Time Series to Images: Monitoring the Condition of Industrial Assets with Deep Learning Image Processing Algorithms

Gabriel Rodriguez Garcia
ETH Zürich,
Zürich, Switzerland

Gabriel Michau
ETH Zürich,
Zürich, Switzerland

Mélanie Ducoffe
Airbus AI Research,
Toulouse France

Jayant Sen Gupta
Airbus AI Research,
Toulouse France

Olga Fink
ETH Zürich,
Zürich, Switzerland

13rd of May 2020

Abstract

The ability to detect anomalies in time series is considered as highly valuable within plenty of application domains. The sequential nature of time series objects is responsible for an additional feature complexity, ultimately requiring specialized approaches for solving the task. Essential characteristics of time series, laying outside the time domain, are often difficult to capture with state-of-the-art anomaly detection methods, when no transformations on the time series have been applied. Inspired by the success of deep learning methods in computer vision, several studies have proposed to transform time-series into image-like representations, leading to very promising results. However, most of the previous studies implementing time-series to image encodings have focused on the supervised classification. The application to unsupervised anomaly detection tasks has been limited.

The paper has the following contributions: First, we evaluate the application of six time-series to image encodings to DL algorithms: Gramian Angular Field, Markov Transition Field, Recurrence Plot, Grey Scale Encoding, Spectrogram and Scalogram. Second, we propose modifications of the original encoding definitions, to make them more robust to the variability in large datasets. And third, we provide a comprehensive comparison between using the raw time series directly and the different encodings, with and without the proposed improvements. The comparison is performed on a dataset collected and released by Airbus SAS 2018, containing highly complex vibration measurements from real helicopters flight tests. The different encodings provide competitive results for anomaly detection.

1 Introduction

With the globalisation of many industrial markets, it becomes increasingly important for the operators to optimise the use and the operating costs of their assets. This optimisation requires the ability to maintain a high availability at regulatory safety levels. Since the maintenance of industrial assets accounts for a large part of the operating costs, the ability to monitor the system to perform maintenance based on its condition would benefit the operators greatly. The ability to detect anomalies could help in the short term to reduce the amount of required unscheduled maintenance. Once it has been demonstrated that the algorithms detect anomalies reliably, the algorithms could also help to reduce the requirements in preventive maintenance.

With the recent decrease in the costs of sensors and the possibility to connect large monitoring environments with the industrial Internet of Things, large data streams are nowadays captured and can be accessed in real time even for high frequency data. The implementation of condition-based maintenance is currently mostly lacking the tools to automatically and reliably process such large quantities of information in real time. In fact, the handling of high frequency data faces very specific challenges, such as the variable length of the data streams and the fact that the fault signatures can appear at completely different temporal or spectral scales, for example over several hours of continuous operation monitored at several thousands or millions Hertz (Valle et al. 2009).

The recent development of deep learning offers the possibility to automatically learn from large data sets the signatures that will maximise its training objective. Deep learning offers nowadays state-of-the art results in supervised setups. That is, in cases where the algorithm is trained with enough representative samples of all conditions to discriminate (Qu et al. 2019; Fawaz et al. 2019). Anomaly detection, however, is a more challenging task since an anomaly is only defined with respect to a main class, and not in itself (Zimek and Schubert 2017). In the industry, it is not possible to define, and, therefore, to collect, representative samples of all possible types of anomalies. Consequently, ways need to be devised to train the anomaly detector on data which is almost exclusively collected in normal conditions (Michau et al. 2020). Indeed, complex industrial systems are almost exclusively operating in healthy conditions, since they are otherwise maintained to avoid major consequences. Healthy operation data are usually highly available, and the task of anomaly detection can, therefore, be defined as the learning of a representation of the healthy operating conditions to detect any deviation from it. Previous works have shown that, for such approaches, monitoring the residuals of an auto-encoder trained on the main class could solve this issue (Hu, Palm, and Fink 2017; Chouhan, Khan, and Rasheed Khan 2019). The recent developments in deep learning have however been driven by image processing, with some major success in the use of auto-encoders (Zhang, Liu, and Fu 2019). Also, recent works suggested several transformations of time series to images to benefit from the latest advances in supervised deep learning (Gondara 2016; Krummenacher et al. 2017; Sasirekha and Thangavel 2020). In addition, encoding time series to images can help to emphasize, capture or condense local patterns that would otherwise be spread over time. To the best of our knowledge, these approaches have, however, never been tested on anomaly detection tasks for time series.

In this paper, we propose to compare, in an unsupervised setup, six time series to image encoding, that have successfully provided improvements on supervised case studies in the literature. These six encodings are the Gramian Angular Field, the Markov Transition Field (Wang and Oates 2015), the Recurrence Plot (Marwan et al. 2007), the Grey Scale Encoding (Xu et al. 2019), the Spectrogram and the Scalogram. The last two, originate from spectral analysis in Signal Processing and are very common tools for signal analysis (Boashash 2015). The framework, described in Figure 1, consists in first, the time series splitting, second, the encoding of each sub series, and last the monitoring of their aggregated residuals when fed through a convolutional auto-encoder (CAE). We compare the results with a similar framework, taking the raw sub series as input, and using a one-dimensional CNN. We demonstrate that, first, the suggested modifications of the encodings are required to obtain meaningful results, and that second, the encodings all offer an advantage for the detection of anomalous time series. We discuss last the computational requirement of the proposed framework.

The remainder of this paper is organised as follows: Section 2 describes the proposed anomaly detection framework; Section 3 presents the different image encodings used to represent time series data; This methodology is applied with the different image encodings to an industrial use case introduced in Section 4; finally, we discuss the methods and the perspectives.

2   Framework for Anomaly Detection

The proposed framework, illustrated in Figure 1, consists in the detection of anomalous input reconstruction based on an auto-encoder neural network with different types on time-series to image encodings as input. The underlying idea is to train an autoencoder to reconstruct its own input while compressing it to a lower dimensional latent space, acting as a bottleneck. The network will learn to reconstruct as well as possible its input data (either time series or images) following the same distribution as the ones it has been trained on. If the the auto-encoder is trained on healthy data, its reconstruction ability will be poor if presented with anomalous conditions that deviate from the healthy conditions used during training. Monitoring the reconstruction error with respect to a threshold allows the detection of such anomalous patterns.

Since the original time series in the dataset are comparably long (several thousands of values), it raises two types of challenges. First, since the monitored residual is an aggregated measure over the reconstruction, localised anomalies may be unnoticeable at the global scale. Second, for most encodings, computational time and size evolve according to a power law of the input size. To overcome these two challenges, we propose to first sub-divide the time series into smaller sub-series of size l_1. These sub-series are then encoded to images of size $l_2 \times l_2$ which are reconstructed by
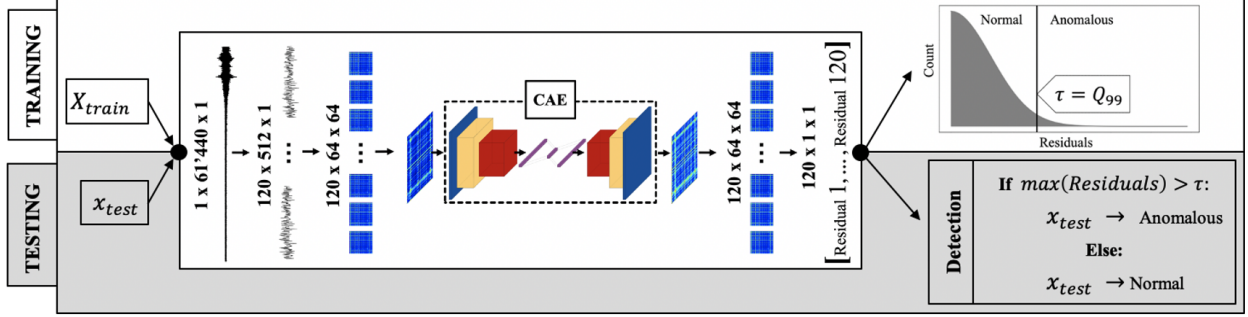
Figure 1: **Framework**. The network is trained on N time series in the training data set X_{train} and the threshold is set as the 99-th percentile over the sample of 120×N residuals (108×N for the GS encoding). For testing an arbitrary test sample x_{test}, each of the 120 slices (108 for the GS encoding) is reconstructed by the CAE and the largest residual is compared with the threshold τ to detect anomalies.. When using the raw time series, the same framework is followed without the encoding step and with 1-D convolutions in the auto-encoder.

the neural network and their residuals computed and aggregated. We define the residual of the k-th slice, Res_k, as the ℓ_1-norm of the difference between the input, x_{or}, of size l and its reconstruction, x_{rc} as:

$$\text{Res}_k = \sum_{i=1}^{l}(|x_{\text{or}}^i - x_{\text{rc}}^i|), \tag{1}$$

where $l = l_1$ for the baseline model using the raw sub-series without any encodings as input to a one dimensional CNN, and $l = l_2 \times l_2$ for all the other cases.

During training, only time series from healthy conditions are used. Based on the distribution of the residuals of all sub-series, we extract the 99-th percentile to define the detection threshold τ. In the test set, for each time series, we monitor the maximum residual over its sub-series, and compare it to the threshold τ. If it exceeds the threshold, the time series is detected as anomalous. Monitoring the maximum of the residuals enables the detection of local anomalies that would impact one or few of the sub sequences.

3   Time Series To Image Encoding

3.1   Gramian Angular Field (GAF)

The Gramian Angular Field (Wang and Oates 2015) consists of a two-step process to transform a time series into a matrix. First, the transformation from the space (*Time*×*Value*) to polar coordinates is performed. Then, the Gramian matrix in that new space is computed.

The transformation to polar coordinates consists of performing the following operation:

$$\begin{cases} \phi = \arccos(X_{norm,i}); \phi \in [0, \pi], \\ r = \frac{n}{L}; r \in \mathbb{R}^+, \end{cases} \tag{2}$$

where L is scaling the temporal dimension but does not influence the final matrix. This transformation is bijective and absolute temporal relations are preserved(Wang and Oates 2015). Since Equation 2 is only defined for $X \in [-1, 1]$, it requires first the scaling of the data. Wang and Oates 2015 propose to scale the input X as

$$X = (x_1, x_2, ..., x_N), \quad X_{norm,i} = \frac{x_i - \text{UB} + (x_i - \text{LB})}{\text{UB} - \text{LB}}; \in [-1, 1], \tag{3}$$

where, UB and LB refer to the upper and lower bound parameters, defined originally as $\text{UB} = \max(x)$ and $\text{LB} = \min(x)$. The final Gramian matrix can be computed as follows:

$$GAF = X_{norm}^T \cdot X_{norm} - \sqrt{I - X_{norm}^2}^T \cdot \sqrt{I - X_{norm}^2} \tag{4}$$

3

$$GAF = \begin{pmatrix} \cos(\phi_1 + \phi_1) & \cos(\phi_1 + \phi_2) & \cdots & \cos(\phi_1 + \phi_N) \\ \cos(\phi_2 + \phi_1) & \cos(\phi_2 + \phi_2) & \cdots & \cos(\phi_2 + \phi_N) \\ \vdots & \vdots & \ddots & \vdots \\ \cos(\phi_N + \phi_1) & \cos(\phi_N + \phi_2) & \cdots & \cos(\phi_N + \phi_N) \end{pmatrix} \tag{5}$$

where I refers to a unit row vector and the superscript $(\cdot)^T$ refers to the transpose operator. By analogy to Gram matrices $G_{ij} = \langle x_i, x_j \rangle$, where $\langle \cdot, \cdot \rangle$ is an inner product, the matrix is named as Gramian Angular Field. One can note, however, that $\cos(x_i + x_j)$ does not satisfy all the conditions of inner-products. In addition, it is noted that, the matrix does not depend on r. Therefore, it does not depend on time and not on the choice of L in Eq 2. As a consequence, the GAF transformation is not reversible, meaning the GAF no longer allows for a unique reconstruction of the time series. This leads to some loss of information after the encoding has been applied. However, since there is often only moderate overlap in the inverse space, the input can be roughly approximated (Wang and Oates 2015).

Proposed modification: In our study, we propose to scale the training samples based on the full training dataset in order to keep the relationships and differences between the samples. Therefore, we propose to define UB and LB based on the training set distribution. If in the test set, a sample contains values exceeding these bounds, we propose to clip the values to the bound.

3.2 Markov Transition Field (MTF)

The Markov Transisition Field (Wang and Oates 2015), consists of building the matrix of probabilities to observe the change of value between any pair of points in the time series. First, the time series is discretized. Then, the transitions w from one bin to another are counted and normalised over every two consecutive data-points in the whole training set. Last, the MTF matrix contains for every pair of points in the time series the transition probability between the two bins the points belong to.

Assuming a discretization of the time series based on $Q + 1$ bin edges, that is, Q bins q_i, the transitions are computed over every two consecutive points in the training set as

$$\forall (i, j) \in [\![1, Q+1]\!], \quad \widehat{w}_{ij} = \text{number of transitions } q_i \rightarrow q_j \tag{6}$$

$$\forall (i, j) \in [\![1, Q+1]\!], \quad w_{ij} = \frac{\widehat{w}_{ij}}{\sum_j \widehat{w}_{ij}} \tag{7}$$

$W = (w_{ij})$ is the Markov Transition matrix, which at this stage does not capture any temporal relations. The final $N \times N$ matrix, called Markov Transition Field, is computed as follows:

$$MTF = \begin{pmatrix} w_{ij|X_1 \in q_i, X_1 \in q_j} & \cdots & w_{ij|X_1 \in q_i, X_N \in q_j} \\ w_{ij|X_2 \in q_i, X_1 \in q_j} & \cdots & w_{ij|X_2 \in q_i, X_N \in q_j} \\ \vdots & \ddots & \vdots \\ w_{ij|X_N \in q_i, X_1 \in q_j} & \cdots & w_{ij|X_N \in q_i, X_N \in q_j} \end{pmatrix} \tag{8}$$

This transformation of the time series is non reversible, meaning that there is a substantial information loss. In particular, the discretization of the time series plays a crucial role in the amount of information kept or lost by the transformation. A large number of bins might lead to sparsity in the image, while small numbers of bins might lead to substantial information loss (Wang and Oates 2015).

Proposed modification: In the original formulation of the MTF, Wang and Oates 2015 proposed to partition the range of values into a number of equi-sized segments. Since in many datasets, the distribution of values tends to follow long tailed distributions, choosing the optimal number and width of bins becomes a challenging step requiring manual tuning. A large bin width would aggregate most values into the bins the closest to the mean, while a small bin width would lead to sparsity in the extreme bins.

To overcome this high dependency on the optimal parameter setting, we propose to extend the original formulation of the bin choice and assignments and to use instead the Symbolic Aggregate approXimation (SAX) algorithm (Lin et al. 2007) that performs a non-uniform bin assignment such that the distribution of the bin frequency roughly follows a Gaussian distribution. As such, setting the right number of bins is a less critical task, since the risk of sparsity is drastically reduced. An arbitrarily large number of bins can be used, and the choice mostly depends on the computational requirements for computing the transitions.

3.3 Recurrence Plot (RP)

The recurrence plot (RP) (Kamphorst and Ruelle 1987) transforms a time series into a matrix of recurrences to reveal in which points some trajectories return to a previously visited state. In this paper, we follow a non-binarized version of RP, as proposed by Souza, Silva, and Batista 2014. The RP encoding consists in the following transformation:

$$\text{RP}_{ij} = ||S_i - S_j||, \quad i,j \in [\![1, K]\!],$$ (9)

where the time series is split into K sub-sequences $(S_i)_{i \in [\![1, K]\!]}$. Here, we set the sub-sequence length to 1 (we only consider the difference between single values), and defined $K = l_2$.

Proposed modification: This transformation is non-reversible and also looses information on the values in the time series since only the differences in values are kept. We propose, therefore, the following transformation:

$$\widehat{\text{RP}} = \text{RP} + \text{mean}(X).$$ (10)

This transformation aims at capturing recurrences in individual time series around their mean.

3.4 Gray-Scale (GS) Encoding

The Grey-scale encoding has been revived by Wen et al. 2018 for the purpose of fault diagnosis in manufacturing systems using convolutional neural networks. The transformation consists of first, the reshaping of a 1D time series into a collection of K sub-series of size K_1, and second, the rescaling of the values as color encoding values (e.g., 8-bit integer). In this work, for consistency with other encodings, we aim at $K = K_1$ and compute the stride s between the start of each sub-series accordingly. The GS encoding corresponds, therefore, to the following operation:

$$GS_{i,j} = \text{round}\left(P \cdot \frac{(x_{(i-1) \cdot s + j} - \text{LB})}{\text{UB} - \text{LB}}\right), \quad i,j \in [\![1, K]\!],$$ (11)

where the operator round rounds the value to the nearest integer, P is a scaling factor, UB and LB are respectively an upper and lower bound to be used for the scaling of X.

Proposed modification: In the original formulation, the authors suggested to use $\text{UB} = \max(X)$, $\text{LB} = \min(X)$ and $P = 255$ such that the resulting values can be interpreted as 8-bit encoded grey scale images. In this work, we propose first, to define UB and LB over the entire training dataset, choosing these bounds in accordance with the GAF encoding. Furthermore, since most common deep learning implementations use 32-bit inputs, we propose to evaluate the necessity of the conversion to integers for this encoding.

3.5 Spectrogram (SP) and Scalogram (SC)

To complement the encodings described above, we propose to add to the study the more traditional time-frequency analysis. Since the spectro-temporal description of a time series provides 2-dimensional representations, it is customary in the literature to interpret these representations directly as images (Verstraete et al. 2017). In this work, we propose two popular transformations: the spectrogram, resulting from the short-time Fourier Transform (STFT), and the scalogram, resulting from the Discrete Wavelet Transform (DWT). For more information on these transformations, the reader is referred to the book from Boashash 2015.

Both the STFT and the CWT rely on the principle of convolving the time series with a windowing function. The difference between the two approaches is that for STFT, the window is fixed and the Fourier Transfrom is applied to the convolved signal, while for the CWT, the window is based on a wavelet function, whose scaling enables the exploration of the spectrum at different levels. For the STFT, it is therefore required to set three hyperparameters: the window type, its size and its temporal stride. The DWT requires the a-priory definition of the wavelet family, the scales explored and the temporal stride.

4 Detection of Anomalous Conditions in Flight Test

4.1 Helicopters accelerometers use case

The use case on which we illustrate different ways to encode time series into images is relative to flight test helicopters vibration measurements. The dataset has been collected and released by Airbus SAS 2018[1]. A main challenge in flight

---

[1] https://doi.org/10.3929/ethz-b-000415151
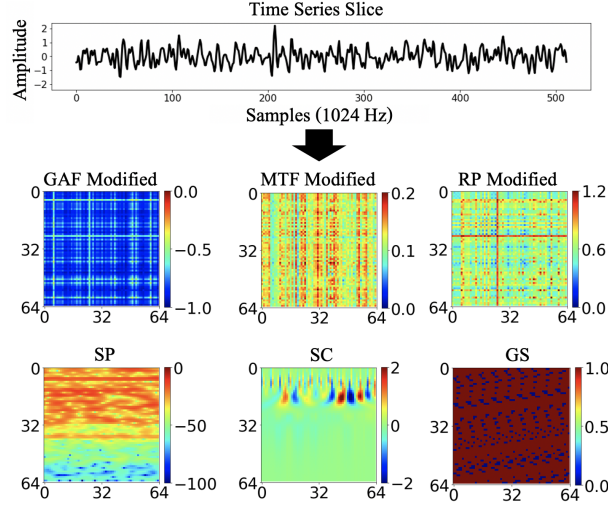
Figure 2: **Encoding Examples**.

tests of heavily instrumented aircraft (helicopters or airplanes alike) is the validation of the generated data because of the number of signals to validate. Manual validation requires too much time and manpower. Automation of this validation is crucial.

In this case, different accelerometers are placed at different positions of the helicopter, in different directions (longitudinal, vertical, lateral) to measure the vibration levels in all operating conditions of the helicopter. The data set consists of multiple 1D time series with a constant frequency of 1024 Hz taken from different flights, cut into 1 minute sequences. Labels (normal, anomalous) are associated per flight and attributed to each sequence of the flight.

The data set is split into two parts:

- the training data set: it consists only of 1677 sequences from normal flights;

- the validation data set: it consists of 594 sequences from normal and anomalous flights.

The challenge here is to train the algorithms on the healthy condition monitoring data from the training dataset that are able to reliably detect anomalies in the validation data set (without any access to the anomalous time series during training and without any prior knowledge on the number and types of possible anomalies).

4.2 Pre-processing and Hyperrameter Settings for the Time-series to Image Encodings

To evaluate the benefits of using the various time-series to image encodings, we compare the 6 selected encodings and their extensions between each other but also to a baseline that consists of raw 1D time series inputs without any encoding. For the 1D input, a 1D-CNN is applied instead of the 2D-CNN that is applied for all the time-series to image encodings. For consistency with the framework designed in Figure 1, we split the time series of length 61 440 into 120 slices of size l_1 equal to 512. No additional operations such as smoothing or normalization is performed.

For all proposed encodings, we generate one image for each of the 120 sub-series of size 512. For computational efficiency, we set all parameters, when possible, to obtain images of size 64×64. The GAF MTF, and RP encodings have by definition a resulting image size equal to that of the input time series. Therefore, we downscale the resulting images using average pooling.

In the following, the hyperparameter settings for the selected encodings are described.

GAF: The GAF only has a single parameter to tune, which is the clipping parameter that determines the allowable upper and lower boundaries for time series values. Following the work in Beirlant et al. 2006, we find a linear relationship in the QQ-plot of the training dataset. By extrapolation, we find that the probability to observe a value beyond 1.2 times the max or the min of X is numerically equal to 0. We choose therefore these values as the upper and lower bounds UB and LB. Values exceeding this threshold in the test set are clipped.

Table 1: Architecture of the two networks: 1D CAE and 2D CAE. The Decoder always exactly mirrors the encoder.

Section	Layer	Kernel Size	Stride	Shape
1D-CAE Architecture				
Encoder	Input	-	-	1x512x1
	Conv1	16	4	1x128x64
	Pool1	2	2	1x64x64
	Conv2	8	2	1x32x128
	Pool2	2	2	1x16x128
	Conv3	4	2	1x8x256
	Pool3	2	2	1x4x256
Fully-Connected	Unfold	-	-	1 024
	Bottleneck	-	-	300
	Dense2	-	-	1 024
Decoder	Mirrors the architecture of the Encoder			
2D-CAE Architecture				
Encoder	Input	-	-	64x64x1
	Conv1	2	2	32x32x64
	Conv2	2	2	16x16x128
Fully-Connected	Unfold	-	-	32 768
	Bottleneck	-	-	300
	Dense2	-	-	32 768
Decoder	Mirrors the architecture of the Encoder			

MTF: The MTF also depends only on one parameter being the number of bins which determines the resolution of the value grid. In this case, 500 bins for SAX algorithm showed a good trade-off between capturing relevant patterns, excluding noise and computational burden.

RP: The subsection length is fixed at 1 in order to avoid loosing information and to achieve the required output dimensions.

GS: For the GS encoding, we took the sub-series of size 512 and turned them into $K = 64$ series of size $K = 64$, using a stride of $s = 7$. This leads to 505 values per sub-series in total. The remaining 7 points are discarded (3 at the beginning, 4 at the end). In addition to the original formulation, we explored three further variants: 1) $P = 255$, 2) $P = 1$, and 3)the min-max scaling only, that is $P = 1$ without rounding to the nearest integer. Since it turned out that the first and last variant were giving extremely similar results, for clarity purposes, only $P = 1$ and the min-max variants are presented in the results. For all the three variants, we set the upper and lower bounds UB and LB to 1.2 the maximum and the minimum observed over the training set, similarly to GAF.

SP: For the Spectrogram, we propose to perform the STFT on the entire time series, using a Hanning window of size 126 and a temporal stride to 8. The output is of size $64 \times 7\,680$ and is split into 120 slices of size 64×64. This approach is similar to that of splitting first the series before performing the STFT but limits the border effects.

SC: For the SC, the discrete wavelet transform with a Ricker wavelet was utilized. The 64 different scales of the window were obtained by the rule illustrated in equation 12

$$scale_j = 2^{\frac{j}{4}}; j \in [1, 64] \tag{12}$$

The SC output of size 64×512 was then downscaled with factor 8 along the horizontal axis using average pooling to achieve the required out dimension of 64x64. Based on visual inspection, these parameters have shown to capture the general time series patterns sufficiently well.

Fig. 2 illustrates, for the same time series, the different images obtained after each encoding.

Table 2: For the different models, we report the true and false positive rates (TPR and FPR), the F1 score, the area under the curve (AUC) and the time required to encode 10 000 slices.

	Performance Measures				
Encodings	TPR	FPR	F1	AUC	Time[s]
No Encoding (1D)	0.61	**0.00**	0.76	0.80	-
GAF Original	0.08	0.10	0.14	0.49	24
GAF Modified	0.63	**0.00**	0.77	0.81	27
MTF Original	0.05	0.03	0.09	0.51	34
MTF Modified	0.74	**0.00**	0.85	0.87	264
RP Original	0.07	0.04	0.13	0.52	34
RP Modified	0.55	0.03	0.76	0.70	36
SP	**0.91**	0.41	0.78	0.75	4
SC	0.85	0.01	**0.91**	**0.92**	62
GS Original	0.11	0.06	0.18	0.52	1
GS - P=1	0.80	0.04	0.87	0.89	1
GS MinMax	0.65	**0.00**	0.79	0.82	1

4.3 Hyperparameters of the DL algorithms

For both the 1D and 2D CAE, we use the Adam optimizer with learning rate 0.001, β_1 equal to 0.9 and mean squared error as the loss function. Additionally, both networks are trained for 10 epochs with batch size 200. For all convolutional and dense layers, a Leaky ReLU activation function is used with alpha equal to 0.3. The hyperparameters for the 1D network are obtained by following a grid search approach, while the hyperparameters for the 2D network are manually selected such that good reconstruction measures are observed for all types of encodings. Detailed information about the network architectures is given in Table 1.

4.4 Results

We compare the different models based on the True and False Positive Rates (TPR, FPR), the F1-score, the Area Under the receiving operator Curve (AUC) and the time required to encode 10 000 images. The performance metrics of all the selected setups are presented in Table 2.

The F1-score is defined as:

$$F1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}, \tag{13}$$

The AUC is the area under the Receiving Operator Curve (ROC) for the six encodings and the proposed extensions are shown in Figure 3.

From the analysis of Figure 3 and Table 2, we can conclude that for the case study dataset and under the proposed setup, the scalogram is the best performing approach. It achieves the highest scores for the F1 and the AUC metrics.

Furthermore, all proposed modifications for the encodings GAF, MTF, RP and GS improve the performance of the anomaly detection algorithms compared to their original formulations. Most modifications consisted in making the transformations dependent of the whole training set, instead of its sliced input only as originally proposed. This allows to better capture the characteristic behaviour of the main class. For the GS encoding, finding the right P value can have a high impact on the results. Finding this parameter *a priori* is a difficult question. Additionally, not using the conversion to integers also outperforms the baseline. The choice of these parameters is, therefore, sensible in a purely unsupervised task.

Based on the F1-metric comparison, all encodings in their modified versions outperform the baseline. Note that the F1 metric is computed based on the proposed threshold as defined above. The Spectrogram and the Recurrence Plots have a lower AUC than the baseline, which means that with a different setting of the threshold, their advantage is not certain. Yet, setting this threshold optimally in an unsupervised detection task is an open research question.

At the dataset level, all approaches achieve very low FPR, highlighting that the main class is well learned. Since the networks were designed and trained based on healthy data only, this finding validates their ability to characterise the healthy conditions. The main differences of the different time-series to image encodings are mostly based on their ability to better discriminate the different types of anomalies. The difference in the performance may be due to the fact that some encodings are better suited for some types of anomalies.
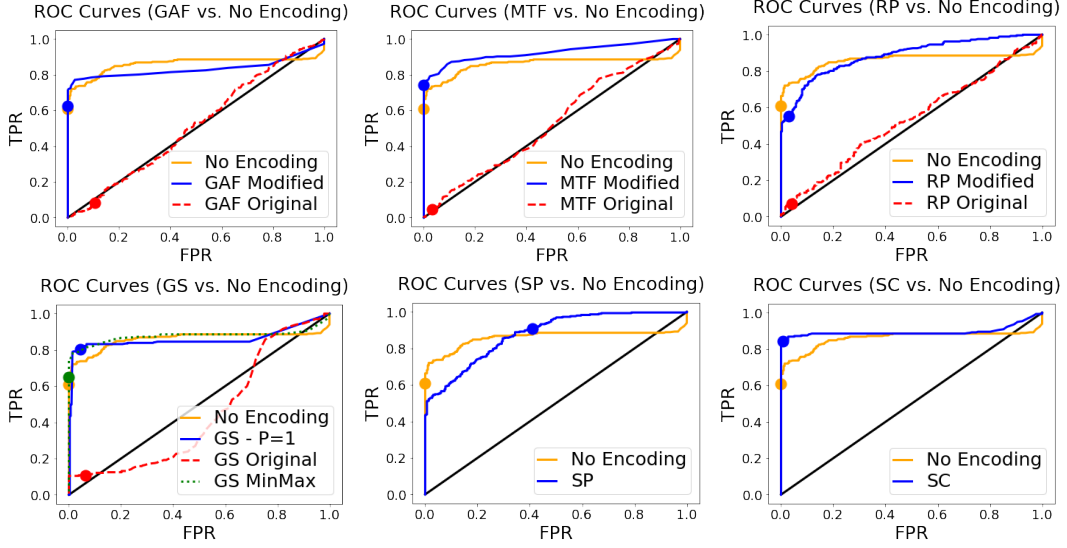
Figure 3: **ROC curves**. This figure illustrates the roc curves for the 2D-model subjected to all individual encodings as well as for the 1D-model. The black diagonal line represents the random number generator baseline.

While time-series to image encodings provide benefits in terms of performance and partly also in terms of interpretability, using them comes at the expense of an increased computational complexity and required memory. Especially the SC and the MTF, which coincidentally are among the best performing encodings, require a considerable amount of time to encode large numbers of time series. For the MTF, this shortcoming only holds for the bin assignment process which is part of the original formulation. The proposed extension with the SAX-Algorithm ovecomes this limitation.

5 Discussion

The results presented above demonstrate the validity of the approach since all encodings bring an improvement compared to the baseline of using directly the one dimensional raw time series data as input to the DL algorithm. The results also open interesting points for discussion and perspectives for further developments and further evaluations.

Defining the transformation: Using the original formulations of the different encodings (Faouzi and Janati 2020) in our case study, led to a poor performance observed in Table 2. Defining the transformation parameters per input tends to make the encoded images more uniform. While it might be beneficial in a supervised setup to learn the very specific class discriminative features and differences in the images, it is prejudicial in an unsupervised setup, based on the learning of the characteristics of a single (healthy) class. Taking the whole dataset into account when performing the transformation was an essential step in the performance of the proposed encodings.

Threshold choice: Learning the right threshold using healthy data only is always a difficult task. In our work, the choice of the 99-th percentile on the residuals seems to provide reasonable results, which are in line with the threshold independent metric AUC. Yet, we highlight the fact that this threshold has been learnt on normal data only. Therefore, it is difficult to state that this choice will be effective for any other problem, any other type of data and anomalies.

Selection of time-series to image encoding: Even though all image encodings provide reasonably good results, it is not yet easy to say beforehand which encoding will be the best for a particular use case. It is quite obvious that each encoding will be more adapted to certain types of time series and will perform differently on various types of anomalies.

Architecture of the CAE: We used a rather simple Convolutional Neural Network since the main goal was the comparison between the different encodings and the same architecture was applied to all the encodings. It could be interesting to explore other architectures to study the impact of the architectures on the performance.

Aggregation of residuals: The computation of the encodings can be quite costly when the number of time steps increases. Computing performance on the same number of time series with the same hardware are given in Table 2 for information. This scalability of the image encodings is one of the reasons why we cut the initial time series into smaller ones. Since the other reason was the ability to detect local anomaly, we used here the maximum residuals on

all slices and compared them to the threshold. Other choices could have been made (mean, quantile over the slices, etc.) and they would have impacted the results. Exploring further the choice of the decision criteria could be the subject for future investigations. Since the main goal of this study was to compare the different encodings, the same decision was applied to all the encodings.

Interpretability of image encodings. An additional advantage of using time-series to image encodings, not addressed in this paper, is their improved and more intuitive interpretability compared to pure time series data. Domain experts may have more difficulties to identify, interpret and track patterns and anomalies in an intuitive way on different time scales in time series data. Some domains already use images to analyze their signals, in particular spectrograms and scalograms. Using other encodings could also become quite natural, given that the experts train themselves in their interpretation to recognize different conditions from the generated images.

6 Conclusion and perspectives

In this paper, we evaluated the application of several time-series to image encodings in an unsupervised anomaly detection setting. Further to applying the originally proposed encodings, we also proposed modifications to the GAF, MTF and GS encodings to make them more suitable within the proposed framework for anomaly detection. Testing all approaches, including the use of the raw time series directly, in similar conditions and on the exact same data, we demonstrated that all the (modified) image encodings led to an improvement compared to using the raw time series. The results are aligned with the intuition that, converting the time series into two dimensional representations allows to capture more complex patterns between measures, such as correlations, recursive behaviors or spectral components. These relationships can then improve the ability to characterise healthy data and to separate anomalies. These results complement, therefore, the previous studies realised in supervised setups that already hinted the benefits of these approaches. In addition, the conversion of the time series to images can provide a tool for domain experts to analyse and interpret the measurements in more intuitive and more efficient way. By making the relevant patterns more easily identifiable, experts would have a simpler way to perform subsequent diagnostics and distinguishing between the different fault types based on the different patterns in the encodings.

In the present paper, the framework was tested on a specific type of data, vibrations data from in-flight measurements. Future research is required to evaluate the the benefits in other contexts, with other types of data and with other anomalies. Ideally, an analysis exploring which encoding would suit best for which type of data and which type of anomaly would be extremely beneficial to the community. Based on such analysis, the encodings could be combined in an ensemble learning framework to perform both detection and diagnostics.

References

Airbus SAS (2018). *Airbus Helicopter Accelerometer Dataset*. DOI: 10.3929/ethz-b-000415151.

Beirlant, Jan et al. (2006). *Statistics of extremes: theory and applications*. John Wiley & Sons.

Boashash, Boualem (2015). *Time-frequency signal analysis and processing: a comprehensive reference*. Academic Press.

Chouhan, Naveed, Asifullah Khan, and Haroon ur Rasheed Khan (2019). "Network anomaly detection using channel boosted and residual learning based deep convolutional neural network". In: *Applied Soft Computing* 83, p. 105612. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105612. URL: http://www.sciencedirect.com/science/article/pii/S1568494619303928.

Faouzi, Johann and Hicham Janati (2020). "pyts: A Python Package for Time Series Classification". In: *Journal of Machine Learning Research* 21.46, pp. 1–6. URL: http://jmlr.org/papers/v21/19-763.html.

Fawaz, Hassan Ismail et al. (2019). "Deep learning for time series classification: a review". In: *Data Mining and Knowledge Discovery* 33.4, pp. 917–963.

Gondara, Lovedeep (2016). "Medical image denoising using convolutional denoising autoencoders". In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, pp. 241–246.

Hu, Yang, Thomas Palm, and Olga Fink (2017). "Fault detection based on signal reconstruction with Auto-Associative Extreme Learning Machines". In: *Engineering Applications of Artificial Intelligence* 57, pp. 105 –117. ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2016.10.010. URL: http://www.sciencedirect.com/science/article/pii/S0952197616301853.

Kamphorst, J-P Eckmann S Oliffson, D Ruelle, et al. (1987). "Recurrence Plots of Dynamical Systems". In: *Europhysics Letters* 4.9, p. 17.

Krummenacher, Gabriel et al. (2017). "Wheel defect detection with machine learning". In: *IEEE Transactions on Intelligent Transportation Systems* 19.4, pp. 1176–1187.

Lin, Jessica et al. (2007). "Experiencing SAX: A Novel Symbolic Representation of Time Series". In: *Data Min. Knowl. Discov.* 15, pp. 107–144. DOI: 10.1007/s10618-007-0064-z.

Marwan, Norbert et al. (2007). "Recurrence plots for the analysis of complex systems". In: *Physics Reports* 438.5, pp. 237 –329. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2006.11.001. URL: http://www.sciencedirect.com/science/article/pii/S0370157306004066.

Michau, Gabriel et al. (2020). "Feature learning for fault detection in high-dimensional condition monitoring signals". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 234.1, pp. 104–115.

Qu, Yongzhi et al. (2019). "Gear pitting fault diagnosis using disentangled features from unsupervised deep learning". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 233.5, pp. 719–730.

Sasirekha, K and K Thangavel (2020). "A Novel Biometric Image Enhancement Approach With the Hybridization of Undecimated Wavelet Transform and Deep Autoencoder". In: *Handbook of Research on Machine and Deep Learning Applications for Cyber Security*. IGI Global, pp. 245–269.

Souza, Vinicius MA, Diego F Silva, and Gustavo EAPA Batista (2014). "Extracting texture features for time series classification". In: *2014 22nd International Conference on Pattern Recognition*. IEEE, pp. 1425–1430.

Valle, E. D. et al. (2009). "It's a Streaming World! Reasoning upon Rapidly Changing Information". In: *IEEE Intelligent Systems* 24.6, pp. 83–89.

Verstraete, David et al. (2017). "Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings". In: *Shock and Vibration* 2017.

Wang, Zhiguang and Tim Oates (2015). "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks". In: *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Wen, L. et al. (2018). "A New Convolutional Neural Network-Based Data-Driven Fault Diagnosis Method". In: *IEEE Transactions on Industrial Electronics* 65.7, pp. 5990–5998.

Xu, Gaowei et al. (2019). "Online fault diagnosis method based on transfer convolutional neural networks". In: *IEEE Transactions on Instrumentation and Measurement*.

Zhang, Changqing, Yeqing Liu, and Huazhu Fu (2019). "Ae2-nets: Autoencoder in autoencoder networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2577–2585.

Zimek, Arthur and Erich Schubert (2017). "Outlier Detection". In: *Encyclopedia of Database Systems*. Ed. by Ling Liu and M. Tamer Özsu. New York, NY: Springer New York, pp. 1–5. ISBN: 978-1-4899-7993-3. DOI: 10.1007/978-1-4899-7993-3_80719-1. URL: https://doi.org/10.1007/978-1-4899-7993-3_80719-1.