

Deep semi-supervised generative adversarial fault diagnostics of rolling element bearings

Structural Health Monitoring

2020, Vol. 19(2) 390–411

© The Author(s) 2019

Article reuse guidelines:

sagepub.com/journals-permissions

DOI: 10.1177/1475921719850576

journals.sagepub.com/home/shm



David Benjamin Verstraete¹, Enrique López Droguett^{1,2} ,
Viviana Meruane² , Mohammad Modarres¹ and Andrés Ferrada³ 

Abstract

With the availability of cheaper multisensor suites, one has access to massive and multidimensional datasets that can and should be used for fault diagnosis. However, from a time, resource, engineering, and computational perspective, it is often cost prohibitive to label all the data streaming into a database in the context of big machinery data, that is, massive multidimensional data. Therefore, this article proposes both a fully unsupervised and a semi-supervised deep learning enabled generative adversarial network-based methodology for fault diagnostics. Two public datasets of vibration data from rolling element bearings are used to evaluate the performance of the proposed methodology for fault diagnostics. The results indicate that the proposed methodology is a promising approach for both unsupervised and semi-supervised fault diagnostics.

Keywords

Generative adversarial networks, fault diagnostics, deep learning, health monitoring, ball bearings, vibration analysis

Introduction

Condition health monitoring systems are becoming a standard specification for customers purchasing large capital assets. With the proliferation of cheap sensing technology, these assets are now streaming massive quantities of data at an unprecedented rate. The fields of structural health monitoring (SHM) and fault diagnostics have grown from the need to make sense of these data. The primary drawback to fault diagnostics within these systems is the requirement of labeling millions, and potentially billions, of data points. To label a dataset of this magnitude is resource intensive, costly, computationally expensive, and subject to confirmational data biases of the engineers interpreting the data. Thus, labeling the output of an extensive sensor system data output requires significant investment. Moreover, there is a strong assumption within supervised fault diagnosis that everything is known about a preset class of faults. This restricts the ability of the supervised model to generalize. If the model only knows what the engineer knows, it is reasonable to assume the model's knowledge of the system could be incomplete; therefore, traditional feature learning would have a fundamental generalization problem.

The general problem within unsupervised learning is extracting information or value from unlabeled data. Unsupervised learning is an ill-posed problem because appropriate downstream tasks are unknown at the time of training. Therefore, unsupervised learning should disentangle the relevant unknown tasks that are helpful for the problem. For instance, a useful disentangled representation for a dataset of cracks in a concrete structure would be dimensions for crack length, crack width, neighboring cracks, or the presence of the crack intersections.¹ These representations may be relevant for natural tasks like damage evaluation or crack propagation. For irrelevant tasks, like the percentage of white pixels, this representation would be extraneous. Therefore, a useful unsupervised learning algorithm

¹Department of Mechanical Engineering, University of Maryland, College Park, MD, USA

²Department of Mechanical Engineering, University of Chile, Santiago, Chile

³Department of Computer Science, University of Chile, Santiago, Chile

Corresponding author:

Enrique López Droguett, Department of Mechanical Engineering, University of Chile, Av. Beauchef 851 Santiago, Chile.

Email: elopezdroguett@ing.uchile.cl

must guess the likely set of subsequent classification tasks correctly without knowledge of what the tasks are. This is a challenge deep learning attempts to solve. Deep learning makes up much of the recent unsupervised fault diagnostic research.^{2–10} All these approaches, except that of Langone et al.,³ are restricted to unsupervised feature learning followed by supervised fault diagnostics. Moreover, none of these methods attempt unsupervised learning with an image representation of the data.

Most recently, generative adversarial networks (GANs) have been developed within the computer vision community.¹¹ Training this deep generative modeling is done using a minimax game. The goal of training is to learn a generator distribution that fools the discriminator into classifying it as from the true data distribution. Unlike variational autoencoders (VAEs), which tries to assign probability to every data point in the data distribution,¹² a GAN learns a generator network, which transforms a noise variable in a sample by generating samples from the sample distribution. With a sharper image from GANs, one gets more precise image features. However, currently there are no agreed upon methods to assess the training of, or comparison of, a GAN without visually inspecting the images. This is difficult to accomplish without an image of the signal. A vector of data would not suffice. Therefore, GANs provide a better foundation for fault diagnostics based on rich images of signals.

In this article, we propose a novel deep learning generative adversarial methodology for a comprehensive approach to fault diagnostics on time–frequency images. This article explores both deep convolutional GANs (DCGAN) and information maximizing GAN (InfoGAN) architectures. From the proposed architectures for these two types of GANs networks, clustering is done via spectral and k-means++ clustering on the down-sampled activation output of the discriminator. To improve clustering results, semi-supervised learning is included as a second stage to the methodology by altering the cost function to account for data labels. In addition, both 32×32 pixel and 96×96 pixel images are explored as inputs to methodology. This methodology is then evaluated with both the Machinery Failure Prevention Technology (MFPT) Society¹³ and Case Western Reserve (CWR) University Bearing Data Center¹⁴ bearing datasets. The proposed methodology's results are then compared with unsupervised learning via autoencoders (AEs) and VAE. To evaluate the proposed unsupervised methodology, traditional supervised learning metrics are inappropriate. A confusion matrix and its associated metrics are unable to evaluate clustering techniques. The ground truth is known; therefore, purity,¹⁵ normalized mutual information

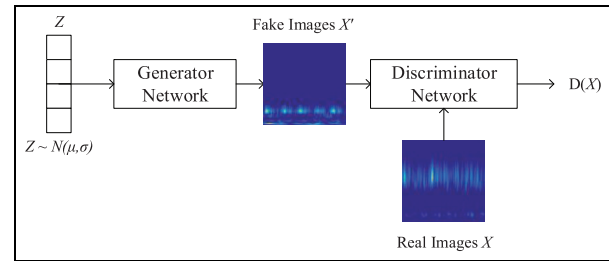


Figure 1. GAN overview.

(NMI),¹⁶ and adjusted rand index (ARI)¹⁷ are used to evaluate the quality of the clusters.

The rest of this article is organized as follows: section “Background on adversarial training” provides an overview of GANs. Section “Proposed generative adversarial fault diagnostic methodology” outlines the proposed unsupervised and semi-supervised methodology constructed to aid the diagnostic task of fault detection. Section “Examples of applicationu applies the methodology to both the MFPT and CWR experimental datasets. Section “Comparison with AE and VAEo compares these results to unsupervised AE and VAE. The last section finishes with some concluding remarks.

Background on adversarial training

GANs were first proposed by Goodfellow et al.¹¹ GANs consist of a generator network and a discriminator model network. Generative models seek to learn the underlying joint probability distribution $P(x, y)$ of the random variables to categorize a signal. Discriminative models, on the other hand, disregard how the data were generated and simply categorize the data points based on a conditional probability distribution $p(y|x)$.¹⁸ Within the context of fault diagnostics, generative models attempt to learn every potential fault to then classify the faults, whereas discriminative models attempt to determine fault differences absent of learning every fault. GANs seek to utilize both models’ strengths. The generator attempts to create a synthetic dataset, X' , which matches the real data, X that only the discriminator can see and classify as shown in Figure 1. The generator samples from a noise distribution, Z (e.g. normal) and the discriminator determines whether the sampled data (e.g. an image) is real or fake.

Functionally, GANs train two convolutional neural networks (CNNs) at the same time. The generator, which is depicted in Figure 1, utilizes deconvolutional layers to take the noisy input Z and creates the specified size image. The parameters in Z is then updated continuously throughout the training of the network. This image is then fed into the discriminator network

to judge whether the generated image is real or fake. The discriminator is a CNN with convolutional layers, pooling, and nonlinear activations. This is all done in a feedforward operation where the weights, biases, and errors are set throughout. To update the networks and adjust these hyperparameters, backpropagation is used to send the errors back through the networks to update the weights and biases. This process removes redundant, uninteresting features.

To accomplish this, the fundamental foundation of the GANs algorithm is the two-player minimax game. The generative network maps a noise source to an input space to generate a fake image. The discriminative network receives the generator's input (a fake image) and classifies it as real or fake. This amounts to a two-player game with the two networks competing against each other as in equation (1)¹¹

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim P_{data}(x)} [\log(D(x))] + \mathbb{E}_{z \sim P_{noise}(z)} [\log(1 - D(G(z)))] \quad (1)$$

where $P_{data}(x)$ is data distribution, $P_{noise}(z)$ is noise distribution, $D(x)$ is discriminator objective function, and $G(z)$ is the generator objective function.

For each generator parameter update the discriminator is trained to optimality. The minimization of the value function leads to the minimization of the Jensen-Shannon (JS) divergence between the real data and the trained model distributions on x . This minimization frequently results in vanishing gradients as the discriminator saturates. While the ideal training results in optimality, in most practical applications this is not necessarily the case. At the moment the training of GANs requires visual inspection of the output images; therefore, an image representation of the signal is needed.

There has been, and there continues to be, a large amount of research surrounding the architectures and training of a GAN. For this article both DCGAN and InfoGAN are used. Fundamentally, these two GANs are identically trained to the proposed method by Goodfellow et al.¹¹ However, their architectures and cost functions are modified to account for the applied datasets and unsupervised versus semi-supervised objective functions.

Clustering

This article examines two primary clustering algorithms for classification (k-means++ and spectral) and principal components analysis (PCA) for visualization. K-means++ was chosen to explore the robustness of the methodology to simple clustering algorithms. K-means++ differs from the traditional k-means algorithm by

first choosing the initial cluster center uniformly at random and then choosing each subsequent center with probability proportional to the square of its proximity to the nearest center.¹⁹

Algorithm 1. K-means++ algorithm.

Initialize k-means++ algorithm

- Take one center, c_1 , chosen uniformly at random from data points, X .
 - Take a new center, c_i , choosing $x \in X$ with probability $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$, where $D(x)$ denotes the shortest distance from a data point to the closest center already chosen.
 - Repeat previous step until all k centers are taken.
 - Proceed with standard k-means algorithm.
-

Spectral clustering on the other hand is a graph clustering technique where eigenvectors of the data matrices are used. Data are mapped to a low-dimensional space for spectral clustering. This dimensionality reduction is more computationally expensive than the k-means++ algorithm; however, it can achieve superior results.²⁰

Proposed generative adversarial fault diagnostic methodology

A two-stage fault diagnostic methodology is proposed within this article. Stage one consists of fully unsupervised generative adversarial fault diagnostics, and stage two semi-supervised generative adversarial fault diagnostics. In practice, sensor signals are gathered, and stage one can be used at the start to assess the baseline of the system when labels for the data are unavailable. As knowledge of the system signals improve, fault signals can be identified, labeled, and then incorporated into stage two. The intention is that unsupervised clustering, operating on the automatic identified features by the GAN, identifies fault clusters away from the baseline. Once labeled data is available, it can be added to the model to improve the maintenance decision-making. Upon completion, the engineer can then visually monitor the system via PCA to begin labeling some of the signals being gathered. This labeled data can then be input into the GANs methodology, with a modification to the cost function, to further improve the clustering results until a predefined criterion of performance is met. Once the system signals move to a fully supervised labeled dataset, the engineer can then transition the modeling to a fully supervised deep learning framework.²¹

Algorithm 2. Spectral clustering algorithm.

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L .
- Compute the first k eigenvectors v_1, \dots, v_k of the generalized eigenproblem $Lv = \lambda Dv$.
- Let $V \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors v_1, \dots, v_k as columns.
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i -th row of V .
- Cluster the points $(y_i)_{i=1, \dots, n}$ in \mathbb{R}^k with the k -means algorithm into clusters C_1, \dots, C_k .

Output: Clusters A_1, \dots, A_k with $A_i = \{j | y_j \in C_i\}$.

The discriminator network provides the ability to train itself against generated images as an adversary within both DCGAN and InfoGAN architectures. Since the discriminator is trained to predict the fake from the real dataset, it can provide a robust feature set of the real data. To accomplish this, the GAN discriminator training automatically generates a high-level feature representation of the data from the input image to an output vector. The goal of the GANs training is then to take this high-level representation feature set as an input to clustering algorithms. This allows the generator to avoid overfitting on the raw data by only having access to the gradients. Two GAN architectures explored in this article are not a restriction on the methodology; these were two architectures chosen because of their strong results in other tasks, such as image generation. To use the InfoGAN, the encoder dimension must be given as the number of system health states believed to exist, whereas this is not a requirement for training the DCGAN. For instance, to validate the proposed methodology the encoder dimension was set to three. The DCGAN training, on the other hand, does not require the encoder dimension.

The DCGAN architecture developed for this article incorporates the guidelines proposed in Radford et al.,²² however, some adjustments were made to the architecture to handle the datasets used in this article and thus provide superior results for unsupervised fault diagnostics. DCGANs were the first major advancement on the original GAN architecture.²² Through exhaustive model exploration, this work resulted in the following five GANs architecture guidelines:

1. Pooling layer replacement with strided convolutions for both the discriminator and the generator networks;
2. Batch normalization (BN) is required for both the discriminator and generator networks;
3. Fully connected (FC) hidden layers should be removed for deep architectures;
4. Rectified linear unit (ReLU) activation use in all layers of the generator except the output should use tanh;
5. Leaky ReLU activation use on all layers for the discriminator.

DCGANs are the main baseline to implement GANs; however, as stated in Radford et al.,²² model instability still exists within the training of the model. The longer the model trains, the higher the risk of mode collapse. This occurs when a filter subset collapses to a single oscillating mode.

There have been many studies on the effectiveness of individual wavelets and their ability to match a signal. One could choose between the Gaussian, Morlet, Shannon, Meyer, Laplace, Hermit, or the Mexican Hat wavelets in both simple and complex functions. To date, there is no defined methodology for identifying the proper wavelet to be used and this remains an open question within the research community.²³ For the purposes of this article, the Morlet wavelet is chosen because of its similarity to the impulse component of symptomatic faults of many mechanical systems.²⁴

As shown in Figure 2, the proposed methodology starts with the training of a GAN with the unlabeled dataset. This will train two CNNs, one discriminator, and one generator. The discriminator needs to learn distribution of the real vibration fault dataset to be able to discriminate between the generated fake samples and the real samples. The generator attempts to trick the discriminator by learning the underlying distribution of the generated data. The last activation layer of the training is then concatenated and visually inspected via PCA to evaluate the ability of the GAN to separate the data. At this point, the engineer will be looking for a robust representation of the baseline signals from the asset. From there, the engineer weighs the value of labeling the incoming data versus the cost to label.

In the following sections, we discuss and detail the proposed methodology for fault diagnostics. The

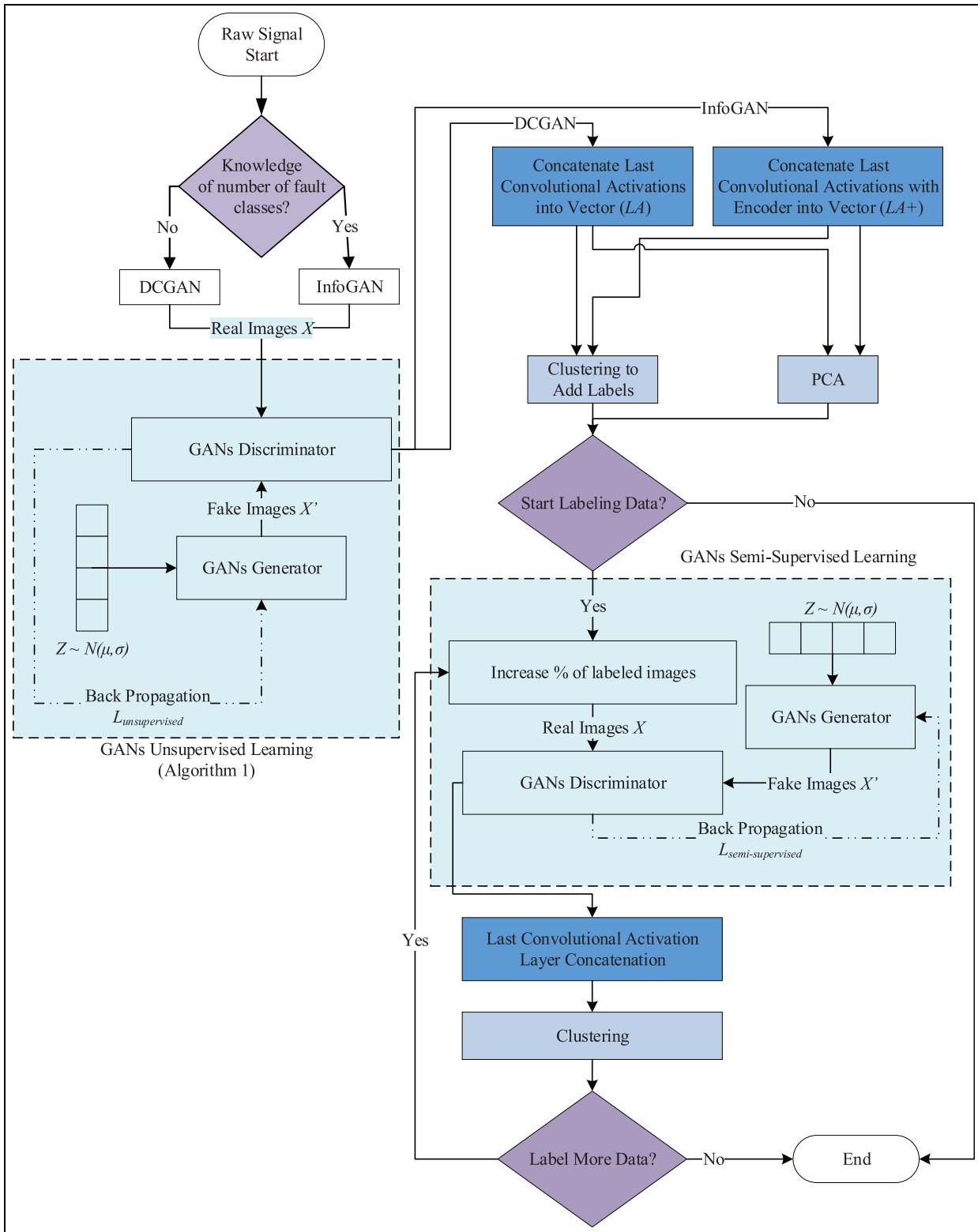


Figure 2. Proposed generative adversarial fault diagnostic methodology.

following sections include discussions regarding the architectures for the DCGAN and InfoGAN models

underpinning the methodology followed by a detailed discussion on the proposed methodology steps.

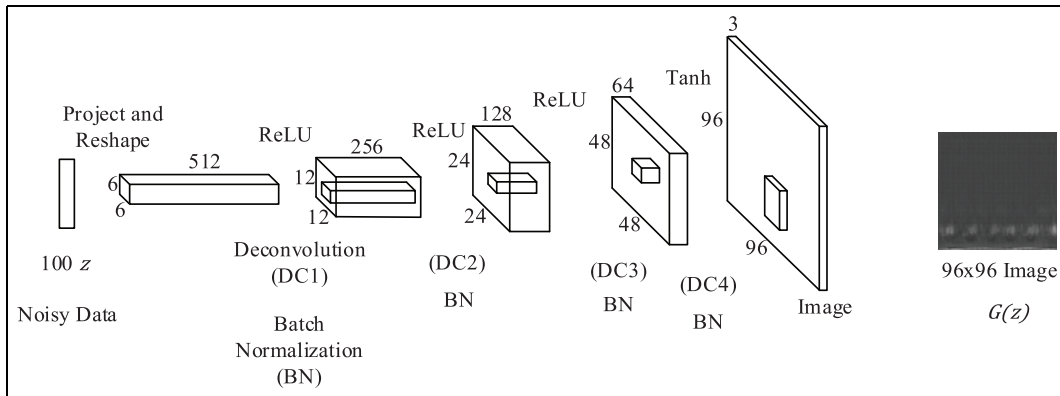


Figure 3. Generator network.

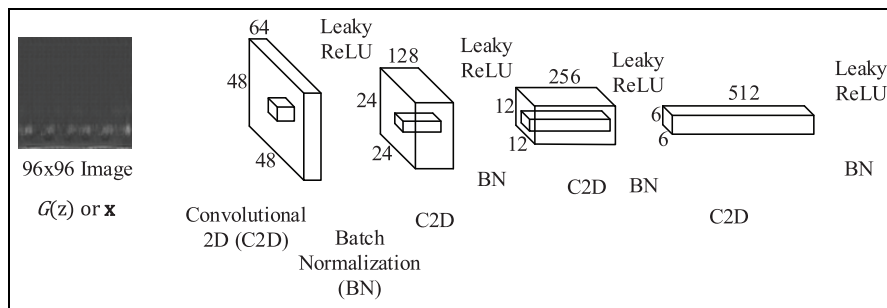


Figure 4. Discriminator network.

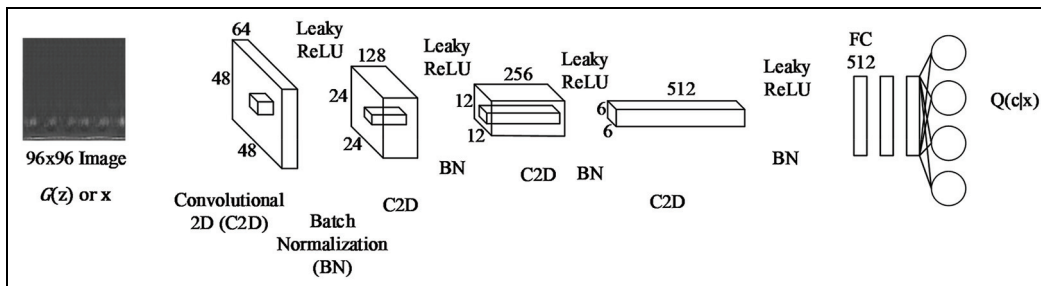


Figure 5. InfoGANs discriminator network.

Functionally, the training of the DCGAN involves tuning the parameters on two CNNs. In the generator architecture in Figure 3, noise Z is used to generate a vector of data. This is then used to project and reshape to $512 \ 6 \times 6$ features. From these features, $256 \ 12 \times 12$ features are deconvolved. Following to $128 \ 24 \times 24$ features, then $64 \ 48 \times 48$ features, and finally $3 \ 96 \times 96$ images are generated. Between each of these layers, BN and ReLU are used, and finally tanh is used for the last layer.

The discriminator CNN in Figure 4 shows the reduction from the image into smaller features. The

discriminator takes the 96×96 image and convolutes the image into $64 \ 48 \times 48$ size. The 64 feature maps are then again convoluted to 128 feature maps of size 24×24 , then 256 feature maps of 12×12 size, and finally 512 feature maps of 6×6 size. Between each of these layers, the data are passed through BN and leaky ReLU. The final activation layer of $512 \ 6 \times 6$ feature maps results in features automatically learned from the data and maps to the subsequent step in the proposed methodology discussed in section “Proposed Generative Adversarial Fault Diagnostic Methodology.”

This comprises both networks for the DCGANs training. To update both networks through each step, a cross-entropy backpropagation is used. This backpropagation allows updating of the weights and biases throughout the network to optimize toward the intended outputs. This is done with the gradients out of the discriminator to help avoid overfitting on the raw data.

InfoGANs take the unsupervised objective function into account as a mutual information variable in the input of the generator network.²⁵ This input now consists of z and c vectors. The latter is used in the mutual information term to represent some latent variable in the data. The InfoGAN's objective remains the same as the GAN objective function; however, it now makes use of the dataset descriptive latent variables c and z , as shown in equation (2)

$$\min_G \max_D V_{\text{InfoGAN}}(G, D, Q) = V(G, D) - \lambda L_I(c; G(z, c)) \quad (2)$$

where Q is the auxiliary distribution to approximate the posterior, G is generator, D is discriminator, c is the latent code, z is incompressible noise, $G(z, c)$ is generator network in terms z and c , and L_I is variation lower bound of mutual information I .

The hyperparameter λ is introduced within the InfoGAN optimization to control the scale of the GANs objective function. A λ set to 1 suffices for discrete latent codes, and a smaller λ is useful for continuous variables to ensure the scale remains the same.

Figure 5 shows the proposed architecture for the InfoGAN discriminator. Note that this discriminator network, more specifically the FC encoder layer within InfoGAN, is the only difference versus DCGANs. The distribution $Q(c|x)$ is the posterior approximation of the true posterior $P(c|x)$. The approximate posterior Q is parameterized as a neural network.

The benefit to using the InfoGAN algorithm is the ability to extract meaningful features of the data created by the generator encoder vector. This means, for a fault diagnostic problem, c encodes the semantic features (fault classes, e.g. baseline, inner race fault, outer race fault) of the data distribution and z encodes the unstructured noise of the distribution (e.g. width of the impulse, background noise of the signal). Even with the mutual information objective function, there is no guarantee that the latent variables found by the trained InfoGAN will be the desired structure in the data. Therefore, InfoGANs still require visual inspection of the generator images to assess its image quality. In the following sections, we discuss the steps in the proposed methodology.

Read raw signal and image representation construction

Prior to GAN initialization, it is necessary to generate images of the accelerometer data streaming from the rolling element bearing. Scalogram images contain time and frequency on the axis and the color depicts the magnitude. Once the images are generated, the entire dataset is subdivided into three groups: training, test, and validation. It is common to have the bulk of the images in the training set, with the remainder used as a test set to evaluate the model's ability to predict the system's health classes.

Unsupervised GAN initialization

Once the scalogram images are generated, Algorithm 3 outlines the process as a means for a feedforward pipeline for fault diagnosis. Global average pooling is used to reduce the last convolutional layer filters to k vectors of 1×1 . This is then concatenated to form a $1 \times k$ vector and fed into a clustering algorithm.

Algorithm 3. Unsupervised feedforward pipeline for images, i .

Train GAN Architecture to data dependent, context dependent epoch count.

for i images **do**

- Feed forward pass through the discriminator.
- Global Average Pooling for k filters out of last convolutional layer to output $k \times 1$ filters.
- Concatenate last convolutional layer activations (with encoder for the InfoGAN) from each image i as a $1 \times k$ vector.
- Normalize this vector with L2 Norm (Euclidean Distance):

$$|\mathbf{x}| = \sqrt{\sum_{k=1}^n |x_k|^2}$$

end for

- Vector is labeled LA vector for DCGAN.
 - Vector is labeled $LA + \text{encoder}$ vector for InfoGAN.
 - The resulting vector is fed into a clustering algorithm (k-means ++, spectral) to obtain labels for images.
-

Once the GAN model is trained (DCGAN or InfoGAN), two additional steps are needed to evaluate the model. The first step is a visual inspection of the trained generator network to evaluate the quality of the generated images. Visual inspection of the output images of the generator network is a key indicator to how well the GAN architecture is training and whether any of the known drawbacks are surfacing, such as mode collapse,²⁶ vanishing gradients,²⁷ non-

convergence,²⁸ and checkerboarding artifacts.²⁹ The second step consists of sampling images from a random uniform input vector between 0 and 1. For the InfoGAN, the c input vector is a random one-hot encoded categorical vector. This step uncovers problems in the convergence of the network, mode collapse to a specific kind of image, or the inability of the model to generate similar images to the ones in the original dataset.

Concatenation, normalization, and clustering

To extract the discriminator information after training, a feedforward pass is done with each image (i) in the dataset to obtain each last convolutional layer activation. These activations are pooled via global average pooling for each filter (k). This means that, given k filters in the last layer, the output is $k \times 1 \times 1$ vectors for each scalogram. After global average pooling, these vectors are concatenated into a $1 \times k$ vector and normalized with Euclidean L2 normalization. From this point, the vector output of the DCGAN is referred to as the last layer activations vector, or LA vector. Moreover, the output of the InfoGAN includes the encoder output. Therefore, from this point, this encoder concatenated with the LA vector output of the InfoGAN is referred to as the $LA + encoder$ vector.

The last step is to use this LA vector or $LA + encoder$ vector (C or C_{en} , respectively) as an input into clustering algorithms. For the purposes of this article, k-means++ and spectral clustering are examined. Again, this is not a restriction on the methodology; it is a means to display the robustness of the methodology to two common straightforward clustering algorithms.

Unsupervised visual evaluation—PCA

Once the output of unsupervised clustering is complete, a method to assess the clustering results without the real labels is needed. For the proposed methodology, one could evaluate the clustering output of the LA or $LA + vector$'s visually to choose the appropriate number of clusters to proceed with the remainder of the methodology. Note that the GANs training creates a suitable underlying manifold representation of the data that can be used in a two-dimensional visual inspection. Engineering knowledge can then be utilized to provide meaning to the evaluation of the visual results of PCA.

Label data

One of the strengths of the proposed methodology is the ability to feed in an incrementally increasing amount of labeled data into the training dataset of the

GANs algorithm to increase fault class identification results from the clustering. This has practical importance because, when a new asset comes online, initially there may be little knowledge of the system faults and their respective raw signals. As more knowledge is gained, labeled data can be incorporated into the model. The results section of this article validates the methodology with metrics for increasing percentages of labeled data (for validation purposes, it is assumed that labels are known) within the training dataset for semi-supervised fault diagnostics.

Semi-supervised GAN initialization

Semi-supervised GAN initialization involves training of the chosen GAN architecture with an incrementally increasing set of labeled data. This is an important aspect to explore because as the engineer gains more knowledge about a new system, one can label small sets of data which are known to be faults to increase the system's health state identification via clustering. This approach improves the quality of the clustering results via a semi-supervised cost function (equation (6)) as described in Salimans et al.²⁸ In the unsupervised training, the discriminator learns features to avoid classifying the generated data as real data, but these features might not be the best representation given the implicit labels the problem has. One way to help the discriminator get improved and more meaningful features for these labels is to use the discriminator as a classifier for these classes. This is possible with a minor change to the proposed GAN pipeline outlined in the first step of Algorithm 1. Indeed, the loss function L is modified to equation (3), as follows

$$L = L_{supervised} + L_{unsupervised} \quad (3)$$

where

$$\begin{aligned} L_{supervised} &= -\mathbb{E}_{x, y \sim p_{data}(x, y)} \log p_{model}(y|x, y < K + 1) \\ L_{unsupervised} &= -\{\mathbb{E}_{x \sim p_{data}(x)} \log[1 - p_{model}(y = K + 1|x)] \\ &\quad + \mathbb{E}_{x \sim G} \log[p_{model}(y = K + 1|x)]\} \end{aligned}$$

This cost function adds a cross entropy loss for the first k discriminator outputs. The unsupervised cost is the same as the original GAN (equation (1)). However, there is a slight change as now $K + 1$ corresponds to the probability of the sample being false. The discriminator is used as a competent classifier given a subset of the dataset. In this case, the discriminator will be used as a feature extractor given a subset of the dataset to improve the system's health state identification results based on clustering. Labels are used as clues for the structure of the data with the aim of creating an improved discriminator. This assumes that images

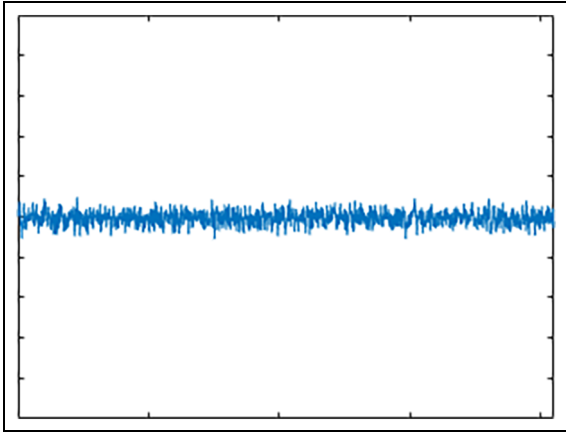


Figure 6. Baseline signal.

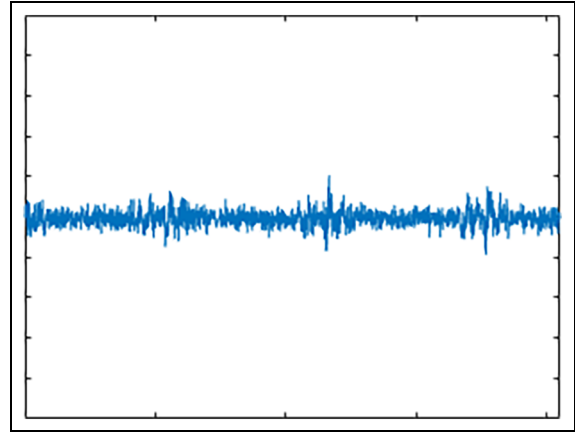


Figure 7. Outer race fault signal.

generated with semi-supervised learning have better quality than the ones generated in an unsupervised manner. However, notice that the main objective of a GAN is to generate data points or images that resemble the training dataset and not to predict any system's health states. Thus, if we use a few labeled data points and generated data we are performing a semi-supervised training.

Semi-supervised stop criteria

For a qualitative analysis, the GAN's last activation layer outputs are used to generate a two-component PCA plot. The ideal result would be a clear separation between the health states (classes). If there is not a clear separation, then adding labeled data would help aid in the separation and provide better system health state diagnosis. It is at this point the engineer, now with a small number of labels of the fault conditions, can begin using metrics to evaluate whether the model is performing suitably to cease labeling additional data. Eventually the quantity of labeled data reaches a point at which the decision can be made to explore a deep learning enabled fully supervised fault diagnostic methodology.

Examples of application

In this section, the proposed methodology is applied to both the MFPT and CWR bearing datasets. To validate the proposed methodology, known labels are available. Therefore, metrics like purity, NMI, and ARI can be used. Graphics processing unit (GPU) computing was utilized throughout this article using a system with a Nvidia GPU Titan XP, CPU Core i7-6700 K 4.2 GHz, 32 GB RAM, Tensorflow 1.0, cuDNN 5.1, and Cuda 8.0.

MFPT dataset

This dataset was provided by the MFPT Society.¹³ An experimental test rig with a NICE bearing gathered accelerometer data for three conditions. First, a baseline condition was measured at 270 lbs of load and a sampling rate of 97,656 Hz. Second, 10 total outer race-way faults were tracked. Three outer race faults were loaded with 270 lbs with a sampling rate of 97,656 Hz, and seven outer race faults were assessed at varying loads: 25, 50, 100, 150, 200, 250, and 300 lbs. The sampling rate for the faults was 48,828 Hz. Third, seven inner race faults were analyzed with varying loads of 0, 50, 100, 150, 200, 250, and 300 lbs. The sampling rate for the inner race faults was 48,848 Hz. Scalogram images, as shown in Table 1, were generated from the raw signal with the following classes: normal baseline, inner race fault, and outer race fault. The total scalogram images used for each class was 3423, 1981, and 5404 respectively. With 10,808 total images, the training set size used was 50%. Bilinear interpolation³⁰ was used to scale the images down to a manageable size for the training. The MFPT dataset is a good test for any algorithm's ability to separate the baseline healthy data with the outer race fault condition. This can be seen in the similarity of the raw signals in Figures 6 and 7, respectively. Figure 8 shows the inner race fault condition.

Within the scalograms of the MFPT dataset there are a few areas to be noted. The noise level within the baseline and outer race data appears to be higher than the inner race. This is confirmed from the plots of the raw signals. The baseline and outer race faults look similar, hence the potential difficulty in the conducting fault diagnosis on this dataset.

Although labels are available for this dataset, the results presented in this section were obtained with fully unsupervised training on both DCGAN and InfoGAN architectures, with complete datasets and without labels. Visual inspection of the output images

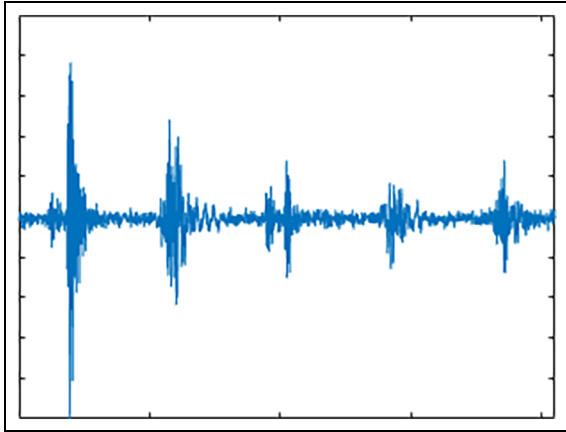


Figure 8. Inner race fault signal.

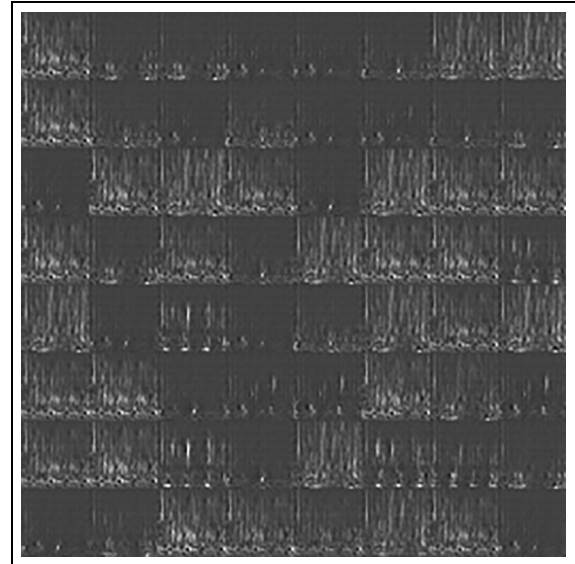


Figure 10. Output images of InfoGAN generator training model.

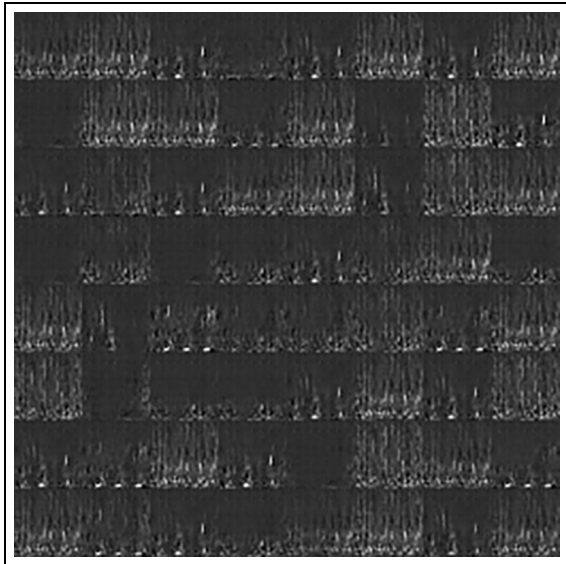


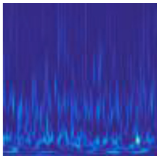

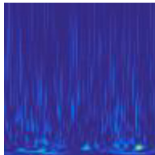
Figure 9. Output images of DCGAN generator training model.

of the generator network, as shown in Figures 9 and 10, is a key indicator to how well the GAN architecture is training and whether mode collapse, vanishing gradients, non-convergence, or checkerboarding artifacts is

occurring. Checkerboarding artifacts occur when the stride length is not directly divisible by the convolutional filter size. The output images on this dataset show that the generator performed well in converging to the distribution of the data. It is clear which images are the inner race fault images, and there is a slight variation in the images generated for the baseline and outer race conditions.

Following the proposed methodology in section “Proposed generative adversarial fault diagnostic methodology” (see Figure 2), the alternative models can be qualitatively evaluated based on the two-component PCA. Thus, Figure 11 shows the PCA results for the best model corresponding to the spectral clustering based on the InfoGAN LA vector with an output image of 32×32 pixels. Indeed, Supplemental Appendix A contains the results for the two-component PCA based on both the InfoGAN and DCGAN data representations. For the sake of brevity, only the results for the best performing clustering method, spectral clustering,

Table 1. 96×96 pixel MFPT scalogram images (actual size).

Baseline	Inner race	Outer race
		

MFPT: machinery failure prevention technology.

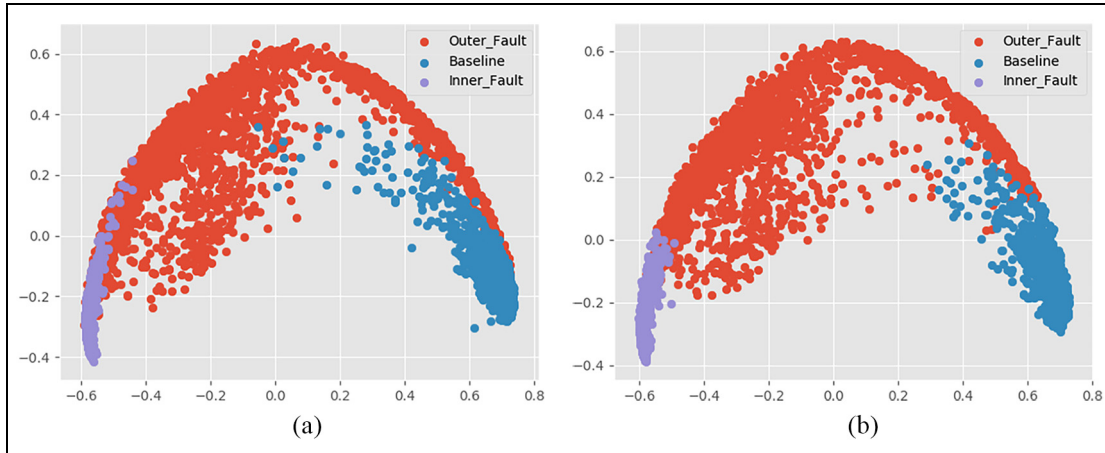


Figure 11. Spectral clustering PCA, InfoGAN LA output image 32×32 pixels. (a) Real labels. (b) Predicted labels.

Table 2. Fully unsupervised 32×32 generator output, InfoGAN LA output and spectral clustering.

Labeled data (%)	ARI	Purity	NMI
0	0.89	0.96	0.88

InfoGAN: information maximizing generative adversarial networks; ARI: adjusted rand index; NMI: normalized mutual information.

are shown. Both Supplemental Appendix A and Figure 11 compare the predicted labels with the real labels. Note that, from the results in Supplemental Appendix A, the InfoGAN LA vector with an output image of 32×32 pixels provides the best separation and identification of the system's health states. This model is closely followed by the InfoGAN LA + vector with an output image of 32×32 pixels that, when contrasted to the real labels, shows some difficulty in separating the baseline health state.

This qualitative evaluation is important within the proposed methodology, because for unsupervised fault diagnostics, the first step for this data representation is a clear separation between the baseline healthy data and any faulty unhealthy data. The faults themselves do not necessarily need to be separated from each other at this stage, as the goal of this step is to separate healthy from unhealthy. Isolating faults between each other can be assessed in a later stage of the proposed methodology as the engineer begins to label data and has further knowledge into the ground truth of the signals. As can be seen from Figure 11, for the best model, InfoGAN LA vector with image output of 32×32 pixels, the baseline is separated well from the rest of the fault signal data.

The last convolutional layer activation of the GAN's generator allows visualization of the manifold GAN developed during training of the underlying gradient

basis of the raw data. This layer holds valuable information about the underlying distribution of the data.

The effectiveness of the proposed methodology can be evaluated with the following metrics: ARI, NMI, and purity. ARI and NMI are well-known evaluation metrics; however, purity is somewhat new but used often. Purity, simply put, is the ratio between the dominant class in the cluster and the size of the cluster. More formally, purity is as in equation (7)

$$\text{Purity}(w_i) = \frac{1}{n_i} \max_n (n_{ij}) \quad j \in C \quad (4)$$

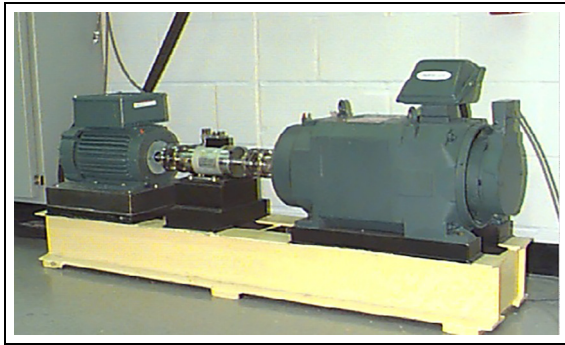
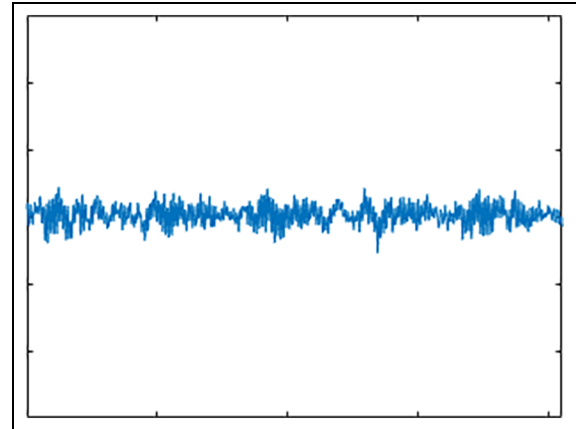
where w are clusters, n are members, and C are the number of classes.

The three metrics used for evaluation all measure different aspects of the effectiveness of unsupervised learning algorithms. Purity values range from 0 (poor clustering) to 1 (perfect clustering). A high purity would be easy to achieve if the selected number of clusters is high. For instance, if every feature from the proposed methodology had its own cluster, the purity would be 1. Therefore, purity cannot be used to evaluate the number of clusters. NMI allows one to evaluate the tradeoffs of the number of clusters. However, NMI has the same drawback as purity does where if there are one-image clusters, NMI has a value of 1. The last metric used to evaluate the clustering output is ARI. ARI, simply put, is the accuracy of the clustering and

Table 3. 32×32 generator output, InfoGAN LA output, and spectral clustering.

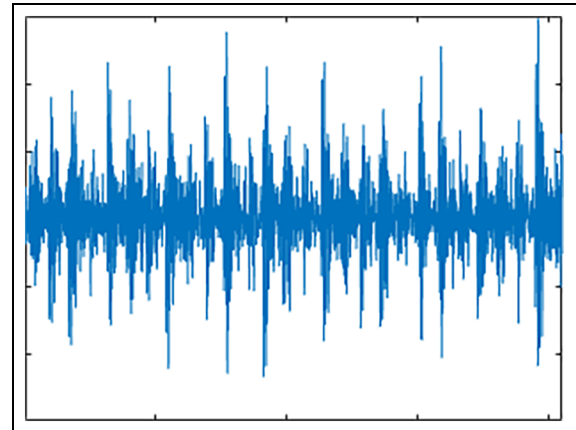
Labeled data (%)	Amount of labeled data	ARI	Purity	NMI
0	0	0.89	0.96	0.88
1	54	0.37	0.73	0.45
2	108	0.46	0.79	0.59
4	216	0.82	0.94	0.81
8	432	0.88	0.96	0.87
10	541	0.90	0.96	0.88
20	1081	0.98	0.99	0.96

InfoGAN: information maximizing generative adversarial networks; ARI: adjusted rand index; NMI: normalized mutual information.

**Figure 12.** CWR experimental test stand for roller bearing.**Figure 13.** Baseline raw signal.

measures the percentage of correct decisions. ARI gives equal weight to the false positives and false negatives. This accounts for the shortcomings of purity and NMI where, at times, ARI can perform worse when separating similar data points than clustering dissimilar data points. From the complete set of results shown in Supplemental Appendix B, Tables B.1 to B.4, the proposed architectures for the DCGAN and InfoGAN provide a robust underlying manifold representation of the data and they have solid performance for unsupervised fault diagnostics. The InfoGAN LA vector with 32×32 output images and with spectral clustering is the one that achieves the best results: ARI of 0.89, purity equal to 0.96 and NMI of 0.88 as shown in Table 2. This indicates the proposed methodology is creating pure clusters, the number of clusters is generating a high NMI, and the ARI accuracy of 0.89 is high for unsupervised learning.

Moreover, the InfoGAN architecture with the 32×32 generator output outperformed the 96×96 output. This could be explained by the similarities between the baseline and the outer race fault condition. With increased generator resolution potentially blurring the images, the GAN models could therefore have a harder time classifying them. Based on the ARI, NMI, and purity results, there is no definitive optimal image resolution for both the architectures. Spectral

**Figure 14.** Inner race fault raw signal.

clustering outperformed k-means++ across the board. The ability of spectral clustering to map to a lower dimensional space allowed for better predictions. Therefore, for the MFPT dataset, the InfoGAN outperformed the DCGAN. Given the noise the MFPT dataset has within two of the classes, the InfoGAN did a better job of encoding the experimental noise into the z vector.

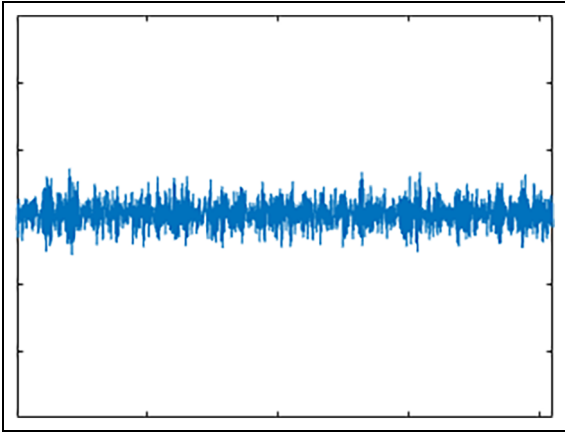


Figure 15. Outer race fault raw signal.

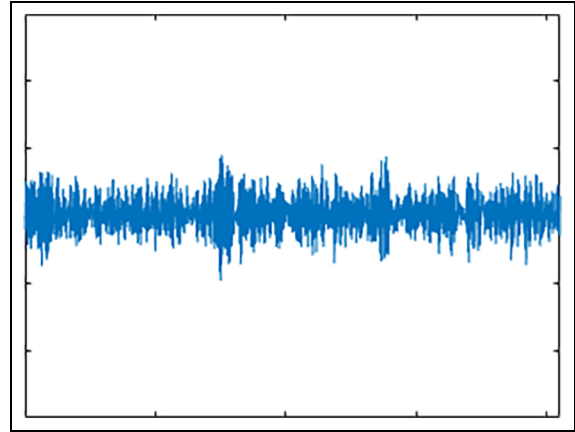


Figure 16. Ball fault raw signal.

The next step would be to monitor the system as baseline data is collected. As faults arise, inspection and knowledge of faults must be completed to ensure the fault diagnostic system improves. These results indicate a strong value proposition for the proposed methodology. The following section explores increasing the percentage of labeled data within this methodology.

As the results of the unsupervised learning are obtained, semi-supervised learning may be required if some of the results do not meet the user requirements for prediction capability. Although the fully unsupervised results for this dataset are satisfactory, with best purity scores of 0.96 and 0.82 for the InfoGAN and DCGAN, respectively, and good separation on the PCA plot for the identification of health states, the semi-supervised case is explored to see how good the results can be with an investment in resources to label the data. This is a time-consuming and expensive process, so we analyze different cases that incrementally add labels to the dataset. The percentage of the labeled data is dependent on knowledge of the failure process, degradation, application, quality of the data, feeling/expert knowledge, and associated costs. For the sake of brevity, in this section, we focus on the architecture with the best performance in the unsupervised stage discussed in the previous section, that is, the InfoGAN LA vector with 32×32 generator output. Note also that the models are trained with only a small portion of the dataset that is labeled.

The top results are reported in Table 3, which are for the InfoGAN architecture with LA output image 32×32 pixels using spectral clustering. To evaluate effectiveness of the semi-supervised fault identification pipeline, the actual labels are compared with predicted clusters (predicted health states).

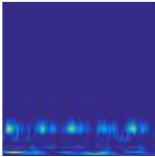
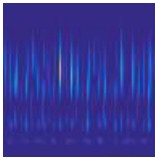
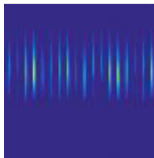
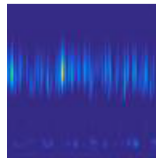
Something peculiar in the results is the fact that the metrics performance decreases initially with the addition of labeled data. This is because the semi-supervised models are trained with labels on a very small portion of the full dataset. The most important part of these validation metric results is the point at which the semi-supervised case begins outperforming the unsupervised case. This happens at 8%. The semi-supervised case is able to match the unsupervised results with only 4% labeled data and surpass it with 8%. This gives the engineer a decision point with which to make an economic decision to start labeling data. Compared with the fully unsupervised, the semi-supervised results show a better separation overall of the baseline versus the fault data.

In sum, at a 0.94 purity from the spectral clustering results out of the InfoGAN c vector, it is worth exploring this unsupervised approach for this dataset before spending engineering resources on labeling the vast amount of data for similar systems in industry. Also, with the addition of the labeled data, there are few points worth commenting. First, spectral clustering still outperformed k -means++. The results for the low percentage labeled data show almost equal performance compared with the unsupervised results, as shown in Supplemental Appendix B, Tables B.1 to B.4. This is not surprising as the unsupervised results were already high. These results indicate that unsupervised results can be achieved with a small labeled subset.

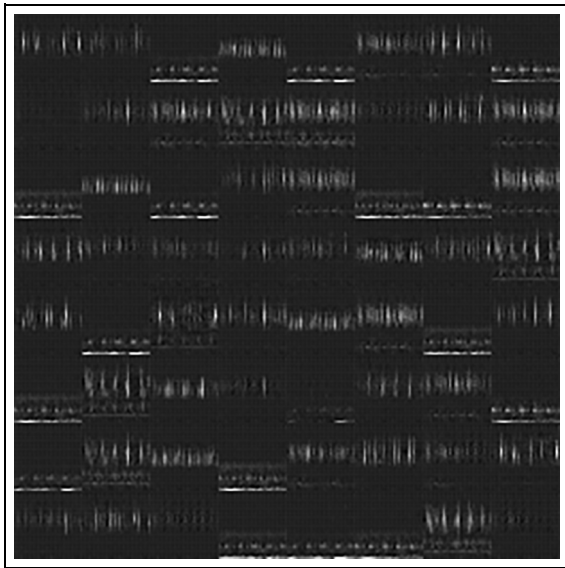
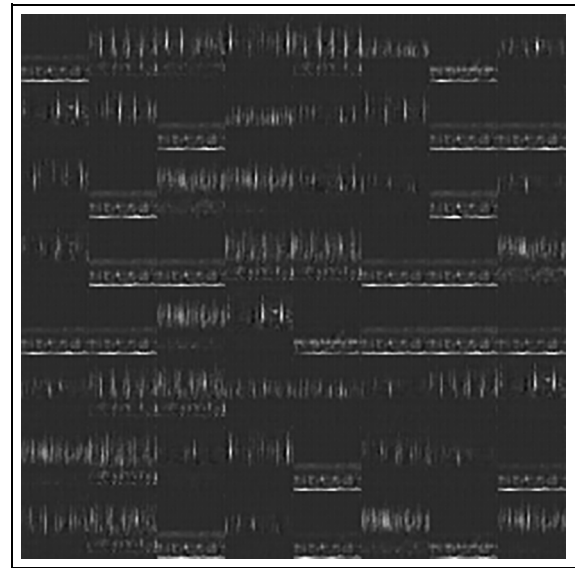
CWR University bearing dataset

The second experimental dataset was provided by CWR University Bearing Data Center.¹⁴ A Reliance electric motor, 2 horsepower, was used with ball bearings in experiments for the acquisition of vibration accelerometer data on both the drive end and fan end

Table 4. 96×96 pixel CWR scalogram images of the faults.

Baseline	Inner race	Outer race	Ball fault
			

CWR: Case Western Reserve.

**Figure 17.** Output images of DCGAN generator training model.**Figure 18.** Output images of InfoGAN generator training model.

bearings, as shown in Figure 12. The signal is generated from the bearings supporting the motor shaft. Single point artificial faults were seeded in the bearing with an electro-discharge machining. Location and diameter of the faults varied for the outer raceway. In addition, 0–3 horsepower motor loads were included within the experimental data. Accelerometers were attached via magnets to the housing on the 12 o'clock location.

For the purposes of this article four classes were used: baseline, inner raceway, outer raceway, and rolling element (ball). In total, the images generated for each class was 3304, IR 2814, OR 2819, BF 2816, respectively. These classes were assembled by combining the fault sizes, motor speed, and motor load. The training set size again was set to 50% of the 11,753 total images. To ensure the images were a computationally efficient size, bilinear interpolation³⁰ was used to scale the images down to a manageable size for the training. For the CWR dataset, any analysis incorporating the rolling element (ball fault) data requires more sophisticated algorithms than envelope analysis.³¹ Visually, one

can see from Figures 13 to 16 that this would hold true. The ball fault signal (Figure 16) appears to mimic parts of both baseline and outer race fault signals.

From the raw signals, the following scalograms were generated based on the procedure presented in section “Background on adversarial training.” Bilinear interpolation was used to scale the image down to a usable size (96×96 and 32×32 pixels) for training the GAN. Samples of these images are shown in Table 4. One can see the ball fault images may mimic the higher frequency outputs of the outer race faults, and the lower frequency response of the baseline signals. Also note that, overall, the noise in this dataset appears to be less than that of the MFPT dataset.

After training both proposed architectures for DCGAN and InfoGAN, the output images on this dataset, as shown in Figures 17 and 18, appear to show that the generator performed well in converging to the distribution of the data.

The PCA of the first two components of the LA vector (DCGAN) and $LA +$ vector (DCGAN and

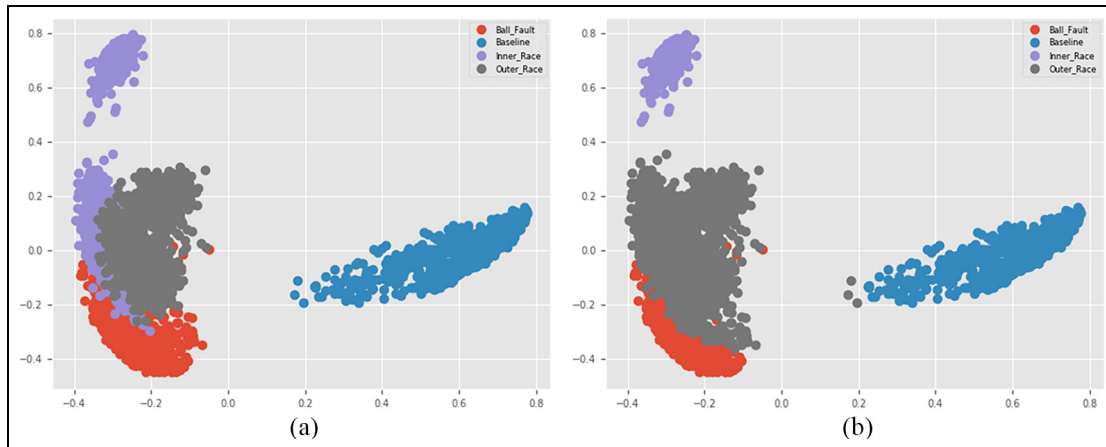


Figure 19. K-means++ PCA, DCGAN LA output image 96×96 pixels. (a) Real. (b) Predicted.

Table 5. CWR 96×96 generator output, DCGAN k-means++ clustering.

Labeled data (%)	ARI	Purity	NMI
0	0.69	0.82	0.78

CWR: Case Western Reserve; DCGAN: deep convolutional generative adversarial networks; ARI: adjusted rand index; NMI: normalized mutual information.

InfoGAN) representations are compared. The predicted and real labels are shown in the Supplemental Appendix B for all the models based on the best performing clustering method. For the CWR dataset, this is k-means++ with the DCGAN LA vector on 96×96 generator images as shown in Figure 19. However, one can observe that PCA operating on the DCGAN and InfoGAN training had difficulties with the baseline data separation. It appears that the ball fault data results in two sets of clusters in the PCA, which is difficult for the clustering methods that do not employ a higher dimensional space to separate.

The ARI, purity, and NMI metrics are again used to validate the proposed methodology on this dataset. The complete set of results can be found in Supplemental Appendix C, Tables C.1 to C.4. For the CWR dataset, it is the 96×96 generator output on the DCGAN utilizing k-means++ clustering that delivers the best results with ARI, purity, and NMI scores equal to 0.69, 0.82, and 0.78, respectively, and is shown in Table 5. Note that these are much lower than the unsupervised results of the MFPT dataset.

The unsupervised results for the CWR dataset are low and appear as though they could benefit from the addition of labeled data to the training. Based on these results, the next section explores increasing the percentage of labeled images within the GANs training. Semi-supervised learning of the fault detection should be

explored given the lower results of the unsupervised learning.

Again, once the first model is trained, the dataset is incrementally labeled. NMI, purity, and ARI were again used to evaluate the model since the labels are known. As in the previous section, we restrict our discussion to the DCGAN architecture as it achieved the best fault diagnosis results in the fully unsupervised stage. Thus, the results are reported in Table 6 from the 96×96 generator output, using the DCGAN architecture and k-means++ clustering to separate the system health states.

The first evaluation of the results also indicates the same pattern the MFPT results had. The metric performance decreased as labels were added in smaller quantities. The point with which the semi-supervised results outperformed the unsupervised results for this dataset is between 8% and 10%. The CWR dataset benefited greatly from the addition of the labels.

K-means++ operating on the representation from the DCGAN for this dataset had better system state separation with labeling a portion of this dataset. The purity is much improved with the top model achieving 0.98 purity with 20% labeled data, whereas the unsupervised case was only 0.82. The CWR dataset is an easily separable dataset using the baseline data, inner race, and outer race faults. With the addition of the ball fault data, however, one must use more

Table 6. CWR 96×96 generator output, DCGAN k-means++ clustering.

Labeled data (%)	Amount of labeled data	ARI	Purity	NMI
0	0	0.69	0.82	0.78
1	117	0.40	0.62	0.42
2	235	0.47	0.71	0.59
4	470	0.51	0.69	0.61
8	940	0.51	0.70	0.55
10	1175	0.88	0.95	0.88
20	2350	0.95	0.98	0.94

CWR: Case Western Reserve; DCGAN: deep convolutional generative adversarial networks; ARI: adjusted rand index; NMI: normalized mutual information.

Table 7. MFPT unsupervised AE and VAE results.

Model	ARI	Purity	NMI
MLP AE k-means++	0.44	0.76	0.49
MLP AE spectral	0.61	0.82	0.73
Conv-AE k-means++	0.38	0.73	0.49
Conv-AE spectral	0.50	0.81	0.53
Conv-VAE k-means++	0.51	0.81	0.67
Conv-VAE spectral	0.54	0.81	0.69

MFPT: machinery failure prevention technology; AE: autoencoder; VAE: variational autoencoder.

sophisticated methods to perform fault diagnosis. The CWR dataset, in general, has less noise throughout the scalograms than the MFPT dataset. Even without the information from the latent space of the InfoGAN, the DCGAN architecture provided a better representation for these data, which benefited greatly from the addition of labeled data.

In sum, the CWR predictions performed worse than the MFPT dataset predictions as found in the Supplemental Appendix C for unsupervised training. K-means++ outperformed spectral clustering but not always. A 32×32 generator output versus a 96×96 generator output was less clear. For the DCGAN and k-means++, the 96×96 output performed better. However, for spectral clustering, both output sizes performed poorly. K-means++ with a DCGAN architecture and a 96×96 pixel generator output had the highest purity measure.

Comparison with AE and VAE

To evaluate the proposed methodology, a baseline against AE and VAE was completed on the same set of scalogram images. The same external clustering evaluation metrics are used to assess the methodology. The features are extracted from the encoder output for the autoencoder architecture and from the z-mean output in the VAE case.

For the AE two architectures are considered: one based on FC layers (multilayer perceptron (MLP)-AE) and another with convolutional layers (Conv-AE). At least two layers are used for the encoder/decoder (thus using at least four layers given symmetric encoder–decoder) to allow the AE to generate complex enough features. Given this base architecture, layers or hidden units are added until the following qualitative criteria is met: after 10,000 iterations we reconstruct 10 images and decide based on image quality if the decoder generated is a good reconstruction. The loss function is the mean square error between reconstruction and input image.

Based on the procedure established by the proposed methodology, Figures 20 to 22 show the PCA visualizations based on the results obtained from the MLP-AE, Conv-AE, and Conv-VAE architectures, respectively. Note that all PCAs have an explained variance near 90% so the visualizations are a good approximation of the general structure of the data.

In the MLP-AE results, the structure of the features is not so clear given the overlap and the spread of the data structure. An explanation for this behavior is the nature of MLP when they are used on images: the spatial information is hard to encode, so more complex transformations are required. This hypothesis is supported comparing this with the structure found by the Conv-AE where a half-moon structure is found. From the results reported in Tables 7 (MFPT) and 8 (CWR),

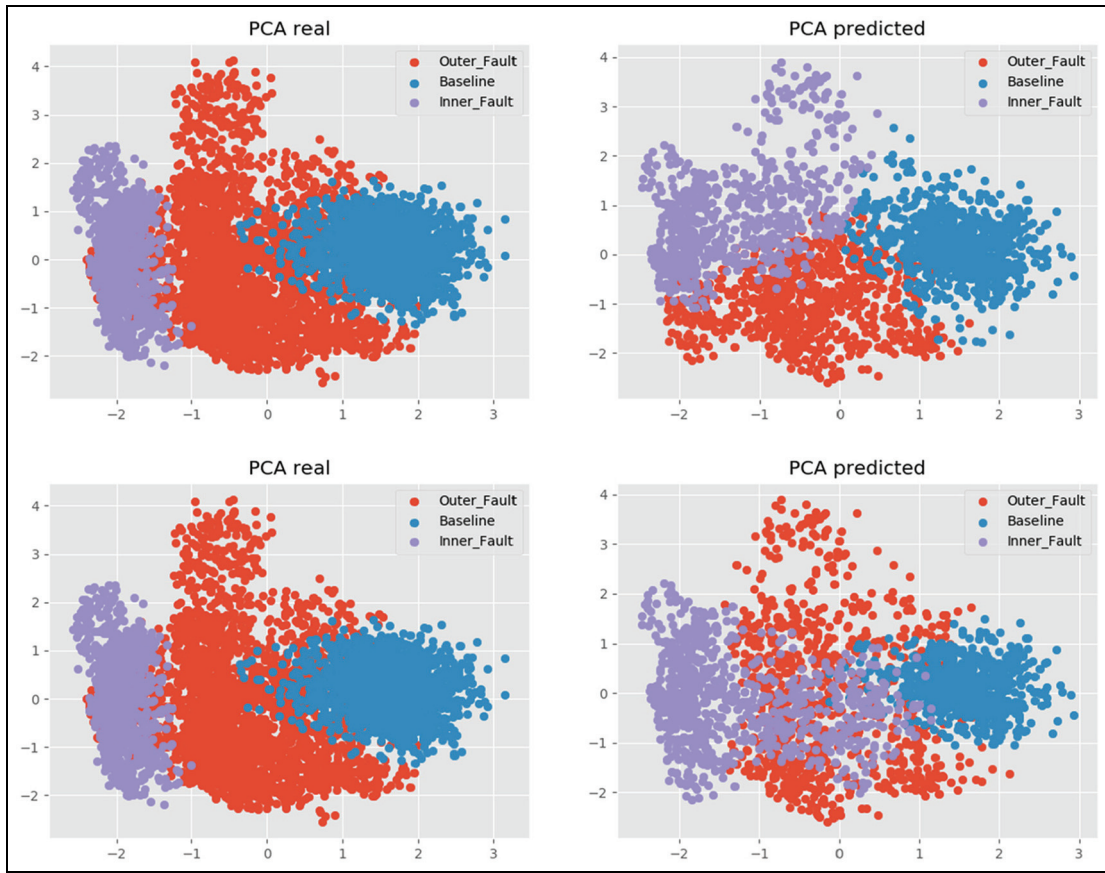


Figure 20. MFPT AE MLP architecture.

we get consistently high results in most of the metrics for Conv-VAE. If we consider only purity, the Conv-VAE is marginally outperformed by MLP-AE for both the MFPT, Figure 24, and CWR, Figure 23, datasets but, in terms of representation, the Conv-VAE is preferable as shown in Figures 22 (MFPT) and 25 (CWR). The Conv-VAE architecture has the best baseline signal separation among the three models. Despite these results for the MLP and VAE-based approaches, the proposed GAN-based methodology outperforms all models, as it can be substantiated by comparing the results in Tables 7 (MFPT) and 8 (CWR) with Tables 2 (MFPT) and 5 (CWR).

These results indicate a limitation of the proposed methodology where the available clustering evaluation metrics only measure the clustering of a representation, not the representation itself. However, the representations do exhibit a consistent intrinsic structure between them. The convolutional VAE and AE together with the GAN representation result in similar structures. The main difference seems to be the ease of the clustering algorithm to separate this structure. The results indicate that complex models tend to ease this process

with higher ARI, NMI, and purity scores. For example, in the case of the MFPT dataset, if we compare these results with the top GAN results, the best non-GAN model ranks fifth best (see Table 2 and Supplemental Appendix B). This indicates that a lower computational cost methodology can achieve reasonable results; however, to increase ARI, NMI, and purity scores a considerably more complex model is required.

Concluding remarks

Unsupervised fault diagnostics is a critical area of research with applications in many industries. The ability to detect faults when there is almost zero ground truth, with little to no labeled data, and from big multi-dimensional machinery data has vast economic benefits. In this article, a novel deep generative adversarial multistage methodology is proposed for fault diagnostics. This methodology achieved superior unsupervised prediction results over both AEs and VAEs. These results are then further improved with the addition of a subset of labeled data.

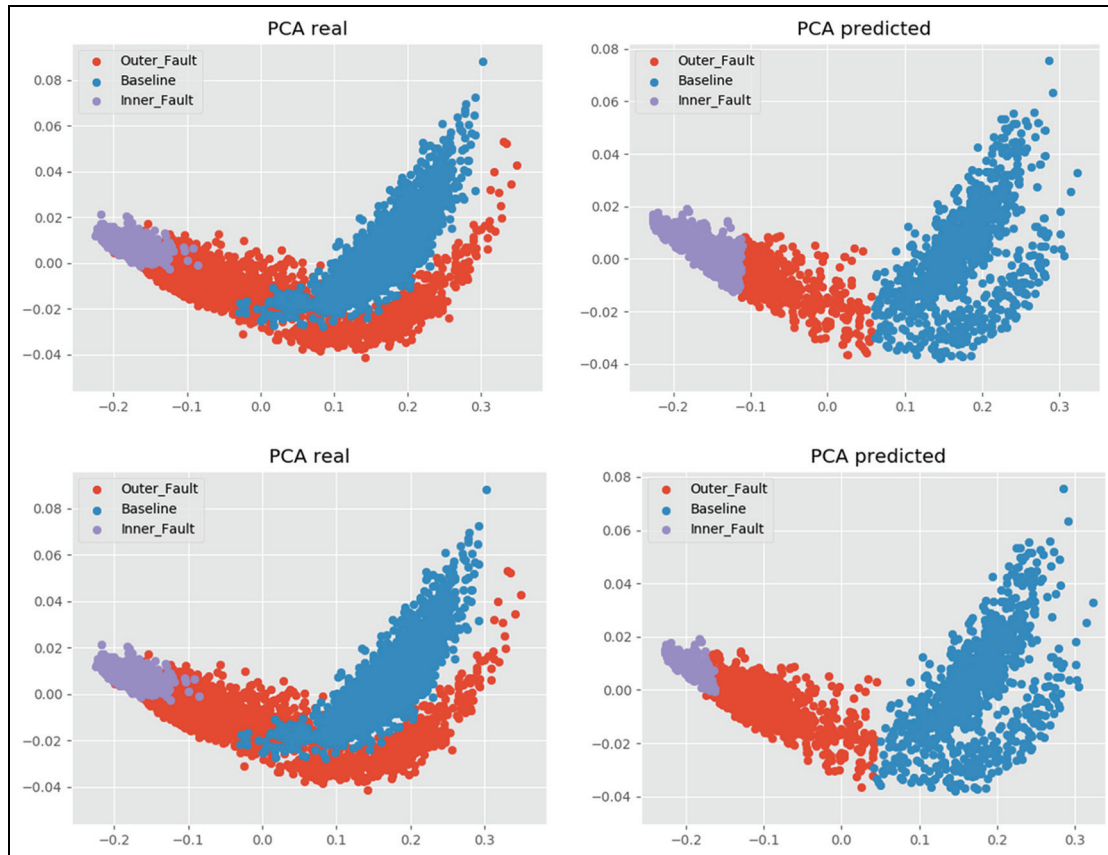


Figure 21. MFPT AE convolutional architecture.

To achieve the results presented in this article, the outputs of the activation layers in both DCGANs and InfoGANs were examined within two traditional clustering algorithms: (1) k-means++ and (2) spectral. These two clustering algorithms were chosen to prove the robustness and flexibility of the GAN-based methodology to simple clustering techniques. The InfoGAN encoder vector was tested as an additional feature for clustering; however, the addition of the encoder information had mixed results. It appears the InfoGAN architecture outperforms the DCGAN on noisier data like the MFPT set. Both architectures' performance benefited from labeling even a small portion of the data.

While these initial results showed promise, there are limitations and research is in process to address them. The MFPT dataset is simple enough for envelope analysis classification if the signals are known; however, the CWR dataset cannot be diagnosed with envelope analysis alone. The MFPT dataset did include varied loads and multiple sampling frequencies which were not explored in this work. The amount of data within each of those datasets was insufficient for the methodology

and resulted in overfitting. Varied rotational speeds were also not explored as the datasets did not contain them. It is widely known that training a GAN architecture can be challenging. To complete the work in this study, the training was done multiple times to ensure the GAN converged toward the Nash equilibrium without mode collapse and vanishing gradients occurring.

Generative adversarial fault diagnostics paired with the automatic feature learning inherent with deep learning has great potential benefits for many industries as more adopt a predictive maintenance program. GANs as a research topic is still, relatively speaking, in its infancy. It has been accelerating and proliferating through other research communities at a fast pace since 2014. This is the first article to incorporate it into fault diagnostics. The proposed methodology proves that it is flexible enough to incorporate engineering expertise as that expertise grows. In fact, the proposed methodology demonstrates that fault diagnostics are strengthened by the meaning engineering expertise can give to the learned GAN feature representations. DCGANs prove their ability to diagnose faults with zero information on the real classes within the dataset. Moreover,

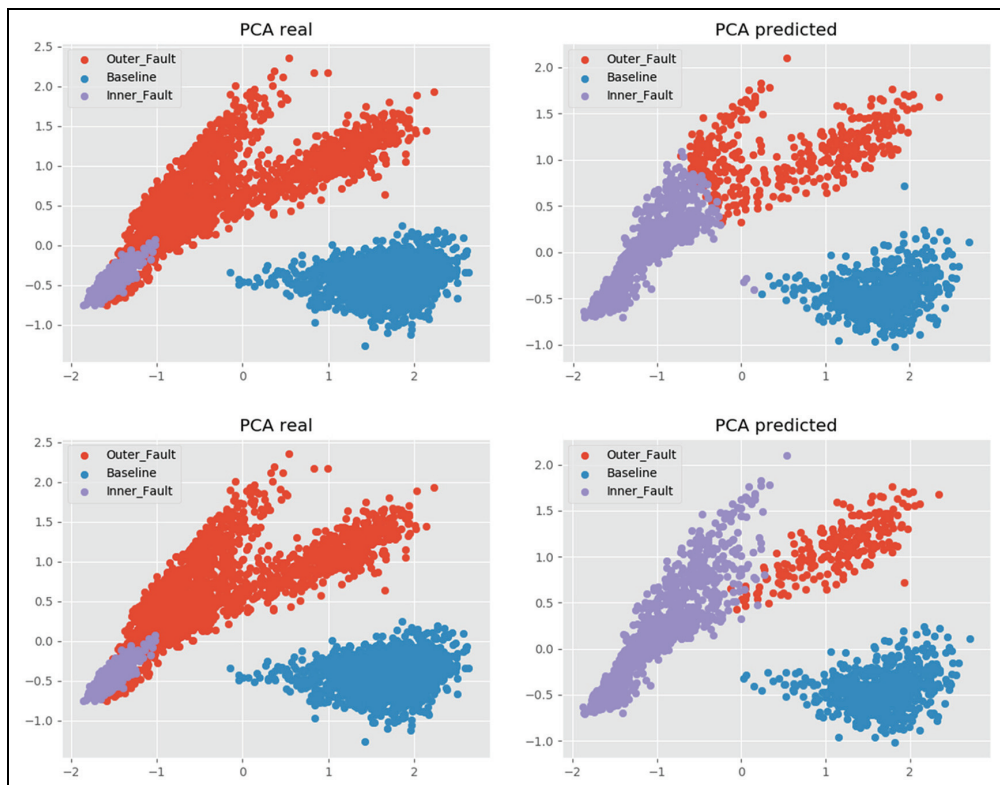


Figure 22. MFPT VAE convolutional architecture.

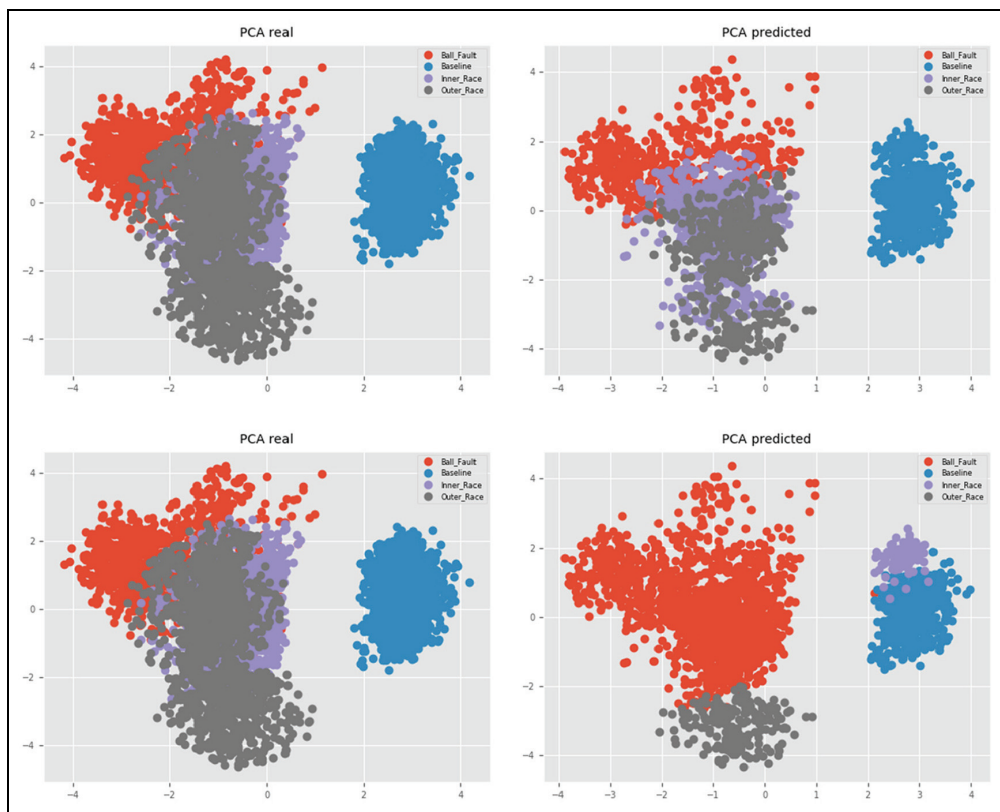


Figure 23. CWR AE MLP architecture.



Figure 24. CWR AE convolutional architecture.

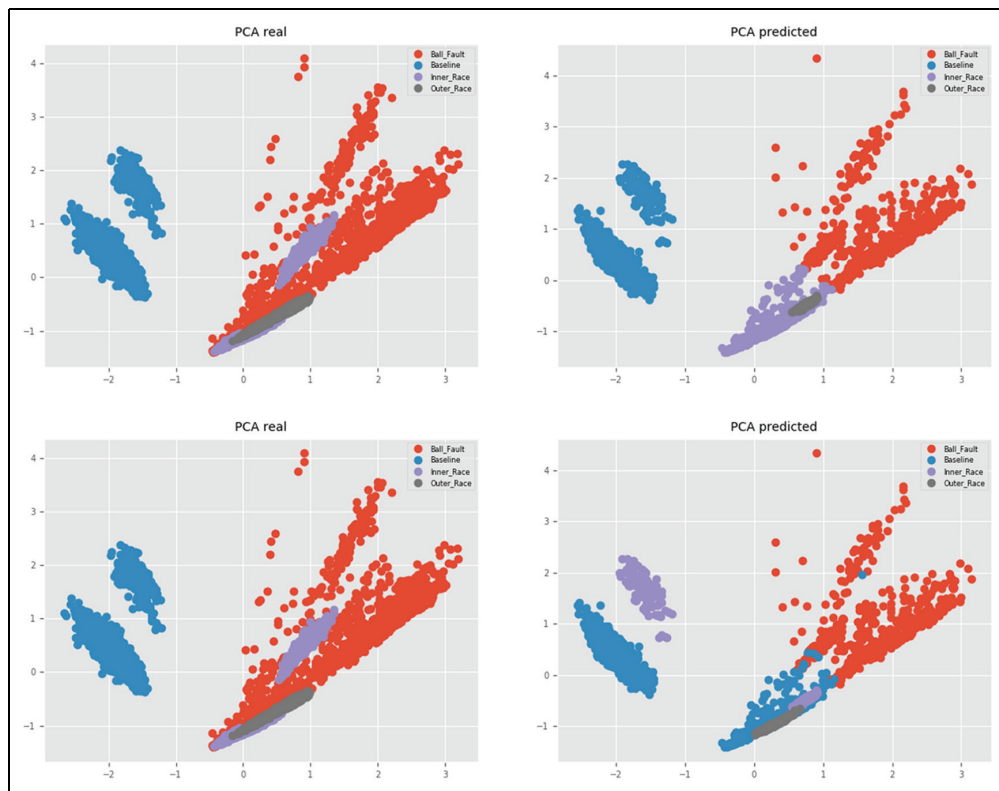


Figure 25. CWR VAE convolutional architecture.

Table 8 CWR unsupervised AE and VAE results.

Model	ARI	Purity	NMI
MLP AE k-means + +	0.49	0.69	0.55
MLP AE spectral	0.35	0.60	0.59
Conv AE k-means + +	0.21	0.53	0.27
Conv AE spectral	0.38	0.66	0.57
Conv VAE k-means + +	0.50	0.68	0.61
Conv VAE spectral	0.19	0.55	0.34

CWR: Case Western Reserve; AE: VAE: variational autoencoder.

InfoGANs show that, with slight knowledge of how many potential driving failure modes the rolling elements may have, the diagnostics results may be improved with little economic investment. With integrated unsupervised and semi-supervised fault diagnostics, industries such as aerospace, wind power, oil and gas, and automotive are poised to unlock new potentials for diagnostic and structural health management systems.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding


The authors acknowledge the partial financial support of the Chilean National Fund for Scientific and Technological Development (Fondecyt) under Grant No. 1160494.


Supplemental material

Supplemental material for this article is available online.

ORCID iDs

Enrique López Droguett  <https://orcid.org/0000-0002-0790-8439>

Viviana Meruane  <https://orcid.org/0000-0002-6692-2687>

Andrés Ferrada  <https://orcid.org/0000-0002-5435-8978>

References

1. Modarres C, Coburger A, Droguett EL, et al. Computer vision for damage recognition and type identification: a deep learning based approach. In: *Proceedings of the ESREL*, 18–22 June 2017, Portorož.
2. Wang L, Zhao X, Pei J, et al. Transformer fault diagnosis using continuous sparse autoencoder. *Springerplus* 2016; 5(1): 448.
3. Langone R, Reynders E, Mehrkanoon S, et al. Automated structural health monitoring based on adaptive kernel spectral clustering. *Mech Syst Signal Pr* 2017; 90: 64–78.
4. Lei Y, Jia F, Lin J, et al. An intelligent fault diagnosis method using unsupervised feature learning towards mechanical big data. *IEEE T Ind Electron* 2016; 63(5): 3137–3147.
5. Li X, Ding Q and Sun JQ. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab Eng Syst Safe* 2018; 172: 1–11.
6. Lee S, Ha J, Zokhirova M, et al. Background information of deep learning for structural engineering. *Arch Comput Meth Eng* 2018; 25: 121–129.
7. Jiang P, Hu Z, Liu J, et al. Fault diagnosis based on chemical sensor data with an active deep neural network. *Sensors* 2016; 16(10): 1695.
8. Sun W, Shao S, Zhao R, et al. A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement* 2016; 89: 171–178.
9. Liao L, Jin W and Pavel R. Enhanced restricted Boltzmann machine with prognosability regularization for prognostics and health assessment. *IEEE T Ind Electron* 2016; 63(11): 7076–7083.
10. Wang J and Zhang C. Software reliability prediction using a deep learning model based on the RNN encoder–decoder. *Reliab Eng Syst Safe* 2018; 170: 73–82.
11. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. *Adv Neur Inform Process Syst* 2014; 27: 2672–2680.
12. Kingma DP and Welling M. Auto-encoding variational bayes, 2013, arXiv:1312.6114
13. Bechhoefer E. A quick introduction to bearing envelope analysis, 2016, <https://www.scribd.com/document/110769949/A-Quick-Introduction-to-Bearing-Envelope-Analysis>
14. Loparo KA. Bearing Data Center, Case Western Reserve University, 2013, <http://cseggroups.case.edu/bearingdata-center/pages/welcome-case-western-reserve-university-bearing-data-center-website>
15. Manning CD, Raghavan P and Schütze H. *Introduction to information retrieval*. New York: Cambridge University Press, 2008.
16. Kuncheva LI. *Combining pattern classifiers: methods and algorithms*. Hoboken, NJ: John Wiley & Sons, 2004.
17. Hubert L and Arabie P. Comparing partitions. *J Classif* 1985; 2(1): 193–218.
18. Jebara T. Generative versus discriminative learning. In: Jebara T (ed.) *Machine learning* (The international series in engineering and computer science), vol. 755. Boston, MA: Springer, 2004.

19. Arthur D and Vassilvitskii S. k-means++: the advantages of careful seeding. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms*, 2007. Society for Industrial and Applied Mathematics, <http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>
20. Shi J and Malik J. *Normalized cuts and image segmentation*. Departmental Papers (CIS): 107, 2000, http://www.cis.upenn.edu/~jshi/papers/pami_ncut.pdf
21. Verstraete D, Ferrada A, Droguett EL, et al. Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings. *Shock Vib* 2017; 2017: 5067651.
22. Radford A, Metz L and Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016, arXiv:1511.06434
23. Feng Z, Liang M and Chu F. Recent advances in time-frequency analysis methods for machinery fault diagnosis: a review with application examples. *Mech Syst Signal Pr* 2013; 38(1): 165–205.
24. Lin J and Qu L. Feature extraction based on Morlet wavelet and its application for mechanical fault diagnosis. *J Sound Vib* 2000; 234(1): 135–148.
25. Chen X, Duan Y, Houthoofd R, et al. InfoGAN: interpretable representation learning by information maximizing generative adversarial nets, 2016, arXiv:1606.03657
26. Metz L, Poole B, Pfau D, et al. Unrolled generative adversarial networks, 2016, arXiv:1611.02163
27. Gulrajani I, Ahmad F, Arjovsky M, et al. Improved training of Wasserstein GANs, 2017, arXiv:1704.00028
28. Salimans T, Goodfellow I, Zaremba W, et al. Improved techniques for training GANs, 2016, arXiv:1606.03498
29. Odena A, Dumoulin V and Olah C. Deconvolution and checkerboard artifacts. *Distill* 2016; 1(10): e3.
30. Raveendran H and Thomas D. Image fusion using LEP filtering and bilinear interpolation. *Int J Eng Trend Technol* 2014; 12(9): 427–431.
31. Smith WA and Randall RB. Rolling element bearing diagnostics using the Case Western Reserve University data: a benchmark study. *Mech Syst Signal Pr* 2015; 64: 100–131.