People's Republic of China

ABSTRACT

This paper proposes a novel heat recovery steam generator (HRSG) early tube leak detection model which leverages a convolution neural network classifier by utilising transfer learning with ResNet50 architecture. The design goal of this model was to achieve high classification accuracy with a minimal amount of leakage data. The model is also intended to be user-friendly and require minimal hyperparameter tuning. The proposed neural network was trained on the drum-specific conductivity time series data of HRSGs encoded in the Gramian Angular Difference Field (GADF). The model yielded a validation accuracy of 96.64%, true-positive rate of 93.28% and precision of 100% in regard to the validation set. The study included experiments on the influence of different encoding algorithms, Markov Transition Field (MTF) and Recurrence Plot (RP), and architectures on the performance of the model. This paper further discusses the viability of adapting the design to other time series classification problems.

KEYWORDS HRSG conditional monitoring; machine learning; deep learning; transfer learning; time series classification
CONTACT Hon Fung Chow honfung.chow@clp.com.hk
Received 2 August 2021

1. Introduction

The HRSG is a necessary component for combined cycle gas turbines (CCGTs) to generate steam for steam turbines. HRSGs are mechanically very similar to steam boilers in design. Dual pressure HRSGs like the ones studied in this paper are composed of low-pressure (LP) and high-pressure (HP) circuits. A steam drum may be present in the circuits, the main purpose of which is to separate wet steam from sub-critical boiler water. Boiler tubes in an HRSG absorb heat from gas turbine exhaust gas for transfer to boiler water. Leakage in a boiler tube will therefore adversely impact the efficiency of a CCGT.

Early detection for HRSG poses both operational and maintenance challenges. From an operation standpoint, it is difficult to detect tube leaks by sensor measurements alone. As part of the emission control regime, moisture sensors are installed at the chimney of the HRSGs in Black Point Power Station (BPPS) to provide data regarding the emission content. Operators of HRSGs may suspect a HRSG leakage if there is an increase in moisture in emission content, since the leaked water would enter the exhaust gas path and escape via the chimney. However, this method is rarely reliable if the moisture is low, and also the moisture content in the chimney is heavily influenced by the humidity of the atmosphere. A second method involves regular testing for the rate of water consumption in CCGTs. Operators regularly capture the water consumption data of CCGTs to monitor for any trends in abnormal consumption and perform corresponding corrective measures. The drawback of this method is that the trace amount of water from early boiler tube leakage will not severely impact the overall water consumption of the unit. A slight increase in water consumption due to tube leakage may be attributed to variances in the captured data and ignored. The last detection method is to observe the pattern of chemical dosing in HRSGs. The conductivity of boiler water in BPPS is regulated by trisodium phosphate dosing. Under normal operation, the specific conductivity of boiler water does not undergo drastic fluctuation. A leak in a boiler will invariably cause a loss of phosphate dosing and subsequently cause a fluctuation in specific conductivity. Observant operators may notice an increased frequency in boiler dosing and suspect a leak in the boiler.

All the operational detection measures only provide circumstantial evidence that indicates a possible leak in a boiler, and an internal inspection inside the boiler is necessary to confirm an actual leak. In order to safely perform an internal inspection, operators must request a maintenance outage from the grid operators in order to shut down the unit and allow the interior to cool sufficiently such that personnel may enter the HRSG interior safely for leak detection. For the entire duration of the unit being shut down for inspection, the equivalent availability factor (EAF) will decrease and indirectly increase the operational cost of the unit (IEEE Standard Association, 2007). From a maintenance standpoint, early leak detection can potentially shorten the maintenance time. Allowing a boiler leak to develop also threatens the integrity of other surrounding equipment. The leaked water serves as a coolant to the surrounding metallic objects, e.g., support structures, boiler tubes in the vicinity and the HRSG casing. The cooling of these objects creates a temperature gradient on the surfaces and induces an undesirable strain on the material which can potentially damage or shorten its lifespan (Hu et al., 2019).

This poses a challenge for the operators of HRSGs since one must decide whether to risk performing an early but possibly futile internal inspection at the cost of decreasing the unit EAF or allow the leak to develop until the circumstantial evidence is convincing enough to justify the shutting down of the unit. This paper proposes an alternative detection tool to detect the onset of a tube leak which develops a characteristic time series pattern in the specific conductivity of drum water in order to aid this decision.

The development of a deep-learning classifier requires expertise in the devising of a neural network and domain knowledge to select appropriate data for training. The proposed model aims to maximise the user-friendliness of the model and minimise its engineering cost by eliminating the need to build a training neural network from scratch. Conventionally, a convolution neural network (CNN) or recurrent neural network (RNN) is used to classify a time series. Since boiler tube leaks are rare events in most plants, it is often impractical to gather enough tube leak records to train a deep CNN or RNN effectively from scratch. This motivated the use of encoding algorithms to convert times series data into images and train the classifier using a pretrained neural network such as ResNet50, in order to maximise the classification performance based on the minimal amount of failure data.

Similar previous pursuits to leverage machine learning for boiler leak detection have been impractical because of the use of data from simulated leak conditions in laboratory setups and the need for hyperparameter tuning or knowledge to design neural network architecture. This motivated a study to determine a machine learning solution to the boiler leak problem that can be easily deployed, by adopting transfer learning and pretrained networks, which are consistently accurate, through the use of data from real boiler leak occurrences.

2. Literature review

The research on the detection of early-stage boiler tube leakage can be categorised into sensor-driven and data-driven approaches. The prevalent methods proposed in the first category use acoustic sensors. Acoustic sensors are a traditional, non-contact solution that can be deployed at any location of a boiler. When a boiler tube ruptures, the quickly depressurised boiler water turns into a high-speed steam jet in the flue gas area that emits a loud noise that can be subsequently located by acoustic sensors, thereby locating the source of the leak. This solution has two main disadvantages. The first is that in areas of lower pressure, such as economisers and evaporators, or in boilers with a lower pressure rating, early-stage tube leaks do not create a loud enough noise to be detected by acoustic sensors. The other disadvantage is that the background noise in the operation area can confuse the acoustic sensors (Anujaa et al., 2020). After compensating for background noise by decreasing sensors' sensitivity, their capability to detect early-stage tube leaks is also reduced.

The second category of research attempts to use various algorithms or spectral analysis (Wan et al., n.d.) to classify the time series data from sensors, e.g. acoustic sensors (Ramezani et al., 2020) and tube metal temperature sensors (Ismailet al., 2016). The methodology commonly involves a combination of statistical procedures to clean raw sensor data such as principal component analysis (PCA) and various variants of neural networks such as fully connected neural networks (FCNN) (Sohaib and Kim, 2019) and bidirectional LSTM models (Ramezani et al., 2020). There are several areas of improvements in these attempts which motivated this study. Firstly, most of the research is done on data generated from laboratory setups instead of real data (Shahul Hamid et al., 2013). This makes the trained model susceptible to signal noises that are common in practical environments. The proposed models can only demonstrate the capability of classifying time series data generated in controlled settings but not necessarily in an industrial area. Secondly, the research favours directly classifying time series data with FCNN and RNN. The design of FCNN and RNN requires specialised knowledge of the hyperparameters in optimisation algorithms, effect of different layers on the training outcome and appropriate selection of loss functions. The goal of the model proposed in this paper is to minimise the amount of fine-tuning such that the model is readily deployable in practical situations. Lastly, the previous research all trained their proposed neural networks on large datasets from scratch. Large-scale, balanced datasets with adequate failure examples are difficult to acquire since tube leaks are rare in most boilers. A neural network trained on an unbalanced dataset would make incorrect inferences from the data (Wei and Dunbrack, 2013) and mislead boiler operators.

3. Dataset

The proposed convolution neural network is trained and validated using time series data of the LP drum specific conductivity of the HRSGs in BPPS encoded using the Gramian Angular Difference Field. The LP drum specific conductivity readings are captured by the online chemistry monitoring system which actively and continuously samples the boiler water and analyses it in comparison to standard solutions. The dataset contains 1,280 days (1 July 2017 to 1 January 2021) of the drum specific conductivity readings of eight CCGTs. The CCGT units the data of which was used in this study, belong to the same model. The time series data was labelled as 'normal' and 'leak', corresponding to the state of the HRSGs at the time when the data was captured. Both 'normal' and 'leak' class time series data was purposefully unaltered in order to retain the subtle impact on the time series patterns caused by the normal deterioration and routine operations (such as blowdown) of the HRSGs. This was to ensure that the model trained by the data can preserve its effectiveness

Table 1. Image dataset under different encoding algorithms and settings.

Algorithm	Variant	Sample Window	Time interval (min)	Number of 'Normal' Images	Number of 'Leak' Images	Leak-to-normal Ratio
GAF	Difference	Daily	5	6994	333	4.76%
			10	7004	336	4.80%
			30	7146	341	4.77%
			60	7146	341	4.77%
		Weekly	5	951	47	4.94%
			10	951	47	4.94%
			30	951	47	4.94%
			60	951	47	4.94%
	Sum	Daily	5	7146	341	4.77%
			10	7146	341	4.77%
			30	7004	336	4.80%
			60	6994	333	4.76%
		Weekly	5	951	47	4.94%
			10	951	47	4.94%
			30	951	47	4.94%
			60	951	47	4.94%
MTF		Daily	5	6444	341	5.29%
			10	6444	341	5.29%
			30	6337	336	5.30%
			60	6314	333	5.27%
		Weekly	5	878	47	5.35%
			10	878	47	5.35%
			30	884	47	5.32%
			60	878	47	5.35%
RP	Cross	Daily	5	6994	341	4.88%
			10	6994	341	4.88%
			30	7004	336	4.80%
			60	7146	333	4.66%
		Weekly	5	951	47	4.94%
			10	951	47	4.94%
			30	951	47	4.94%
			60	951	47	4.94%
	Joint	Daily	5	6994	341	4.88%
			10	6994	341	4.88%
			30	7004	336	4.80%
			60	7146	333	4.66%
		Weekly	5	951	47	4.94%
			10	951	47	4.94%
			30	951	47	4.94%
			60	951	47	4.94%

when deployed to classify real-time series patterns and prevent the model from misclassifying normal operation as tube leaks.

The number of images from the 'leak' class varied from 4.66% to 5.35% of all data, which is highly imbalanced. Neural networks trained by an imbalanced dataset yield misleadingly high accuracy by classifying all the images into the dominant class. For example, without proper treatment of the dataset, the neural network would be incentivised to classify all images as the 'normal' class. This is due to the neural network being trapped in a local minimum and is unable to escape as there are too few 'leak' images to learn from. The practical usefulness of the model would therefore be very low despite the high training and validation accuracy.

There are numerous techniques to circumvent imbalanced datasets, such as oversampling, undersampling and data augmentation (Kotsiantis et al., 2005). This paper adopts an undersampling strategy called weighted random sampling. The weighted random sampler first shuffles all images and assigns a weight given by the following formula:

$$Weight\ assigned\ to\ class = \frac{1}{ratio\ of\ number\ of\ samples\ of\ the\ class}\ . \quad (1)$$

During the training, the sampler randomly picks training samples according to the weights of the class. Therefore, the dominant class is less likely to be fed into the network and the dataset can thus be balanced.

4. Encoding Algorithms

4.1. Gramian Angular Field (GAF)

This study adopts the algorithm proposed by Wang and Oates (2015) to encode the time series with polar coordinates represented in a Gramian matrix. The algorithm of GAF first rescales the entire time series and converts all time series values into the interval [-1, 1], as necessitated by the next procedure, which plots the time series as a polar coordinate. The permutation of trigonometric sums and differences of each polar coordinate, which corresponds to one time series value, provides the final Gramian matrix. If the calculation uses trigonometric sums, it is a Gramian Angular Summation Field (GASF); otherwise it is a Gramian Angular Difference Field (GADF).

$$GASF = [cos(\phi_i + \phi_j)]\,, \tag{2}$$

$$GADF = [cos(\phi_i - \phi_j)]\,, \tag{3}$$

And

$$\begin{cases} \phi = arcos(x_i), -1 \le x_i \le 1 \\ r = \frac{t_i}{N} \end{cases}, \tag{4}$$

where t_i is the timestamp and N is a constant factor to adjust the span of the polar coordinate system.

Each element is then mapped to a RBG scale with the interval [-1, 1] on which -1 represents the blue end of the scale and 1 represents the red end of the scale. The final square image is constructed from the pixels corresponding to the elements in the Gramian matrix.

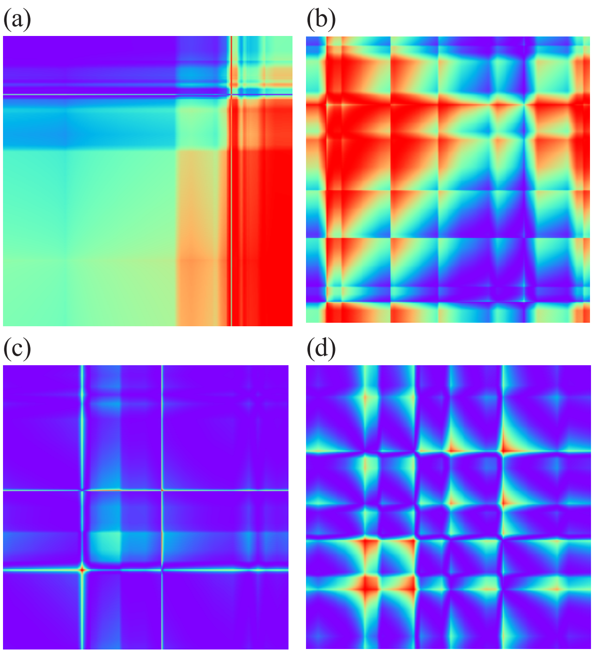


Figure 1. Encoded images from the GADF during (a) normal HRSG operation; and (b) leaking HRSG. Encoded images from the GASF during (c) normal HRSG operation; and (d) leaking HRSG.

In contrast to the traditional Cartesian approach of visualising the time series, which is helpful for human interpreters to gain insight into the time series data, the polar plot is less comprehensible by humans but is able to preserve the absolute temporal relation. This feature is essential in order to obtain distinctive and descriptive images to train the convolution neural network. Other boiler operation, such as continuous blowdown, may mimic the time series pattern of a boiler tube leak in a Cartesian representation, but not necessarily in a polar representation. The images generated by this algorithm, therefore, reduce the number of false positives in its detection and improve its robustness. Convolution neural networks require the images to be of the same dimensions for classification to be properly performed. Since the spatial relationship in the Gramian matrix relates to the temporal relationship of the time series data, the square shape of the images eliminates the need to use preparation techniques such as scaling and padding, which reduce the difficulty of using the model.

4.2. Markov Transition Field (MTF)

The encoding algorithm of transforming time series X into the MTF follows the treatment described by Wang and Oates (2015) by first determining the size of the image Q x Q, and selecting the quantile bins q to assign to each x in the time series X. The Markov transition matrix is constructed by counting the transitions between each q by the first-order Markov chain. The matrix is subsequently normalised such that wij = 1.

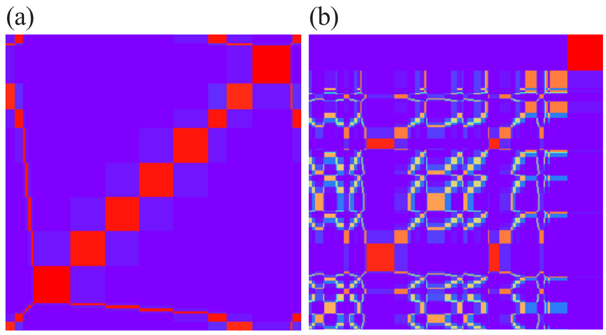


Figure 2. Encoded images from the MTF during (a) normal HRSG operation; and (b) leaking HRSG.

The MTF describes the times series as a multi-span transition probability from one bin to another. The colour is assigned to the elements in the MTF by its magnitude in transition probability and the colours of the pixels reveal the temporal relationship between time series data. The selection of Q directly influences the temporal resolution of the subsequent images. As the temporal resolution increases, the image size increases. As such, it would also increase the VRAM requirement if a GPU were deployed to train the neural network. Therefore, the MTF encoding algorithm provides users an additional control over the ensuing neural network by trading off training speed for additional temporal information in the images.

4.3. Recurrence Plot (RP)

The last encoding algorithm studied in this paper is the recurrence plot introduced by Eckmann et al. (1987) which visualises the periodicity of a time series through phase space. An RP is a two-dimensional plot of time against time on the same scale, where dots represent the recurrence of state at time *i* and a different time *j*. The recurrence plot can be described by the following equations:

$$CR_{ij} = \Theta\left(\varepsilon_i - \|\vec{x_i} - \vec{y_j}\|\right), \tag{5}$$

$$JR_{ij} = \Theta\left(\varepsilon_x - \|\vec{x_i} - \vec{x_j}\|\right) \cdot \Theta\left(\varepsilon_y - \|\vec{y_i} - \vec{y_j}\|\right), \tag{6}$$

where $\Theta(\mathrm{x})$ is the Heaviside step function:

$$\Theta(\mathrm{x}) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}. \tag{7}$$

The RP of a time series can therefore be interpreted as the trajectory of the data re-visiting the same state at different time signatures. There are two variants of RP: cross recurrence plot, which accounts for all the times that a state of one system occurs simultaneously in another system, and joint recurrence plot, which accounts for all the times that a recurrence in one system occurs simultaneously in another system.

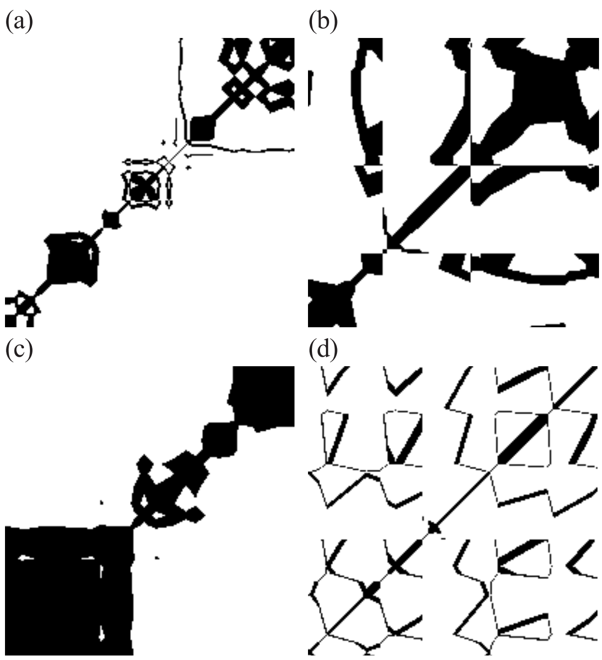


Figure 3. Encoded images from Joint RP during (a) normal HRSG operation; and (b) leaking HRSG. Encoded images from Cross RP during (c) normal HRSG operation; and (d) leaking HRSG.

RP inherently preserves the temporal relationship of the time series data and periodic patterns are visualised as checkerboard-like structures in the plot. Other time series behaviour such as drifts and rising trends drastically alter the appearance of the plots. The main advantage of recurrence plots is the distinctiveness of the plots under different time series patterns which assist the neural network to learn from the training example, since recurrence plots of a time series representing an early leaking boiler would be striking. However, the major downside of recurrence plots is that the generated figures are monochrome, which means that there is less information to be learnt by the neural network which leads to a higher risk of overfitting.

5. Methodology

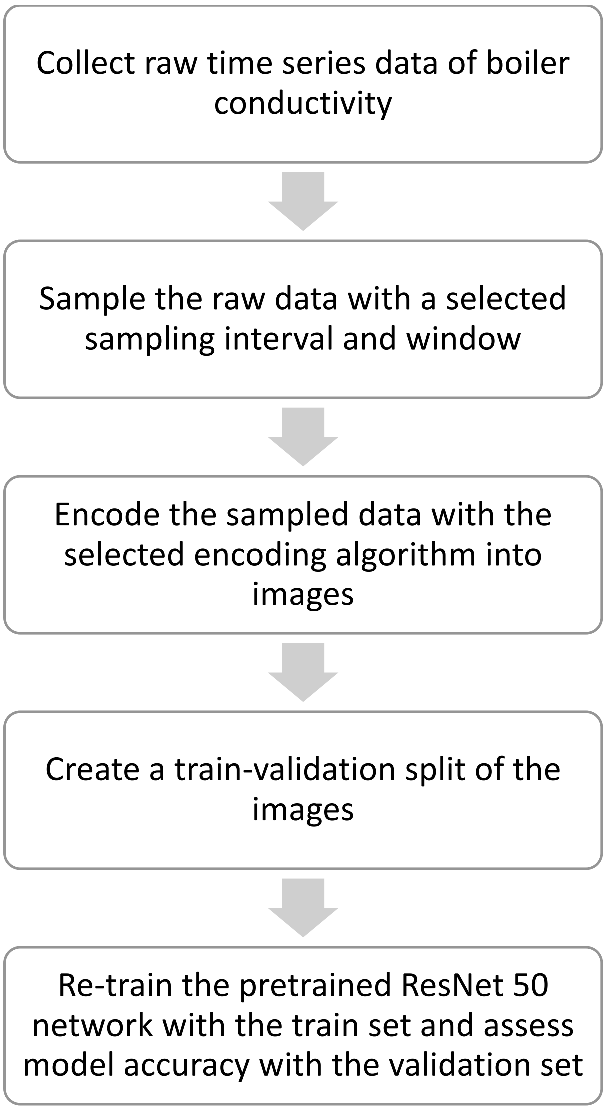


Figure 4. Graphical summary of the model training pipeline.

Table 2. The neural network architectures of the ResNet family.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

The 1,280 days of time series data of drum water specific conductivity were encoded into images under different sampling windows (daily and weekly) and different time intervals (5, 10, 30 and 60 minutes). For each variant of the encoding algorithm, a set of images were generated and trained with the same neural network such that the influence of the aspects of encoding time series data on the training outcome could be examined. In total, 40 image datasets were created containing 157,647 unique images in total.

To balance the dataset before training, the 'normal' class images were randomly selected from the dataset and removed to attain a 1:1 'leak' to 'normal' class ratio. This was done to prevent the neural network from assigning all images to the dominant class. For datasets with few 'leak' class images, such as the ones generated under a weekly sampling window, the class ratio was raised to 1:3 to bulk the dataset while implementing random weighted sampling during the training to weaken the effect of class imbalance in the dataset.

This experiment was divided into two sections. The first one was an iterative experiment on the time series data encoding algorithms. Tests models were trained with ResNet50 architecture with 10 epochs. To isolate the effect of the encoding algorithm, the hyperparameters of the optimiser, the number of epochs and the ResNet50 architecture were unchanged. The second part was an examination of the effect of the number of layers and effectiveness of transfer learning. The results from transfer learning models were compared with a simpler convolution neural network trained from scratch while the time series encoding algorithms were unchanged.

Transfer learning is a technique popularised by deep learning practitioners to improve neural network performance at a target domain by transferring the knowledge acquired from a different but related domain (Zhuang et al., 2019). For computer vision problems, transfer learning of neural networks is achieved by pre-training the neural networks on a large dataset such as ImageNet, the dataset used to pretrain the networks discussed in this study. The dataset originated from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) which contains 1,461,406 images from 1,000 object classes (Russakovsky et al., 2015). After the pre-training, the network weights were frozen and the architecture was unchanged except for the last layer, which was replaced by a fully connected layer with two output neurons to accommodate for the binary classification task in this study.

The model utilised the architecture of ResNet50 as proposed by He et al. (2015). The ResNet is a family of deep convolutional neural networks that were designed to overcome the degradation problem encountered by numerous attempts at creating very deep convolution neural networks. Researchers found that image classification accuracy increases with network depth, i.e. more convolution layers. However, the early attempts at training deep convolution neural networks encountered the problem of vanishing or exploding gradients that prevented the networks from converging. The ResNet family implemented an innovative architecture called 'shortcut connections' to allow for identity mapping to circumvent the degradation problem.

The ResNet family of neural networks feature a largely similar design with different depths. Popular choices are 18, 34, 50, 101 and 152. Raising the number of layers will increase the classification accuracy at a cost of higher FLOPS, which will increase the training time. It was found during the experimentation that the increase in network depth did not justify the longer training time. ResNet50 was chosen as a compromise between computation cost and accuracy.

Optimisation algorithms are an essential component in the training of a neural network to locate the global minima of network loss. Kingma et al. (2014) proposed Adam, which stands for adaptive momentum estimator, and it became a popular optimisation algorithm among deep learning practitioners due to its consistent empirical performance (Ruder, 2016).

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \,, \tag{8}$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \,, \tag{9}$$

$$\hat{m}_t = m_t / (1 - \beta_1^t) \,, \tag{10}$$

$$\hat{v}_t = v_t / (1 - \beta_2^t) \,, \tag{11}$$

$$\theta_t = \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) \,. \tag{12}$$

Adam incorporates the ideas of RMSprop and AdaGrad by updating the gradients using a running average of first and second gradients and annealing the gradients with bias correction to avoid exploding gradients. The main merit of Adam is its versatility in relation to different deep learning problems and that it requires little tuning of hyperparameters. The models studied in this paper used the hyperparameters recommended by Kingma in the paper, 0.9 for β_1, 0.999 for β_2 and 10^{-8} for ϵ.

Table 6. Summary statistics of the validation accuracies and true positive rates of different sampling windows.

	Validation accuracy		True positive rate	
Sampling window	Daily	Weekly	Daily	Weekly
Max accuracy	96.64%	97.22%	95.31%	100.00%
Mean	84.91%	86.93%	85.65%	92.59%
Mode	83.33%	88.89%	77.78%	88.89%
Standard deviation	6.28%	6.18%	7.05%	8.05%

6. Discussion

6.1. Effects of encoding algorithms on model accuracy

The performance of the models can be evaluated by their validation accuracy score, precision, the true positive rate and true negative rate. The model trained with images generated by the GADF under a daily sampling window and five-minute sampling frequency yields the best overall performance, scoring a 96.64% validation accuracy, 100% precision, 93.26% true positive rate and 100% true negative rate.

Among the encoding algorithms, models trained by images generated by the GADF and MTF have a markedly higher maximum validation accuracy, and the GADF yields a higher mean validation accuracy and lower standard deviation. Models trained by images generated by RP yield poorer maximum validation accuracies which could be attributed to the monochrome images generated by RP, which have a lower information entropy given the same pixel counts than other algorithms which have RGB images. In terms of true-positive rates, models trained by GADF images yield better results in terms of maximum true-positive rates, standard deviation and mean.

The models trained under a weekly sampling window have marginally better consistency in regard to validation accuracy, yielding a 0.0618 standard deviation compared to 0.0628 standard deviation from models trained under a daily sampling window. The weekly sampling window also yielded a slightly higher validation accuracy with mode as 88.99% against the models trained on a daily weekly sample which yielded a mode of 83.33%. The difference can be attributed to the fact that the images generated with more time series data points will be larger in dimension, thereby providing more information entropy per image for the neural networks to learn from. However, as shown in Table 3, some of the images using a weekly sampling window required scaling down to fit into the VRAM of the GPU used in this study. The weekly sampling window also decreased the number of 'leak' images generated with the same time series data, which means that to acquire an equivalent amount of 'leak' images, a larger time series dataset with more tube leak occurrences would be required. It could be impractical for some boilers to obtain a large amount of tube leak data since tube leaks are rare events under normal boiler operation.

The sampling intervals did not indicate a significant influence on the validation accuracy and true positive rates of the models. Using a sampling interval of five minutes yielded the highest pixel count for the generated images and therefore demanded the most VRAM out of all the sampling interval options. A higher pixel count could lead to a better training result as it increases the number of parameters in the model.

6.2. Effect of neural network architecture

The purpose of transfer learning with pretrained ResNet50 was to minimise the amount of labelled data necessary to acquire high classification accuracy while reducing the engineering time to construct a neural network. Acquiring an adequate amount of labelled data is difficult due to the rarity of boiler tube leaks and the construction of a deep convolution neural network from scratch requires resources and expertise that will discourage adoption by boiler operators.

A simple convolution neural network was constructed and trained on the image datasets encoded by the GADF with a sampling interval of five minutes and daily sampling window for comparison with the ResNet50 architecture.

Table 3. Summary of model performance and relevant metrics.

Algorithm	Variant	Sample Window	Time interval (min)	Train Loss	Validation Loss	Train Accuracy	Validation Accuracy	Number of 'Normal' Images	Number of 'Leak' Images	Leak-to-normal Ratio	Leak-to-normal Ratio	tn	fp	fn	tp	True Positive Rate	True Negative Rate	Precision
GAF	Difference	Daily	5	0.0295	0.1184	99.76%	96.64%	451	451	1.00	288	119	0	8	111	93.28%	100.00%	100.00%
			10	0.0200	0.2193	99.85%	91.41%	341	341	1.00	144	56	8	3	61	95.31%	87.50%	88.41%
			30	0.0488	0.6339	99.25%	75.40%	336	336	1.00	48	58	5	10	53	84.13%	92.06%	91.38%
			60	0.0786	0.7597	97.45%	76.98%	333	333	1.00	24	44	19	12	51	80.95%	69.84%	72.86%
		Weekly	5	0.2757	0.2641	87.07%	88.24%	58	58	1.00	504 (2016)	15	2	2	15	88.24%	88.24%	88.24%
			10	0.2616	0.4397	91.49%	88.89%	47	47	1.00	504 (1008)	7	2	0	9	100.00%	77.78%	81.82%
			30	0.2593	0.3104	89.36%	94.44%	47	47	1.00	336	9	0	1	8	88.89%	100.00%	100.00%
			60	0.1956	0.3406	96.81%	94.44%	47	47	1.00	168	8	0	1	9	90.00%	100.00%	100.00%
			5	0.1852	0.3247	93.09%	85.11%	174	58	0.33	504 (2016)	16	2	2	15	88.24%	88.89%	88.24%
			10	0.2141	0.3176	91.49%	83.33%	141	47	0.33	504 (1008)	7	2	1	26	96.30%	77.78%	92.86%
			30	0.1420	0.3996	95.21%	83.33%	141	47	0.33	336	7	2	0	27	100.00%	77.78%	93.10%
			60	0.0836	0.2841	99.47%	88.89%	141	47	0.33	168	4	5	1	26	96.30%	44.44%	83.87%
	Sum	Daily	5	0.0468	0.2475	99.12%	90.63%	341	341	1.00	288	60	4	8	56	87.50%	93.75%	93.33%
			10	0.0216	0.2718	99.27%	89.84%	341	341	1.00	144	55	9	4	60	93.75%	85.94%	86.96%
			30	0.0219	0.3942	99.55%	83.33%	336	336	1.00	48	53	10	11	52	82.54%	84.13%	83.87%
			60	0.0522	0.6116	98.50%	81.75%	333	333	1.00	24	51	12	11	52	82.54%	80.95%	81.25%
		Weekly	5	0.4542	0.5677	78.72%	61.11%	47	47	1.00	504 (2016)	3	6	1	8	88.89%	33.33%	57.14%
			10	0.2924	0.4579	90.43%	77.78%	47	47	1.00	504 (1008)	6	3	1	8	88.89%	66.67%	72.73%
			30	0.2846	0.4761	90.43%	88.89%	47	47	1.00	336	7	2	0	9	100.00%	77.78%	81.82%
			60	0.0407	0.2397	99.27%	89.94%	47	47	1.00	168	8	1	4	5	55.56%	88.89%	83.33%
			5	0.1911	0.3316	92.55%	88.89%	141	47	0.33	504 (2016)	7	2	3	24	88.89%	77.78%	92.31%
			10	0.2516	0.3588	92.02%	83.33%	141	47	0.33	504 (1008)	4	5	1	26	96.30%	44.44%	83.87%
			30	0.1407	0.4353	95.21%	83.33%	141	47	0.33	336	6	3	3	24	88.89%	66.67%	88.89%
			60	0.0980	0.2858	97.87%	88.89%	141	47	0.33	168	7	2	2	25	92.59%	77.78%	92.59%
MTF		Daily	5	0.0275	0.3648	99.71%	85.94%	341	341	1.00	288	51	13	5	59	92.19%	79.69%	81.94%
			10	0.0132	0.3564	100.00%	86.72%	341	341	1.00	144	52	12	5	59	92.19%	81.25%	83.10%
			30	0.0208	0.5764	99.55%	84.13%	336	336	1.00	48	55	8	12	51	80.95%	87.30%	86.44%
			60	0.0919	0.6900	98.05%	70.63%	333	333	1.00	24	46	17	20	43	68.25%	73.02%	71.67%
		Weekly	5	0.3194	0.4256	87.23%	77.78%	47	47	1.00	504 (2016)	7	2	2	7	77.78%	77.78%	77.78%
			10	0.3044	0.3734	89.36%	83.33%	47	47	1.00	504 (1008)	7	2	0	9	100.00%	77.78%	81.82%
			30	0.2603	0.3439	90.43%	94.44%	47	47	1.00	336	9	0	1	8	88.89%	100.00%	100.00%
			60	0.2066	0.4196	95.74%	83.33%	47	47	1.00	168	6	3	0	9	100.00%	66.67%	75.00%
			5	0.2623	0.1862	90.43%	97.22%	141	47	0.33	504 (2016)	8	1	0	27	100.00%	88.89%	96.43%
			10	0.1972	0.2115	92.02%	94.44%	141	47	0.33	504 (1008)	7	2	0	27	100.00%	77.78%	93.10%
			30	0.2020	0.3911	90.43%	88.89%	141	47	0.33	336	6	3	1	26	96.30%	66.67%	89.66%
			60	0.0969	0.2865	97.34%	91.67%	141	47	0.33	168	7	2	1	26	96.30%	77.78%	92.86%
RP	Cross	Daily	5	0.0168	0.3651	99.85%	89.06%	341	341	1.00	288	56	8	5	89	94.68%	87.50%	91.75%
			10	0.0121	0.4152	100.00%	88.28%	341	341	1.00	144	58	6	9	55	85.94%	90.63%	90.16%
			30	0.0231	0.5696	99.55%	80.95%	336	336	1.00	48	53	10	14	49	77.78%	84.13%	83.05%
			60	0.0847	0.6109	97.90%	77.78%	333	333	1.00	24	49	14	14	49	77.78%	77.78%	77.78%
		Weekly	5	0.3113	0.3568	86.17%	88.89%	47	47	1.00	504 (2016)	8	1	1	8	88.89%	88.89%	88.89%
			10	0.2831	0.3590	90.43%	83.33%	47	47	1.00	504 (1008)	6	3	0	9	100.00%	66.67%	75.00%
			30	0.2812	0.4292	89.36%	77.78%	47	47	1.00	336	6	3	1	8	88.89%	66.67%	72.73%
			60	0.2377	0.3789	93.62%	88.89%	47	47	1.00	168	7	2	0	9	100.00%	77.78%	81.82%
			5	0.1483	0.2418	93.62%	91.67%	141	47	0.33	504 (2016)	8	1	2	25	92.59%	88.89%	96.15%
			10	0.2054	0.3121	91.49%	86.11%	141	47	0.33	504 (1008)	7	2	1	26	96.30%	77.78%	92.86%
			30	0.1792	0.2515	93.09%	91.67%	141	47	0.33	336	7	2	3	24	88.89%	77.78%	92.31%
			60	0.1192	0.3330	95.74%	83.33%	141	47	0.33	168	4	5	1	26	96.30%	44.44%	83.87%
	Joint	Daily	5	0.0313	0.2764	99.71%	89.84%	341	341	1.00	288	57	7	6	58	90.63%	89.06%	89.23%
			10	0.0185	0.2347	99.71%	92.19%	341	341	1.00	144	60	4	6	58	90.63%	93.75%	93.55%
			30	0.0185	0.5342	99.85%	83.33%	336	336	1.00	48	52	11	10	53	84.13%	82.54%	82.81%
			60	0.0656	0.4497	98.95%	83.33%	333	333	1.00	24	56	7	14	49	77.78%	88.89%	87.50%
		Weekly	5	0.2541	0.3445	90.43%	88.89%	47	47	1.00	504 (2016)	8	1	1	8	88.89%	88.89%	88.89%
			10	0.2336	0.3895	91.49%	88.89%	47	47	1.00	504 (1008)	8	1	1	8	88.89%	88.89%	88.89%
			30	0.2588	0.4005	89.36%	88.89%	47	47	1.00	336	7	2	0	9	100.00%	77.78%	81.82%
			60	0.1794	0.4455	95.74%	88.89%	47	47	1.00	168	7	2	0	9	100.00%	77.78%	81.82%
			5	0.1991	0.2251	90.96%	88.89%	141	47	0.33	504 (2016)	7	2	2	25	92.59%	77.78%	92.59%
			10	0.1639	0.2794	93.62%	88.89%	141	47	0.33	504 (1008)	8	1	3	24	88.89%	88.89%	96.00%
			30	0.1285	0.3544	95.21%	86.11%	141	47	0.33	336	5	4	1	26	96.30%	55.56%	86.67%
			60	0.1112	0.3562	97.34%	86.11%	141	47	0.33	168	7	2	3	24	88.89%	77.78%	92.31%

Table 4. Summary statistics of the validation accuracies and true positive rates of different encoding algorithms sampling intervals and sampling windows.

	Validation accuracy					True positive rate				
Encoding algorithm	GAF		MTF	RP		GAF		MTF	RP	
	Difference	Sum		Cross	Joint	Difference	Sum		Cross	Joint
Max accuracy	96.64%	90.63%	97.22%	91.67%	92.19%	100.00%	100.00%	100.00%	100.00%	100.00%
Mean	87.26%	83.98%	86.54%	85.65%	87.85%	91.80%	87.19%	91.07%	90.67%	90.63%
Mode	88.89%	83.33%	83.33%	77.78%	88.89%	88.24%	88.89%	100.00%	88.89%	88.89%
Standard deviation	6.72%	8.28%	7.55%	4.94%	2.63%	6.05%	11.18%	10.36%	7.52%	6.26%

Table 5. Summary statistics of the validation accuracies and true positive rates of different sampling intervals.

	Validation accuracy				True positive rate			
Sampling interval (m)	5	10	30	60	5	10	30	60
Max accuracy	97.22%	94.44%	94.44%	94.44%	100.00%	100.00%	100.00%	100.00%
Mean	87.25%	87.12%	85.66%	84.99%	90.22%	94.22%	89.77%	86.88%
Mode	88.89%	83.33%	83.33%	88.89%	88.89%	100.00%	88.89%	96.30%
Standard deviation	8.53%	4.32%	5.58%	6.33%	4.76%	4.77%	7.27%	13.03%

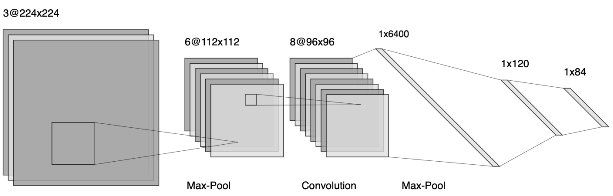


Figure 5. The architecture of the simple CNN constructed to compare its performance with ResNet50.

From the plots of accuracies and losses above, both ResNet50 and the simple CNN had converged at around the 10th epoch. The upward trending accuracy plots and the downward trending loss plots indicate that the neural networks were learning as the epochs progressed. The simple CNN had converged on lower accuracies and higher losses, scoring a validation accuracy of 76.05% and validation loss of 0.5106. It was found that the performance of the neural network increased with the number of layers (or ‘network depth’) (Szegedy et al., 2014). However, upon reaching a certain depth, the network suffered from degradation (He et al., 2015). The residual blocks of the ResNet50 architecture circumvent the issues that come with degradation and transfer learning. Since the neural network is pretrained, it requires a relatively small amount of labelled data.

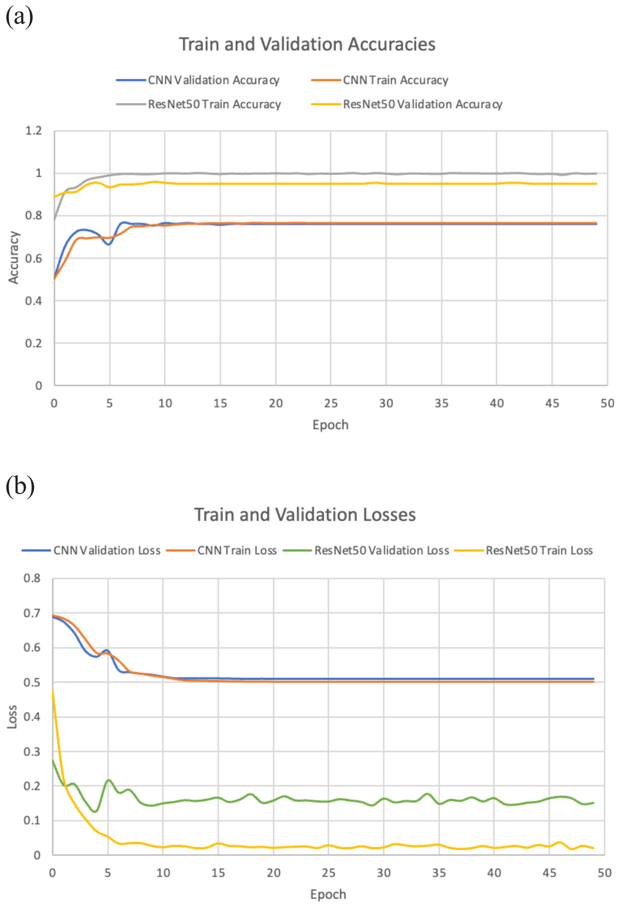


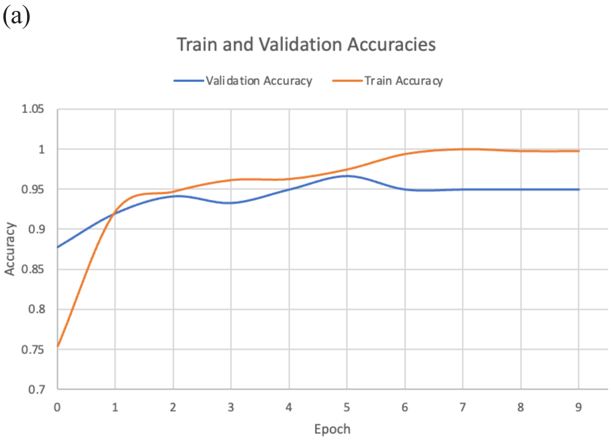
Figure 6. The training and validation (a) accuracies; and (b) model losses of the simple CNN and ResNet50 in 50 epochs.

6.3. Effect of the number of epochs

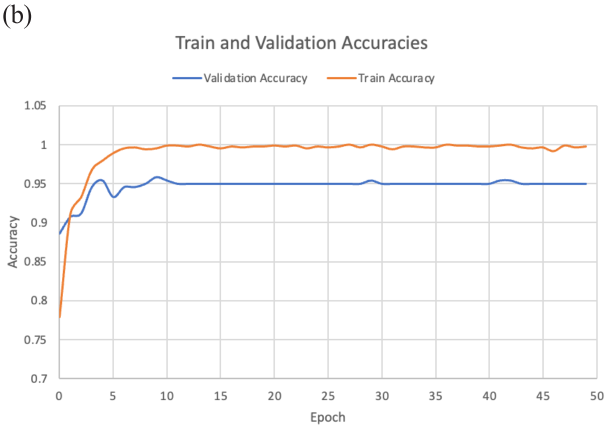
In order to minimise the training difficulty for practical deployment, the pretrained ResNet50 neural network was utilised instead of training the neural network from scratch using the encoded images. A pretrained convolution neural network such as ResNet50 can already detect abstract geometrical patterns, so the pretrained weights are activated by the patterns and features in the encoded images. This can reduce the number of epochs in the training for the model solution to converge.

Using the images generated by the GADF with five-minute interval and a daily sampling window, the model was trained with 10 epochs and 50 epochs to compare the effect of the number of epochs on the model performance. From Figure 6, both training and validation accuracies converged at around the 10th epoch and underwent no further improvement in the subsequent training epochs. Meanwhile, the training and validation losses also converged at around the 10th epoch. As such, the proposed model can be trained using 10 epochs only.

(a)



(b)



(c)

(d)

Figure 7. Training and validation accuracies for (a) 10 epochs; and (b) 50 epochs. Training and validation losses for (c) 10 epochs; and (d) 50 epochs.

Since there is no mathematical treatment to pre-determine the optimal number of epochs, it is recommended that practitioners should start with 10 epochs and observe whether the losses and accuracies converge, and then adjust the number of epochs accordingly.

6.4. Deploying the model in a practical environment

When selecting a model from the candidates to deploy, it is important to maximise the precision of the predictions. A high-precision model means a low false positive rate, which would significantly improve operators' confidence in identifying a leak when the model detects a positive leak in the time series data. The model had a true positive rate of 93.28%; assuming that a model attains a 1% false positive rate, and the probability of tube leak occurrence is 0.1%, by Bayes' theorem:

$$Pr(Leak \mid Positive) = \frac{Pr(Positive \mid Leak)\, Pr(Leak)}{Pr(Positive)}, \quad (13)$$

where

$$Pr(Positive) = Pr(Positive \mid Leak)\, Pr(Leak) + Pr(Positive \mid Normal)\, Pr(Normal)\,. \quad (14)$$

Therefore, *Pr(Leak | Positive)* would be 8.54%, given the assumptions. In other words, out of 100 positive identifications by the model, only around eight would be tube leaks. This phenomenon is due to the fact that tube leaks are rare occurrences and therefore most positive identifications are false positives. By halving the false positive rate to 0.5%, the *Pr(Leak | Positive)* would increase to 15.4%. This incentivises choosing a model with the lowest false positive rate.

6.5. Feasibility to adapt the model to other problems

The design of the proposed model is not restricted to HRSG tube leak problems. The goal of the design is to encode distinctive time series patterns into images so that a pretrained neural network may learn to accurately classify time series data with minimal effort in neural network construction, computation cost in training, data collection and labelling. Operators of other boiler types may adapt the proposed model according to their equipment, if there is an adequate amount of failure data. Operators can achieve this by gathering and labelling time series data during which the boiler suffered damage or failure, followed by encoding the time series data and splitting the images into training and validation sets. Operators are encouraged to experiment with different encoding algorithms under different sampling intervals and windows to optimise the model for the best classification results. To maximise the robustness of the subsequent model, it is crucial to include time series data that reflects normal and typical operation as the 'normal' set. As such, the neural network would be able to learn the effect of general wear and tear on the time series pattern. Tampering with the 'normal' set may yield a high accuracy in training but low accuracy when deployed. In principle, the proposed model should also be able to be applied to conditional monitoring of equipment unrelated to boilers as well, so long as there are enough records of the occurrence of the incidents which cause a certain distinctive time series pattern to emerge.

7. Conclusion

This study aims to propose a model for early tube leak detection in HRSGs based on transfer learning, and describes its design and capability. The model offers several improvements for tube leak detection based on neural networks as proposed in the past. The model proposed in this study was trained on real operational data and tube leak history. The use of transfer learning with pretrained ResNet50 architecture significantly reduced the engineering time and computation cost due to training the neural network. In the experimentations, it was found that the encoding algorithm setting, i.e. encoding algorithm variants, and data sampling interval and window, had a considerable impact on the final performance of the model. A simple CNN was also constructed for comparison to validate the use of ResNet50 architecture. Boiler operators and deep learning practitioners are encouraged to adopt the proposed design for other time series classification problems for enhanced classification accuracy with minimal engineering efforts.

Notes on contributor

Mr Hon Fung Chow is an engineer in CLP working at Black Point Power Station. His specialties include O&M strategies, CCGT operation and HRSG maintenance. He is currently pursuing the application of machine learning to enhance business performance. Mr Chow received his M.Sc. degree in Mechanical Engineering from The University of Hong Kong and his B.Eng. degree in Mechanical Engineering from The Hong Kong University of Science and Technology.

References

[1] Anujaa P, Babysree R, Divya Prabha S, Karthika D and Selvaraj R (2020). Acoustic Steam Leak Detection System. *ETCAN - 2020 Conference Proceedings*, 8(9), pp. 1-5.

[2] Eckmann J-P, Oliffson Kamphorst S and Ruelle D (1987). Recurrence Plots of Dynamical Systems. *Europhysics Letters* (EPL), 4(9), pp. 973-977.

[3] He K, Zhang X, Ren S and Sun J (2015). Deep Residual Learning for Image Recognition. [online]. arXiv. Available at: <https://arxiv.org/abs/1512.03385>. [Accessed on 2 August 2021].

[4] Hu K, Zhu F, Chen J, Noda N-A, Han W and Sano Y (2019). Simulation of Thermal Stress and Fatigue Life Prediction of High Speed Steel Work Roll during Hot Rolling Considering the Initial Residual Stress. *Metals*, 9(9), pp. 966.

[5] Ismail FB, Singh D, Maisurah N and Musa ABB (2016). Early tube leak detection system for steam boiler at KEV power plant. *MATEC Web of Conferences*, 74, pp. 00006.

[6] Kingma DP and Ba J (2014). Adam: A Method for Stochastic Optimization. [online]. arXivI. Available at: <https://arxiv.org/abs/1412.6980v9>. [Accessed on 2 August 2021].

[7] Kotsiantis S, Kanellopoulos D and Pintelas PE (2005). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1), pp. 25-36.

[8] Ramezani MG, Hasanian M, Golchinfar B and Saboonchi H (2020). Automatic boiler tube leak detection with deep bidirectional LSTM neural networks of acoustic emission signals. In: Huang H et al. (eds) *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2020*. Bellingham: Society of Photo-Optical Instrumentation Engineers.

[9] Ruder S (2016). *An overview of gradient descent optimization algorithms*. [online]. arXiv. Available at: <https://arxiv.org/abs/1609.04747>. [Accessed on 2 August 2021].

[10] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC and Li F-F (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), pp. 211-252.

[11] Hamid SS, Jamal DN and Shajahan M (2013). Automatic Detection and Analysis of Boiler Tube Leakage System. *International Journal of Computer Applications*, 84(16), pp. 19-23.

[12] Sohaib M and Kim J-M (2019). Data Driven Leakage Detection and Classification of a Boiler Tube. *Applied Sciences*, 9(12), pp. 2450.

[13] IEEE Standard Association (2007). *IEEE 762-2006 - IEEE Standard Definitions for Use in Reporting Electric Generating Unit Reliability, Availability, and Productivity*. [online]. Available at: <https://standards.ieee.org/standard/762-2006.html>. [Accessed on 2 August 2021].

[14] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A (2014). *Going Deeper with Convolutions*. [online]. arXiv. Available at: <https://arxiv.org/abs/1409.4842>. [Accessed on 2 August 2021].

[15] Wan Q, Koch DB and Morris K (1993). *Multichannel spectral analysis for tube leak detection*. In: Proceedings of Southeastcon '93. Charlotte, USA: IEEE, pp. 4.

[16] Wang Z and Oates T (2015). *Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks*. In: AAAI-15. Palo Alto: The AAAI Press, pp. 40-46.

[17] Wei Q and Dunbrack RL Jr (2013). The Role of Balanced Training and Testing Data Sets for Binary Classifiers in Bioinformatics. *PLoS ONE*, 8(7), pp. e67863.

[18] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H and He Q (2019). A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1), pp. 43-76.