Mechanical Systems and Signal Processing 168 (2022) 108709

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Physics-informed deep learning for signal compression and reconstruction of big data in industrial condition monitoring

Matthew Russell [a], Peng Wang [a,b,*]

[a] Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40506, USA
[b] Department of Mechanical Engineering, University of Kentucky, Lexington, KY 40506, USA

ARTICLE INFO

Communicated by D. Wang

Keywords:
Physics-informed deep learning
Prognostics and health management
Data compression
Big data

ABSTRACT

The onset of the Internet of Things enables machines to be outfitted with always-on sensors that can provide health information to cloud-based monitoring systems for prognostics and health management (PHM), which greatly improves reliability and avoids downtime of machines and processes on the shop floor. On the other hand, real-time monitoring produces large amounts of data, leading to significant challenges for efficient and effective data transmission (from the shop floor to the cloud) and analysis (in the cloud). Restricted by industrial hardware capability, especially Internet bandwidth, most solutions approach data transmission from the perspective of data compression (before transmission, at local computing devices) coupled with data reconstruction (after transmission, in the cloud). However, existing data compression techniques may not adapt to domain-specific characteristics of data, and hence have limitations in addressing high compression ratios where full restoration of signal details is important for revealing machine conditions. This study integrates Deep Convolutional Autoencoders (DCAE) with local structure and physics-informed loss terms that incorporate PHM domain knowledge such as the importance of frequency content for machine fault diagnosis. Furthermore, Fault Division Autoencoder Multiplexing (FDAM) is proposed to mitigate the negative effects of multiple disjoint operating conditions on reconstruction fidelity. The proposed methods are evaluated on two case studies, and autocorrelation-based noise analysis provides insight into the relative performance across machine health and operating conditions. Results indicate that physically-informed DCAE compression outperforms prevalent data compression approaches, such as compressed sensing, Principal Component Analysis (PCA), Discrete Cosine Transform (DCT), and DCAE with a standard loss function. FDAM can further improve the data reconstruction quality for certain machine conditions.

1. Introduction

With ever-accelerating connectivity and the onset of low-power devices, the Internet of Things (IoT) promises to usher in Industry 4.0, providing real-time information streams from distributed device networks [1]. These streams send machine metrics to cloud-based resources for industrial condition monitoring and Prognostics and Health Management (PHM). Fused via state-of-the-art data analysis algorithms, these inputs will facilitate optimization of maintenance activities, eliminate downtime, and improve production efficiency [2]. While the value of Big Data for industrial PHM and other applications has been well demonstrated in smart manufacturing paradigms [3], the new data transmission and analysis pipelines significantly challenge the network infrastructure

* Corresponding author at: Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40506, USA.
E-mail addresses: matthew.russell@uky.edu (M. Russell), edward.wang@uky.edu (P. Wang).

https://doi.org/10.1016/j.ymssp.2021.108709
Received 8 June 2021; Received in revised form 18 November 2021; Accepted 26 November 2021
Available online 23 December 2021
0888-3270/© 2021 Elsevier Ltd. All rights reserved.

on the shop floor with staggering volumes of data for real-time and offline analysis. A popular solution to address this challenge adopts the pairing of data compression (before data transmission, on the shop floor) and data reconstruction (after transmission, in the cloud). However, data compression may run the risk of losing too many details and distorting the data characterization, corrupting the data analytics and PHM decision-making after data reconstruction. The extent of the risk is influenced by raw data quality (e.g., signal-to-noise ratio), compression ratio, balance among data from different conditions, etc. Viable solutions should demonstrate robustness to these uncertainty sources while achieving high-ratio compression. Furthermore, in the context of industrial PHM, a good data compression and reconstruction algorithm must also be able to preserve the important characteristics of faulty data for later analysis, i.e., distinction from normal data and differentiation among faulty conditions [4]. This necessitates special attention be given to protecting critical characteristics of the data when designing an appropriate compression mechanism.

Data compression and reconstruction algorithms can be generally classified into two groups: model-driven and data-driven Machine Learning (ML) algorithms. In contrast to model-driven algorithms like the Fourier Transform (FT), the Discrete Wavelet Transform (DWT), and the Discrete Cosine Transform (DCT), various ML algorithms allow the data compression to be trained for a specific application scenario with certain data characteristics. For example, an ML model could focus on vibration signals in milling process's range of operating frequencies instead of relying on a representation designed to universally compress one-dimensional, time-varying signals. Hence, ML algorithms typically outperform model-driven algorithms in specific applications.

ML—and especially emerging Deep Learning (DL)—algorithms can compress the data adaptively according to the information revealed by the data itself. As a representative of linear ML techniques, Principal Component Analysis (PCA) can decompose signals into explanatory components depending on the extent of information richness. Then the components with the largest coefficients can be used as the compressed signal. Advancing from linear data compression, Autoencoders (AEs) have emerged as powerful, nonlinear, DL extension of PCA that can find more nuanced mappings [5]. AEs use an encoder neural network (NN) to map an input signal into a latent vector (usually smaller than the input), while imposing a mixture of dimensionality, denoising, sparsity, and continuity constraints. The latent encoding is then mapped back to the input space for data reconstruction by a decoder NN, which usually mirrors the structure of the encoder. AE-based methods naturally have received much attention for compression of both one-dimensional time series and two-dimensional images [6]. Interestingly, most 1D time-varying AE compression research is dominated by biometric signals, such as electrocardiograms (ECG) for heart monitoring and electroencephalograms (EEG) for brain monitoring. A standard AE was investigated to compress EEG signals and demonstrated its superiority over DCT, DWT, and PCA methods [7]. AE has also been implemented in a DL architecture, for example with a stack of multiple convolutional and pooling layers in the encoder and multiple upsampling layers in the decoder, to compress and reconstruct time-domain ECG signals [8]. A similar deep AE architecture was used to compress the Fast Fourier Transform (FFT) of ECG signals [9]. These studies evidence the potential benefits of AEs and their DL implementations for data compression and reconstruction.

Despite the similarities between biometric data and industrial sensing signals, AE-based data compression has not been fully investigated for industrial PHM. Both domains rely on quasi-periodic signals with large volumes of raw data, but contemporary applications of AEs in PHM have focused instead on unsupervised anomaly detection and feature learning. An early application of AE for health management is presented in [10], which observed that an AE trained on healthy spacecraft telemetry would produce poor reconstructions of off-nominal inputs. This higher reconstruction error could be utilized to identify faults. This work was extended to manufacturing by training an AE to reconstruct FFTs of healthy acoustic emissions [11]. Anomalous sounds in this application were detected by checking if there was a drop in the reconstruction accuracy. For the purpose of feature learning, AE was demonstrated for hierarchical, unsupervised feature extraction that improved fault diagnosis in wind turbine gearboxes [12], reliability analysis of mechanical truss structures [13], and bearing remaining useful life prediction [14]. AE has also been implemented in different ML (e.g., fully connected NN) and DL architectures (e.g., convolutional and recurrent neural networks) for different application needs [15]. Furthermore, many trials have been done to improve AE's performance on feature extraction when facing a certain challenge (e.g., severe noise contamination and outliers), such as a modified denoising AE that was developed for rolling bearing fault diagnosis in [16]. As with all data-driven approaches, AE techniques can suffer from shifts in the data, which led [17] to research how the normalization of the neural network Laplacian might improve AE generalization in spite of imbalanced training data.

Several broadly applicable extensions of AEs apply optimization constraints that regularize the training of NN or DL network parameters. Weight decay is one such technique used across many NN architectures and was adopted by [7] in their AE for compressing EEG data. A contractive autoencoder (CAE) has been developed to encourage a smooth latent manifold by regularizing higher-order derivatives of the network weights [18] and has been incorporated in some AEs for PHM vibration signal compression [19]. Also, sparsity penalties on the latent vector using the L1 norm have been investigated as regularization for feature extraction [16,20]. Leveraging these general variations of AEs can be valuable, but more targeted approaches are needed for the compression of data with varying characteristics in the PHM field. Surprisingly, little work has been done to discover new loss terms that might improve the performance (e.g., signal reconstruction quality, convergence, robustness) of AEs in the context of PHM with [21] providing a rare contribution by investigating a correntropy term that improves robustness against noise-induced outliers in vibration signals. Study [4] sought to integrate the Pearson's Correlation Coefficient (PCC) into the optimization loss (in addition to MSE loss), but provided no quantitative analysis of PCC's effect. To explicitly handle the variation among fault classes, they also concatenated the compressed representation with a one-hot vector encoding of the fault condition before reconstructing the input signal. However, they provided no substantial analysis or discussion to evaluate the impact of this choice. Despite the need for regularization, existing studies have not sufficiently formulated and discussed a viable approach for PHM-oriented data compression, in which faulty data has varying representations and characteristics and need to be specifically treated.

The need for robust, generalizable constraints in PHM motivates the consideration of how incorporating the underlying physics could act as an effective, domain-influenced regularizer during AE training. In general, Physics-Informed (PI) ML techniques can be grouped into three categories [22]: (1) data preprocessing (e.g., FFT); (2) network architecture design [23]; and (3) additional loss terms [24]. Although the FFT is a well-known signal preprocessing tool, FFT-augmented ML or DL has not been well-studied in the PHM field. A few prior works incorporated physical degradation models into recurrent neural networks for fault prognostics of aluminum aircraft wings [25] and wind turbine bearings [26]. As these studies indicated, the inherent connection of PHM processes to underlying physics supports the notion that PI methods can produce improved PHM models. More work is necessary to close this research gap, especially in the context of PHM data compression.

To address the need for advanced compression techniques for industrial condition monitoring, this study contributes several key components to Physics-Informed Autoencoder (PIAE)-based PHM data compression:

- Novel application of PCC loss to machine condition monitoring that demonstrates quantitative performance gains across two case studies when compared to standard Mean Squared Error (MSE) loss;
- Development of PI loss terms motivated by the physical significance of frequency content in condition monitoring along with intuitive discussions to explain how these terms influence AE training and experimental evaluation of impact across consistent DL network configurations;
- Introduction of Fault Division Autoencoder Multiplexing (FDAM), which further improves reconstruction performance by learning to compress each operating condition separately;
- Quantitative explanations for differences in reconstruction quality among operating conditions by estimating incompressible signal noise via autocorrelation.

The proposed data compression and reconstruction architecture and the PI elements are summarized in Fig. 1. In addition to a PCC-based local structure loss, PI loss terms incorporating autocorrelation and two FFT-based metrics have been investigated to leverage domain knowledge of PHM signal periodicity. Although these loss terms can improve the ability of a single AE to reconstruct signals from distinct fault conditions, the novel FDAM architecture provides an extension that isolates compression on each condition to find an optimal representation for individual fault. In all compression problems, a natural tradeoff emerges to balance reconstruction fidelity and the level of compression. Evaluating the proposed ideas across multiple compression ratios elicits information about their robustness to this tradeoff. Finally, model performance often varies across operating conditions, and thus a quantitative analysis of noise and reconstruction quality discerns whether differences in quality indicate shortcomings in the model itself or inherent incompressibility in the data. To evaluate the proposed methods on Big Data condition monitoring signals, case studies should be performed on data sets representative of the high-volume, high-velocity streams that threaten to overwhelm network resources. While extensively used for fault diagnosis case studies, the Case Western Reserve University (CWRU) Bearing Data Set has not been thoroughly utilized for state-of-the-art, DL-based compression algorithms and provides high-sampling-rate vibration data typical of rotating machinery on a smart factory floor. Therefore, the CWRU Bearing Data Set offers an appropriate case study for multi-condition, high-velocity signal compression, and estimation of relative noise content among the conditions reveals the noise disparity between normal and fault conditions that could prove difficult for a unilateral compression algorithm, necessitating an approach like FDAM. To supplement the bearing case study, a second evaluation considers vibration data from a milling process with varying process parameters to further characterize the generalization behavior of the proposed techniques.

Table 1 provides a definition of acronyms used throughout the study. The remainder of this paper is organized as follows: Section 2 presents the theoretical background for the core concepts of AE, PIAE, FDAM, PCC loss, and PI loss. Section 3 outlines the network architectures, experimental design, and testing environment for the two case studies. Section 4 presents the results of both studies and discusses the variability of reconstruction across fault conditions. Section 5 summarizes the key findings in the context of AE-based compression, PI methods, PHM applications, and the Industrial IoT.

2. Deep autoencoders for data compression

This section elaborates the fundamentals of PIAE, starting from standard AE implemented in a NN architecture to deep AE implemented in a deep convolutional neural network architecture, various PI loss terms that could augment AE's performance in data compression and reconstruction, and FDAM specifically developed for the varying data characteristics issue associated with industrial PHM.

2.1. Autoencoder neural networks

Autoencoders (AEs) implemented in a standard NN architecture seek to learn a nonlinear latent representation of the input data through a series of fully-connected layers. In this context, AEs are considered as unsupervised learning, seeking a representation capable of reproducing the original input data rather than learning classification or regression from labeled data. An AE has two primary components: an encoder that generates latent or encoded representation, and a decoder that reconstructs an approximation of the original data from the latent code. Imposing constraints on the latent space encourages the representation to have desirable characteristics, which could vary with the application area and problem domain. In most application scenarios, the goal of applying standard AEs is to extract a simplified feature representation that can be useful for downstream analysis [5]. As the feature vector is smaller than the original input length, the feature extraction process done by the encoder can be regarded as data compression. To ensure the quality of the feature representation and minimize the information loss during the compression, the decoder must then

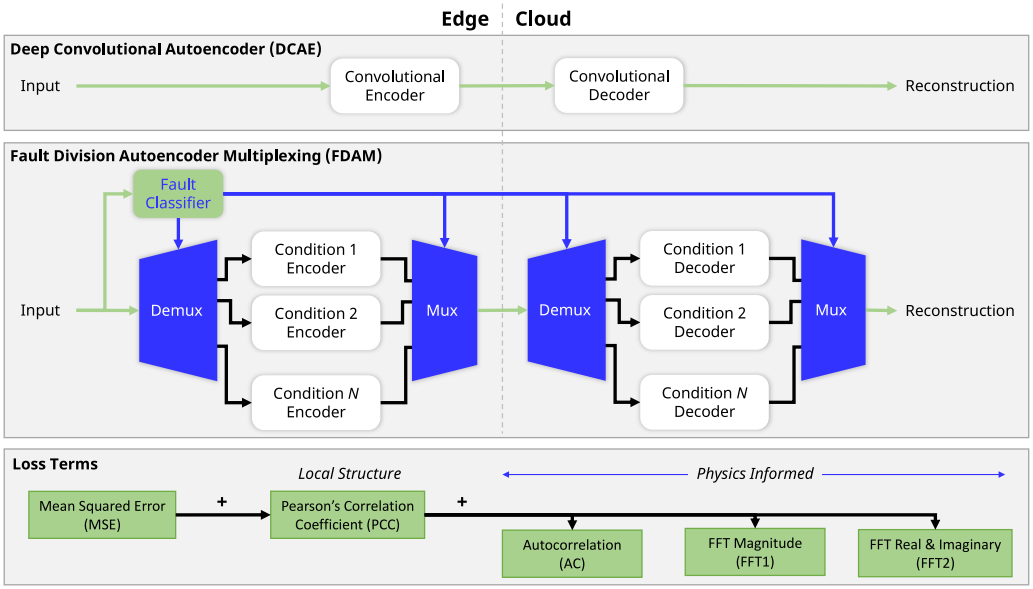

Fig. 1. Overview of the proposed study for vibration signal compression. Deep Convolutional Autoencoder (DCAE) and Fault Division Multiplexing (FDAM) architectures are compared along with combinations of loss terms extending the traditional AE MSE loss. In the FDAM section, "demux" refers to a demultiplexer with output line determined by the fault class, and "mux" refers to a multiplexer with input line selected by the fault class. Each node in the loss term tree represents a tested loss term combination consisting of the sum of all loss components up to that node in the tree.

Table 1
List of acronyms and terms used in this study.

Acronym	Definition
AC	Autocorrelation
AE	Autoencoder
AWGN	Additive White Gaussian Noise
CAE	Contractive Autoencoder
CS	Compressed Sensing
CNN	Convolutional Neural Network
CWRU	Case Western Reserve University
DCAE	Deep Convolutional Autoencoder
DCGAN	Deep Convolutional Generative Adversarial Network
DCT	Discrete Cosine Transform
Demux	Demultiplexer
DL	Deep Learning
DWT	Discrete Wavelet Transform
ECG	Electrocardiogram
EEG	Electroencephalogram
FDAM	Fault Division Autoencoder Multiplexing
FFT	Fast Fourier Transform
FT	Fourier Transform
GAN	Generative Adversarial Network
IoT	Internet of Things
ML	Machine Learning
MSE	Mean Squared Error
Mux	Multiplexer
NN	Neural Network
PCA	Principal Component Analysis
PCC	Pearson's Correlation Coefficient
PI	Physics-Informed
PIAE	Physics-Informed Autoencoder
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SNR	Signal-to-Noise Ratio

expand this encoding back to the original dimensionality and reproduce the input. Reconstruction metrics, such as the Mean Squared Error (MSE) between the output and the original input, serve as the loss functions during the network training. Mathematically, the AE can be represented as the composition of two function approximators (i.e., encoding and decoding) implemented in NN:

$$\hat{\boldsymbol{x}} = g_{\phi}^{-1}(g_{\theta}(\boldsymbol{x})) \tag{1}$$

where $g_{\theta} : \mathbb{R}^n \mapsto \mathbb{R}^m$ (with $m < n$) is the encoder with parameters θ, $g_{\phi}^{-1} : \mathbb{R}^m \mapsto \mathbb{R}^n$ is the decoder with parameters ϕ, $\boldsymbol{x}$ is the input tensor, and $\hat{\boldsymbol{x}}$ is the output tensor (i.e., the AE's approximation of the input). Following this notation, the MSE loss function can be written as

$$J_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{\boldsymbol{x}}^{(i)} - \boldsymbol{x}^{(i)}\right\|_2^2 \tag{2}$$

for N training examples $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}$. Backpropagation techniques can then be used for numerical optimization to find the parameters θ and ϕ of the NN function approximators. Typically, the better the reconstruction is, the smaller MSE achieved.

2.2. Deep convolutional autoencoders

While a shallow, fully-connected NN encoder and decoder may be sufficient to develop a useful latent representation for high-dimensional discrete features [7,10,19], advances in DL techniques have enabled deep AE designs [17,21] for time-series signal compression. Intuitively speaking, AE can be implemented in advanced DL network architectures, such as Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN), by leveraging their superior capabilities in discovering temporal and/or spatial patterns underlying signal. Both a one-dimensional (1D) CNN and RNN can be leveraged as the basic network structure for realizing AE-based continuous signal compression and reconstruction. In this study, 1D CNN is preferred, as CNN is better at preserving local features [27]. 1D CNN consists of two major types of layers: 1D convolutional layer and 1D pooling (e.g., maxpooling) layer. A 1D convolutional layer correlates a 1D kernel with the input signal to determine if the input contains a specific pattern, generating a feature vector. Given a 1D kernel $h \in \mathbb{R}^L$ and an input signal $x \in R^n$, the output of the convolutional layer can be written as

$$y_k = \sum_{l=0}^{L-1} h_l x_{sk+l} \tag{3}$$

where s is the stride length as the kernel steps across the input. Padding can be added to the beginning and end of the signal to preserve the length. By using multiple kernels at each layer, an ensemble of channels can be generated in parallel through the network to detect different features with increasing levels of abstraction. Each CNN layer can change the dimensionality by using a stride greater than one or by inserting pooling layers after each convolution operation. For example, with correct padding, a stride of two would halve the signal length by taking larger steps, while maxpooling with a kernel size and a stride of two would downsample the signal by taking the maximum value within each two-point window. The well-known successes of CNNs for image processing have justified their exploration within deep AE architectures [4,6,8,9], and there are two primary ways they can be used to construct Deep Convolutional Autoencoders (DCAE). The encoder and decoder could consist of convolution/maxpooling and upsampling/convolution, respectively [8], or the encoder could consist of convolution layers (using stride to reduce dimensionality), and the decoder could use transposed convolution (deconvolution) layers as presented in the context of Generative Adversarial Networks (GAN) [28]. When these methods are compared, convolution layers with dimensionality reduction controlled by stride (i.e., stride greater than one) can achieve better validation performance at the end of training as compared to following the convolution operation with maxpooling (see Fig. 2). This is especially relevant for signal compression, since stride-based dimensionality reduction does not discard potentially relevant information like pooling layers do. Thus, while pooling operations may prove effective in classification problems for destructively summarizing regions, the softer approach of convolution with a non-unity stride is intuitively more suited for PHM signal compression tasks.

2.3. Reconstruction correlation loss

For a deep AE or DCAE, MSE loss can only evaluate the reconstruction performance from the perspective of averaged signal behavior, and an AE with only MSE loss could easily lose local details (especially high-frequency dynamics) during the encoding process that can be critical for industrial PHM. Hence, additional loss terms could enforce the constraints on signal compression to minimize the loss of this critical information, and thereby augment the reconstruction quality.

One potential metric to improve the signal reconstruction quality and convergence rate of AEs is the addition of Pearson's Correlation Coefficient (PCC) loss [4]. For a set of samples x_k and y_k for $k = 1, 2, \ldots, n$, the PCC is defined as

$$\rho = \frac{\sum_{k=1}^{n}(x_k - \bar{x})(y_k - \bar{y})}{\sqrt{\sum_{k=1}^{n}(x_k - \bar{x})\sum_{k=1}^{n}(y_k - \bar{y})}} \tag{4}$$

Since the desired PCC is $\rho = 1$ for signal reconstruction (i.e., the input and reconstruction are exactly correlated), the PCC-derived loss term for AE becomes

$$J_{\text{PCC}} = \frac{1}{N}\sum_{i=1}^{N}\left\|1 - \rho^{(i)}\right\|_2^2 \tag{5}$$

where $\rho^{(i)}$ is the PCC of the ith input and corresponding output reconstruction of the AE.

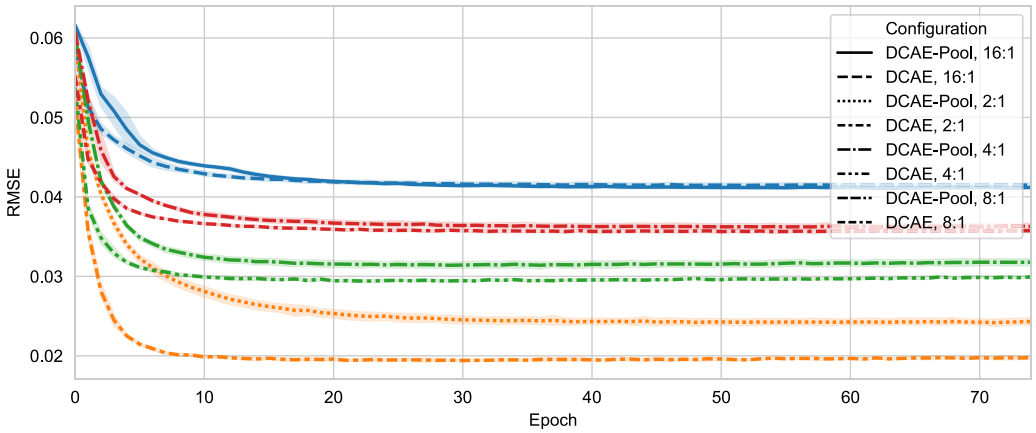

Fig. 2. Comparing validation RMSE between DCAE with stride-based dimensionality reduction and DCAE with maxpooling (DCAE-Pool). The results were collected on the Case Western Reserve University (CWRU) bearing data set using input examples of length 512 and a DCAE with four convolutional layers with 32, 64, 128, and 256 channels, respectively, and a final convolutional layer with number of channels determined by the compression rate (e.g., 128, if the rate was 4:1). The model was trained with the Adam optimizer and a learning rate of 0.001 for 75 epochs. The advantage of DCAE over DCAE-Pool decays as the compression factor increases indicating that at higher ratios the final size of the encoding dominates the reconstruction accuracy rather than the encoder architecture.

Several advantages of adding a correlation-based loss term are apparent. As indicated by the physical meaning of the metric, high correlation means that corresponding elements in the input and reconstruction at the same sample index move up and down together, tracking each other. This introduces a locality to the metric that is notably absent from MSE, which could return identical values for error concentrated at a single sample index and error evenly distributed throughout the signal. Thus, PCC provides a locally sensitive approach for comparing two signals. This is especially critical for machine condition monitoring applications where data sources include rapidly changing vibration or acoustic signals with valuable frequency information. Adding PCC explicitly guides the network to match the shape and structure (i.e., variation) of the output reconstruction with that of the input, elevating the influence of frequency content during training. In addition, [4] alluded to PCC's applicability to small magnitude signals, although the reasoning is not fully explained in their study. More precisely, this advantage stems from the consistent magnitude of PCC-based loss, since PCC will always between −1 and 1 regardless of signal amplitude. This is particularly salient within PHM, because machine data can have dramatically different amplitudes depending on the operating or fault condition. Thus, PCC as a variance-normalized metric can provide a more consistent evaluation of the reconstruction quality, and adding PCC to the loss function would augment MSE to encourage the AEs to achieve higher performance.

2.4. Physics-informed frequency-domain loss

In addition to structural loss terms like PCC, AEs operating on sensing signals (e.g., vibration) can leverage domain knowledge of the underlying physical processes. In many cases, the frequency content of the signal is crucial for identifying the machine condition [29]. Therefore, the expected time-domain reconstruction of sensing signals in the PHM field must preserve the dominant frequency content of the original signal. Including a loss term during AE training that is sensitive to frequency content introduces a physically informed objective that constrains the weight optimization space (see Fig. 3). This loss term could be indirectly sensitive to frequency, e.g., autocorrelation, or explicitly representative of the frequency domain, e.g., through the Fast Fourier Transform (FFT).

2.4.1. Autocorrelation loss

Autocorrelation (AC) is the convolution of the signal using the signal itself as the kernel. (Note that the convolutional kernel is equivalent to the flipped filter: $h_{\text{kernel}}(k) = h_{\text{filter}}(-k)$). This operation essentially searches the original signal for copies of itself at each possible time lag. Signals with large levels of white noise would have AC values near zero at lags other than zero, since the random noise will not display repeating patterns. Also, periodic signals would have large AC values as the component frequencies shift in and out of phase. Therefore, AC is an indirect, time-domain evaluation metric for highlighting periodicity. More precisely, the AC of $\boldsymbol{x} \in \mathbb{R}^n$ is defined as

$$r_{\boldsymbol{xx}}(l) = \sum_{k=l}^{n} x_k x_{k-l} \tag{6}$$

for $l = 0, 1, \ldots, n-1$. Since $\boldsymbol{x}$ is real, expanding this definition to include negative lags would produce symmetry around $l = 0$, so only $l \geq 0$ are considered. For AE training, AC loss can be expressed as

$$J_{\text{AC}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \boldsymbol{r}_{\hat{\boldsymbol{x}}\hat{\boldsymbol{x}}}^{(i)} - \boldsymbol{r}_{\boldsymbol{xx}}^{(i)} \right\|_2^2 \tag{7}$$

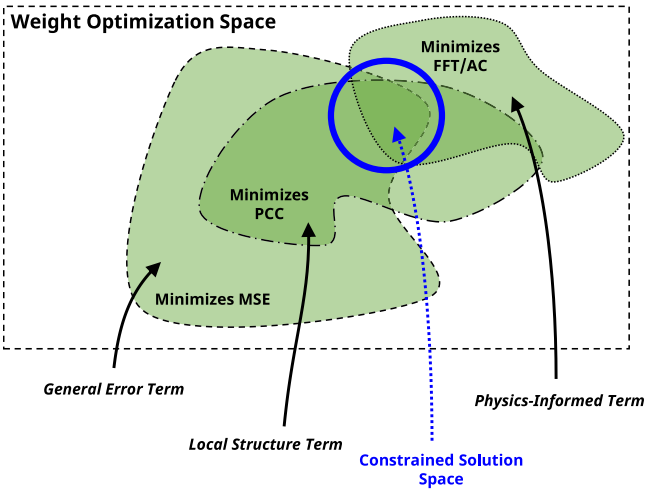

Fig. 3. Each unique loss term constrains the search space the training algorithm must explore to find good solutions. By combining multiple constraints, the solution space becomes limited to the intersection of the component solution spaces. Note that while this illustration represents solution spaces as contiguous, the high-dimensionality of the network weight space will often produce many disconnected local minima regions.

where $\boldsymbol{r}_{\hat{\boldsymbol{x}}\hat{\boldsymbol{x}}}^{(i)} \in \mathbb{R}^n$ is the vector of the AC of the ith output reconstruction for lags 0 to $n-1$, and $\boldsymbol{r}_{\boldsymbol{xx}}^{(i)} \in \mathbb{R}^n$ is the corresponding AC vector for the ith input signal. As a loss term, AC acts as a rudimentary filter that suppresses uncorrelated white noise (since the AC of white noise is zero except with zero lag) and highlights the presence of repeating patterns. By comparing the AC values of the input and reconstruction, the loss becomes less sensitive to noise content and more easily able to access the periodicity of the input signal from the time domain. In turn, this produces a metric more suited to training the AE to preserve the frequency content needed for PHM than MSE, for which frequency preservation is a side-effect of attempting to minimize error between time-domain samples.

2.4.2. Fast Fourier transform loss

Directly optimizing the AE network parameters with respect to frequency content, or in other words, explicitly preserving frequency components of interest during signal compression and reconstruction, can be accomplished by including a loss term generated in the Fourier Transform domain. While a time-domain signal and its Fourier transform are theoretically equivalent, this does not imply that they are functionally identical for the numerical optimization task of training a DL model. Practically, each domain provides a distinct perspective on the quality of the output reconstruction at each step of the optimization process, and therefore distinct weight updates. As a result, this numerical, rather than analytical, nature of training DL models suggests that having two such viewpoints on the reconstruction objective could improve the speed and quality of model convergence by modifying the loss landscape to avoid local minima or saddle points. Frequency components can be efficiently computed for use in the loss function via the Fast Fourier Transform (FFT) in a form of complex-valued outputs. One loss term based on the FFT relies on magnitude:

$$J_{\text{FFT1}} = \frac{1}{N}\sum_{i=1}^{N}\left\| \left|\hat{X}_f^{(i)}\right| - \left|X_f^{(i)}\right| \right\|_2^2 \tag{8}$$

where $X_f^{(i)}$ and $\hat{X}_f^{(i)}$ are the FFTs of the ith input signal and output reconstruction, respectively. This will encourage the reconstruction to contain frequency components with the correct magnitudes—a critical task for rotating machinery PHM, as the faulty information can be well revealed by the frequency components. While this magnitude-based approach might be sufficient to match power spectra, it is conceivable that the time-domain reconstruction could still be inaccurate—magnitude optimization fails to account for phase information that shifts the sinusoids in the time domain and localizes signal features. Most fault detection and classification applications can successfully rely on the identification of changes in magnitudes of frequency components, while phases of frequency components are unnecessary. However, phase of FFT spectra is vital for signal reconstruction tasks as studied here, as phase plays a role in determining the signal shapes and manifolds. Multiple signals can produce the same frequency amplitudes, so using only the amplitudes would reinforce an ambiguous network training objective with multiple local minima. Thus, a second version of the FFT loss term that includes the real and imaginary loss is proposed to capture both the magnitude and phase information, disambiguating the frequency-domain optimization criteria for the purpose of accurate signal reconstruction:

$$J_{\text{FFT2}} = \frac{1}{N}\sum_{i=1}^{N}\left[\left\|\Re\left[\hat{X}_f^{(i)} - X_f^{(i)}\right]\right\|_2^2 + \left\|\Im\left[\hat{X}_f^{(i)} - X_f^{(i)}\right]\right\|_2^2\right] \tag{9}$$

While the magnitude-only FFT loss only encourages similar frequency components, incorporating the phase information ensures that these components appear with the correct offsets to constructively interfere and produce the local features present in the input signal.

Both of these FFT-based methods offer frequency-domain constraints for AE network parameter optimization and are motivated by the domain-relevance of frequency analysis for industrial PHM.

2.5. Fault division autoencoder multiplexing

Despite PI loss terms, the heterogeneous machine operating conditions and sensing data patterns may also necessitate larger, architectural innovations to maintain acceptable PHM performance. Condition monitoring techniques often suffer from the Achilles heel of data-driven ML and DL: performance is dictated by the quality, quantity, and diversity of the data. If the data is homogeneous (e.g., all normal operating examples), DL methods will absorb information from the training examples and likely perform well on data from the same domain. While in many cases data diversity improves generalization, data sets with a wide variety of distinct modes (e.g., multiple faulty conditions alongside the normal condition) might lead to non-optimal condition-wise solutions. The best solution for all the data may not be the best solution for each individual condition. In an attempt to reach the lowest loss for heterogeneous fault signals with varying characteristics, the network will eventually reach a point where any further improvement for one condition is prone to diminish performance across the other conditions. In AE compression applications, this translates to reduced compression efficiency and reconstruction accuracy when a single AE must compress data from distinct conditions. Thus, while accurate reconstruction of data from different faulty conditions is critical, these faulty data might be the most difficult to reconstruct accurately using a single AE, as one network is not enough to learn multiple (maybe contradictory) patterns associated with different conditions, especially when the amount of normal training samples surpasses the number of faulty training samples [4,19].

To address this challenge, Fault Division Autoencoder Multiplexing (FDAM) is proposed to improve compression performance on data sets with these disjoint conditions. FDAM applies separate AEs for individual operating conditions and trains a fault classifier alongside them (see Alg. 1). This classifier allows FDAM to select the condition-appropriate AE to encode and decode a signal sample. During inference, the fault classifier identifies the sample's operating condition, and then encodes it with the encoder trained for that condition. The compressed code is transmitted along with the fault class, and the receiver uses a corresponding decoder to reconstruct the original signal (see Alg. 2). Fig. 1 summarizes the FDAM concept. In its simplest form, each individual AE can have the same DCAE architecture (e.g., number of layers, number of channels, etc.), but the AEs could be adapted to best suit each condition class if needed. Together with PI loss terms, FDAM seeks to improve compression performance across disjoint conditions by leveraging domain knowledge.

Algorithm 1: FDAM Training

Input: Labeled data set D with C disjoint conditions, label space $\mathcal{Y} = \{1, 2, \dots, C\}$, minibatch size m, and learning rate λ
Result: C trained AEs
Train a C-way classifier $f : D \mapsto \mathcal{Y}$;
Initialize AEs $G_i = g_{\phi_i}^{-1} \circ g_{\theta_i}$ for $i = 1, 2, \dots, C$;
for $\boldsymbol{X} \in \mathbb{R}^{m \times n}, \boldsymbol{y} \in \mathcal{Y}^m$ *in* D **do**
  for $i \in \mathcal{Y}$ **do**
    $\boldsymbol{X}_i = \{X_{j,:} \mid y_j == i\}$;
    $\hat{\boldsymbol{X}}_i = G_i(\boldsymbol{X}_i)$;
    Update G_i encoder weights: $\theta_i + \lambda \nabla_{\theta_i} J(\hat{\boldsymbol{X}}_i, \boldsymbol{X}_i) \rightarrow \theta_i$;
    Update G_i decoder weights: $\phi_i + \lambda \nabla_{\phi_i} J(\hat{\boldsymbol{X}}_i, \boldsymbol{X}_i) \rightarrow \phi_i$;
  end
end

Algorithm 2: FDAM Testing

Input: C trained AEs $G_i = g_{\phi_i}^{-1} \circ g_{\theta_i}$ for $i = 1, 2, \cdots, C$, condition classifier $f : D \mapsto Y = \{1, 2, \cdots, C\}$ and signal $\boldsymbol{x} \in \mathbb{R}^n$ to compress
Result: Reconstructed (decompressed) signal $\hat{\boldsymbol{x}}$
Classify condition: $y = f(\boldsymbol{x})$;
Encode signal: $\boldsymbol{z} = g_{\theta_y}(\boldsymbol{x})$;
Transmit $(\boldsymbol{z}, y)$ to decoder;
Recover signal: $\hat{\boldsymbol{x}} = g_{\phi_y}^{-1}(\boldsymbol{z})$;

3. Experimental case studies

The proposed DCAE and FDAM architectures augmented by physics-informed loss terms are evaluated on two PHM-related data sets.

3.1. Bearing data set

The bearing fault data set from Case Western Reserve University (CWRU) [30] is selected to compare the effectiveness of FDAM and the proposed loss functions to a single-AE baseline. While the faults within the CWRU data set are not difficult to diagnosis with DL classifiers, the high-sampling rate vibration signals accurately reflect the high-velocity Big Data that compression algorithms must handle to reduce traffic on cloud networks. Therefore, even though the CWRU data contains simple faults that make it ineffective for studying state-of-the-art fault diagnosis techniques, the presence of multiple normal and fault conditions in the 12 kHz vibration data along with varying levels of fault-related background noise make the bearing data set a suitable choice for evaluating the proposed compression methods. In the CWRU data set, accelerometer vibration data was collected while operating a 2-horsepower (HP) motor under 0-, 1-, 2- and 3-HP loads. In addition to normal operating conditions, ball faults, inner race faults, and outer race faults in the drive-end bearing were tested with 0.007″, 0.014″, and 0.021″ crack sizes. Outer race faults could occur at the 3 o'clock, 6 o'clock, and 12 o'clock positions; data from 6 o'clock faults were used in this study. With these loads, conditions, and severities, the data represents 40 unique operating configurations. Both drive-end and fan-end accelerometer data are available, and the 12 kHz fan-end data are used in this case study.

In order to split each multi-second vibration time series into input samples for the AEs, a 512-point window is slid across each time series with a step of 32 points. As a result, a total of 48 000 samples with even distribution on normal and faulty conditions (three severities each) and all four loads are generated. Thus, the entire data set is compiled from 12 000 normal samples (300 per load), 12 000 ball fault samples (100 per load per severity), 12 000 inner race fault samples (100 per load per severity), and 12 000 outer race fault samples (100 per load per severity). The entire data set is separated using an 80% training, 10% validation, and 10% testing split. A Min–Max scaling is applied to scale the training signal to the range $[-1,1]$ and the same Min–Max boundaries are applied without modification to the validation and testing data splits. These splits are cached to disk to ensure that each model is trained on the same subset of examples.

3.2. Milling data set

To provide additional insight into the proposed methods, a second case study is performed on the Milling Data Set from the Prognostics Data Repository at NASA Ames Research Center [31]. Vibration and acoustic emission data were collected during a variety of milling runs. Each run used either steel or cast iron with a feed rate of 0.5 mm/s or 0.25 mm/s and a depth of cut of 0.75 mm or 1.5 mm, giving eight unique configurations of process parameters. For the vibrations, each run consisted of 36 s of accelerometer data collected at 250 Hz. This case study uses spindle vibration signals for milling operations on cast iron with all combinations of feed rate and depth of cut. The first and last 10 s of each run were discarded to avoid transient effects. Following the same preprocessing steps as the bearing data, the signals are split into 512-point windows with a sliding step of 32 points. Each sample is zero-meaned and normalized between -1 and 1. To keep the data balanced across process parameter combinations, 1853 samples are generated for each of the four combinations of feed rate and depth of cut, for a total of 7412 examples. The splits among training, validation, and testing samples adopt the same strategy as with the bearing data.

3.3. DCAE network architecture

The baseline DCAE architecture is shown in Fig. 4. Five convolution layers are used in the encoder, each doubling the number of channels and halving the length of the signal. Rectified Linear Unit (ReLU) activations are used at each step except the final encoding output, which uses a linear activation. The decoder directly mirrors in the encoder, using convolution transpose layers to double the length of the signal while halving the number of channels. As with the encoder, the decoder uses ReLU activations at each step except the final output, which uses a tanh activation to limit the values from -1 to 1. This architecture is derived from the popular Deep Convolutional Generative Adversarial Network (DCGAN) presented in [28]. For compression applications, the DCGAN approach of convolution and convolution transpose layers is a more logical choice than using of convolution, maxpooling, and upsampling because it avoids discarding information through the maxpooling operation. Avoiding this preemptive removal of information allows a richer formulation of the encoding at the last layer of the encoder network.

3.4. FDAM network architecture

The FDAM implementation shown in Fig. 5 consists of a condition classifier and four DCAEs, corresponding to normal, ball fault, inner race fault, and outer race fault conditions in the bearing case study. Each DCAE used the same baseline DCAE architecture as depicted in Fig. 4. The condition classifier presented in Fig. 6 consists of four convolution layers with a kernel size of 3 and a stride of 1, each followed by a maxpooling layer with a stride of 2 to halve the signal length. (Maxpooling is appropriate here for the classifier—the DCAE networks retain stride-based dimensionality reduction, since they are doing compression.) The first convolutional layer generates 32 output channels, and successive convolution layers double the number of channels. The CNN output features are then passed to a 128-neuron fully-connected layer before reaching the output layer with four nodes, one for each condition class. ReLU activations are used throughout.

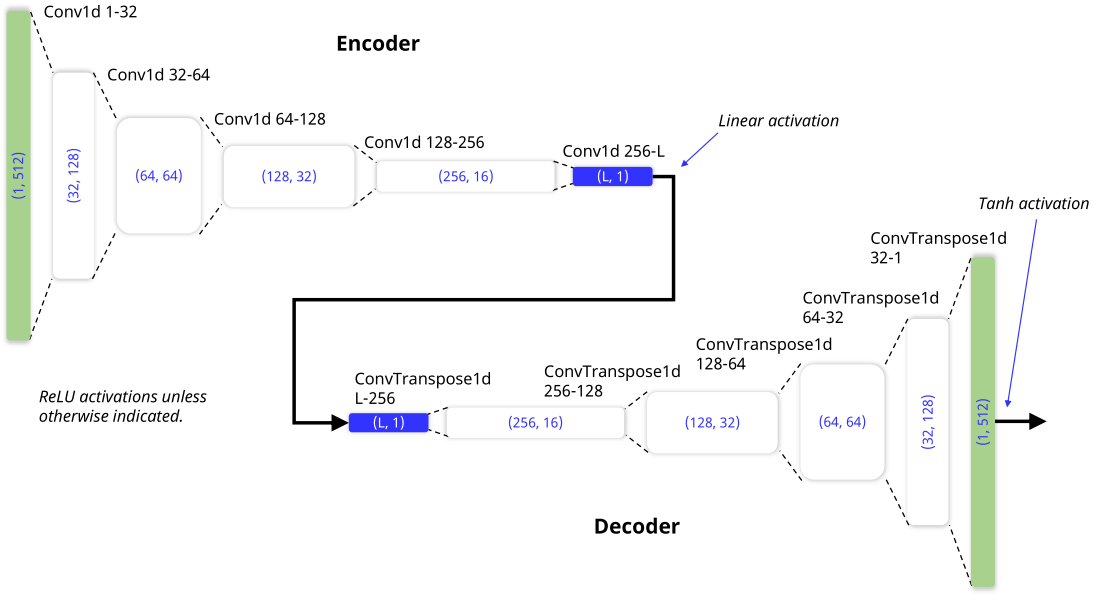

Fig. 4. Deep Convolutional Autoencoder (DCAE) used for the single-AE baseline architecture and for each condition-specific AE in the FDAM architecture. Convolution layers with a stride of 2 were used instead of pairing stride-1 layers with maxpooling layers which would have discarded information.

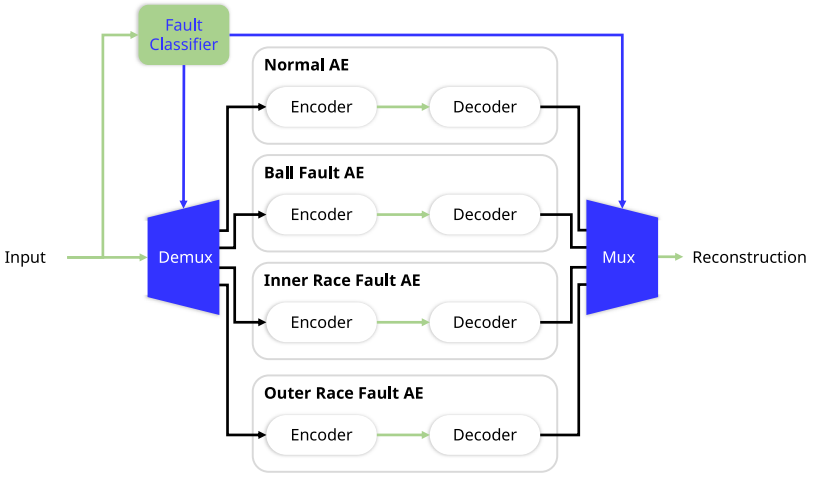

Fig. 5. FDAM architecture consisting of a classifier and multiplexed fault-specific DCAEs. Each DCAE (encoder–decoder pair) follows the architecture shown in Fig. 4. This implementation is equivalent to the concept shown in Fig. 1 but without transmission channel modeling.

3.5. Training configurations

All models are implemented using PyTorch [32] and PyTorch Lightning [33] with rich data capture via Neptune [34]. Network training runs are accomplished on a Linux computer with a 2.1 GHz, 16-core Intel Xeon Gold 6130 CPU with 192 GB of RAM and an NVIDIA Tesla V100 GPU with 24 GB of RAM. Consistently using 16-bit tensors enables the trainer to leverage the native 16-bit precision of the GPU. Every configuration (i.e., every model/compression rate combination) is repeated five times to broadly gauge variability. The random seed is changed in each trial to randomly shuffle the samples into batches and initialize weights, while maintaining a deterministic choice of underlying cudaNN algorithms.

For the bearing data set, the fault classifier is trained once and reused for all the FDAM-based models to compute validation metrics. After 50 epochs with a learning rate of 0.0001 and the Adam optimizer, it achieves an accuracy of 98.25% on the test data set. While this study does not attempt to identify an optimal compression rate, which would be application-dependent due to other constraints, each architecture and loss function is tested with a variety of compression rates to characterize relative performance as the size of the latent vector is reduced. To this end, the experiments considered compression rates of 2:1, 4:1, 8:1, 16:1, and 32:1

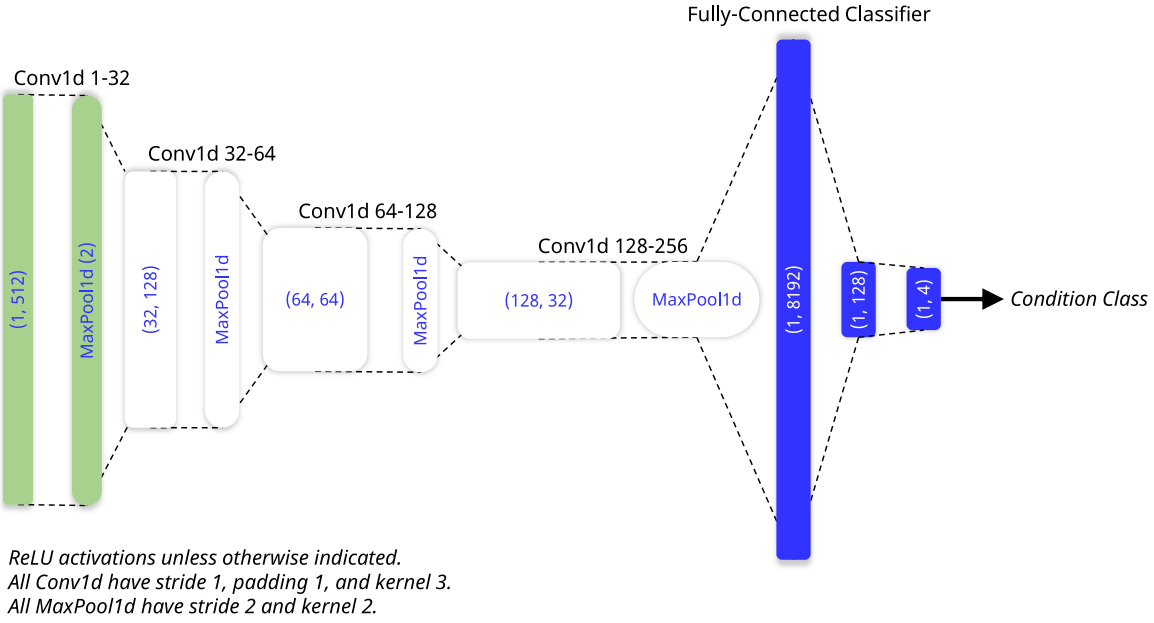

Fig. 6. Architecture of FDAM fault condition classifier.

which correspond to latent vector lengths of 256, 128, 64, 32, and 16 given the input data length of 512. Nine different variations of the single-AE DCAE and multi-AE FDAM architectures with different loss functions are tested: DCAE, DCAE-C, DCAE-C-AC, DCAE-C-FFT1, DCAE-C-FFT2, FDAM-C, FDAM-C-AC, FDAM-C-FFT1, and FDAM-C-FFT2. In these identifiers, C, AC, FFT1, and FFT2 represent the inclusion of PCC (Eq. (5)), AC (Eq. (7)), FFT magnitude (Eq. (8)), and FFT real and imaginary loss (Eq. (9)), respectively. While the baseline DCAE model includes only MSE loss, the corresponding complete loss functions for the additional loss terms are:

$$J_{\text{C}} = J_{\text{MSE}} + J_{\text{PCC}} \tag{10a}$$

$$J_{\text{C-AC}} = J_{\text{MSE}} + J_{\text{PCC}} + J_{\text{AC}} \tag{10b}$$

$$J_{\text{C-FFT1}} = J_{\text{MSE}} + J_{\text{PCC}} + \lambda_1 J_{\text{FFT1}} \tag{10c}$$

$$J_{\text{C-FFT2}} = J_{\text{MSE}} + J_{\text{PCC}} + \lambda_2 J_{\text{FFT2}} \tag{10d}$$

Each loss function seeks to leverage different perspectives on the signal to improve reconstruction. Eq. (10a) adds PCC loss to the standard MSE metric to leverage PCC's sensitivity to signal structure and invariance across signal amplitudes. From a time-domain perspective, AC loss in Eq. (10b) extends Eq. (10a) with information about signal periodicity to guide preservation of frequency content. Both Eqs. (10c) and (10d) opt for a more direct frequency domain objective that supplements the time-domain MSE and PCC terms and facilitate numerical optimization, with the FFT2 term in Eq. (10d) incorporating magnitude and phase information for an unambiguous quantification of frequency reconstruction fidelity. Upon evaluating the validation metrics, appropriate scaling values are found to be $\lambda_1 = 0.001$ and $\lambda_2 = 0.1$. Each model is trained five times for 75 epochs with the Adam optimizer and using a learning rate of 0.001.

Since the milling data set does not contain normal and faulty conditions, only the baseline DCAE architecture is evaluated. Five different variations of DCAE are trained: DCAE, DCAE-C, DCAE-C-AC, DCAE-C-FFT1, and DCAE-C-FFT2. Each variation is trained five times for 50 epochs with the Adam optimizer and using a learning rate of 0.002.

4. Results

The results for each case study were evaluated using the Signal-to-Noise Ratio (SNR) in decibels (dB). In the context of compression, the noise is not the noise in the original signal, but the noise introduced by the compression process and quantified in terms of the reconstruction error. Thus, the calculation becomes

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \left(\frac{\sigma_{\boldsymbol{x}}^2}{\sigma_e^2} \right) \tag{11}$$

where $\sigma_{\boldsymbol{x}}^2$ is the power of the input signal, and σ_e^2 is the power of the residual error $\hat{\boldsymbol{x}} - \boldsymbol{x}$. This metric contextualizes the compression performance by accounting for the amplitude of the original signal when calculating the distortion caused by the compression process.

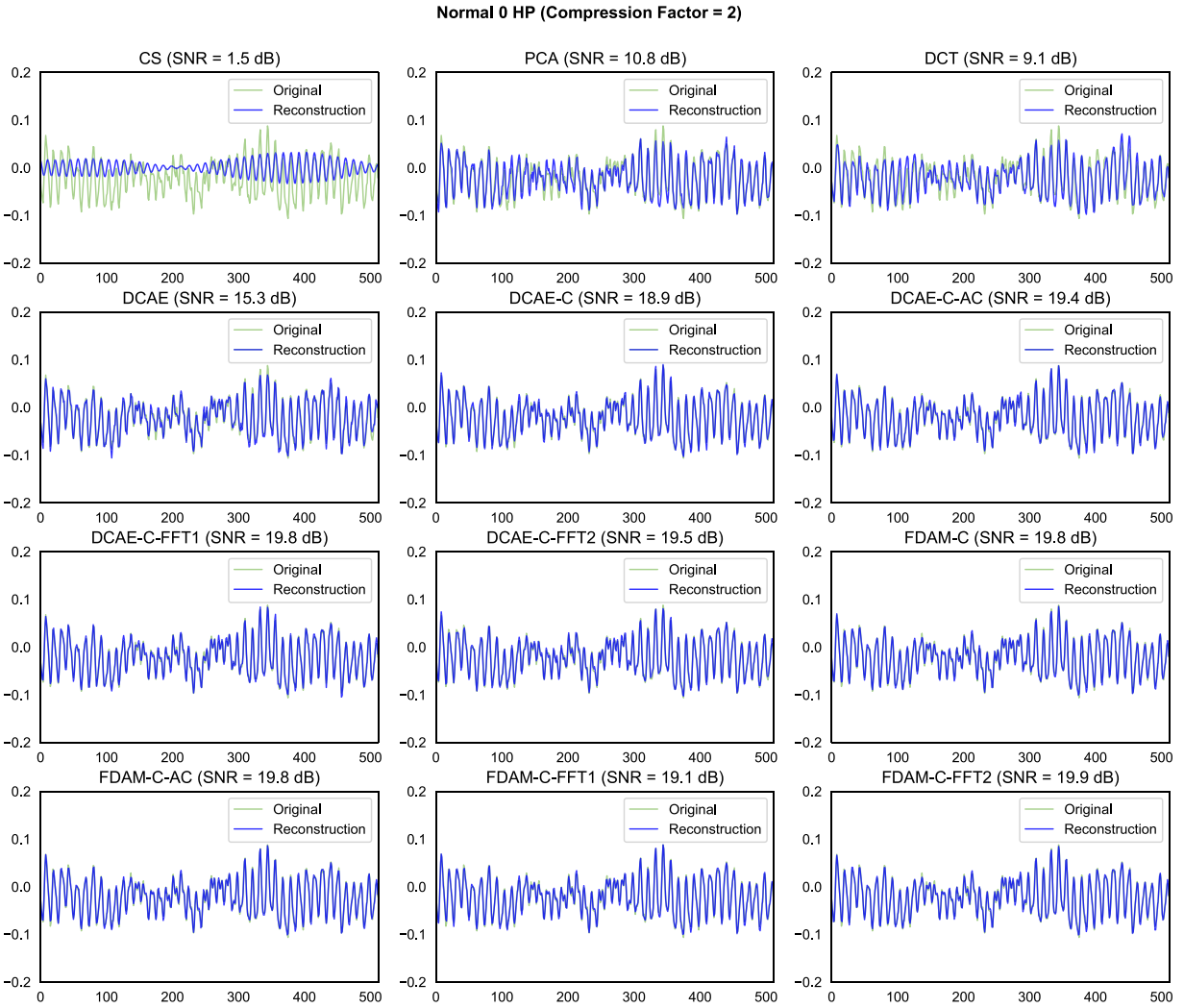

Fig. 7. Reconstructions of normal bearing data with a 0 HP load with a 2:1 compression ratio.

Compressed Sensing (CS) [35], Principal Component Analysis (PCA) [36], and Discrete Cosine Transform (DCT) [37] are common techniques related to compression, and are hence used as non-DL reference methods for performance comparison. CS is implemented by randomly selecting sampling indices to achieve the desired compression factor while using the DCT as the sparse transform domain [38]. Since CS-based compression process is subject to randomness, ten different sets of sampling indices are tried for each compression rate. For PCA, the number of output components is chosen according to the compression factor, and the model is fit to the training data set. The DCT-based approach computes the transform coefficients across the training set, averages the coefficient magnitudes, and finds the positions of the largest coefficients to achieve the desired compression rate. Test data is compressed using these coefficients; the rest are zeroed out. This method potentially could be improved by adaptively selecting coefficients for each example; however, this would necessitate either increasing the amount of data transmitted or stored to include the locations of the coefficients or reducing the number of coefficients to keep the same compression ratio. Together, these three methods provide a way to determine if the DL methods had any advantage over traditional signal processing techniques in both the bearing and milling case studies.

4.1. Bearing case study

For the bearing case study, 285 model instances are created, including five trials of nine DL configurations across the five compression factors (225 DL networks), PCA across the five factors, DCT across the five factors, and 10 trials of CS across five factors (50 CS instances). Figs. 7–10 present examples of signal reconstructions in various operating conditions for 2:1 and 32:1 compression ratios. Even without quantitative metrics, CS clearly produces the lowest quality reconstructions. PCA and DCT generate relatively

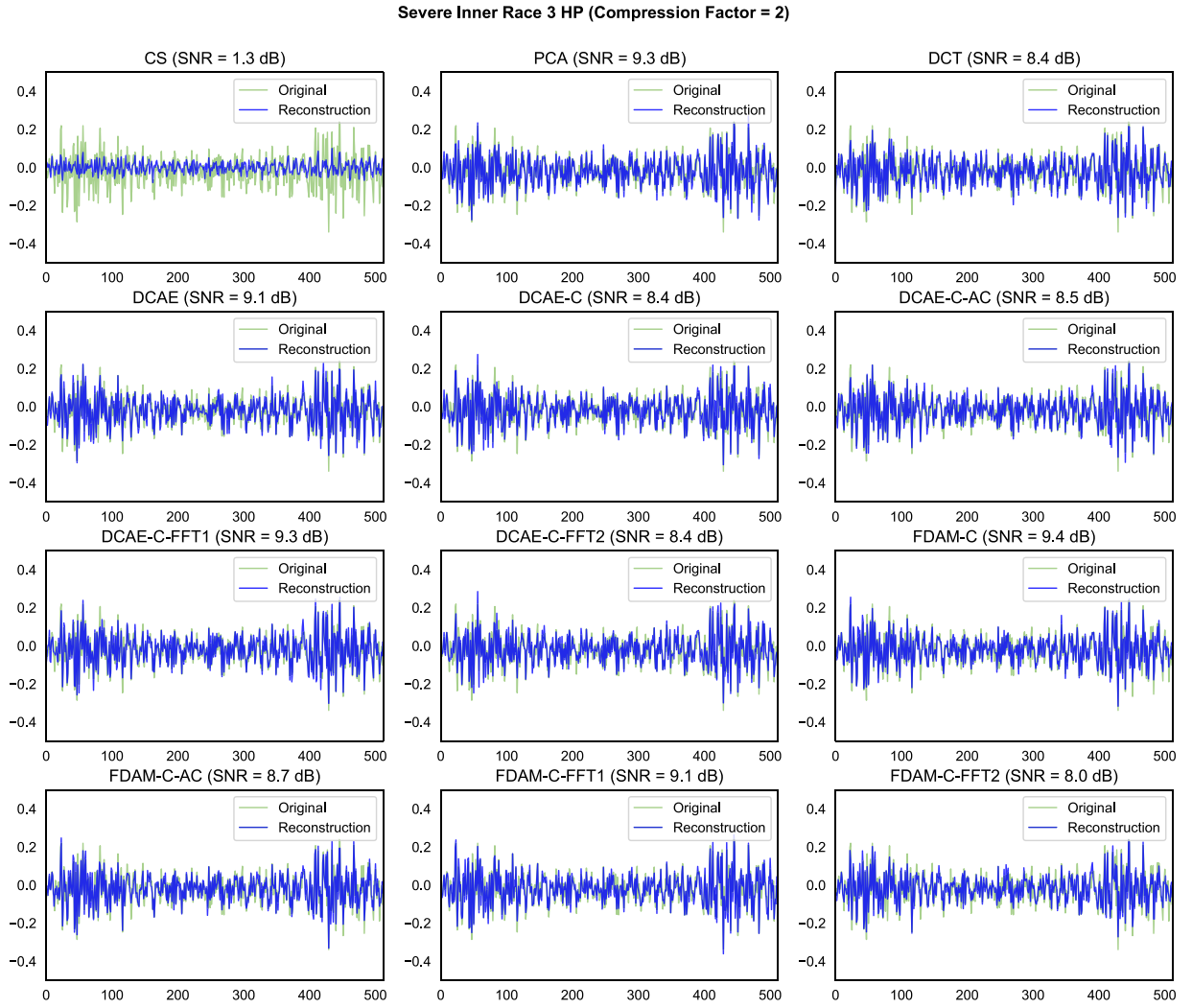

Fig. 8. Reconstructions of inner race fault bearing data with a 3 HP load with a 2:1 compression ratio.

good visual approximations to the signals at a lower, 2:1, compression ratio (Figs. 7 and 8), but demonstrate significantly degraded reconstructions at the 32:1 ratio (Figs. 9 and 10). The SNR vs. compression factor plot in Fig. 11 supports this conclusion, and Tables 2–5 present these results numerically. As the compression rate increased, linear approaches like PCA and DCT struggle to effectively encode the necessary information in the latent representation, observing a 81% and 82% decrease in SNR dB, respectively, when moving from 2:1 to 32:1 compression. This performance drop is consistent across all the operating conditions as shown in Fig. 11. Thus, top-performing non-DL techniques appear inadequate for high compression rates and significantly distort the original signal.

4.1.1. Performance of baseline DCAE model

As expected, the standard DCAE model which uses only MSE loss demonstrates advantages over the non-DL methods. The reconstructions are visually closer to the original signal at the 2:1 compression rate than CS, PCA, and DCT (see Fig. 7), and the improvement is confirmed by an SNR dB improvement of 10% over the top non-DL PCA model across the four fault conditions. The visual superiority of DCAE over CS, PCA, and DCT when compressing normal signals at 32:1 is also apparent in Fig. 9. While continuing to perform well relative to the non-DL models at high compression rates (e.g., 20% better than PCA at 32:1), DCAE struggles to consistently reconstruct faulty signals with high fidelity at these higher rates (see Fig. 10). The pattern is supported by Fig. 12, which shows that DCAE produces a lower SNR than PCA and/or DCT for faulty signals across most compression rates. This poor performance on faulty signals impacts the overall SNR for increasing compression rates, and Fig. 11 reveals that DCAE falls much closer to the performance of the non-DL models than the proposed techniques. Furthermore, the DCAE results exhibited higher variability than other methods, especially at the 8:1 compression ratio, as indicated by the standard deviations in Tables 2–5

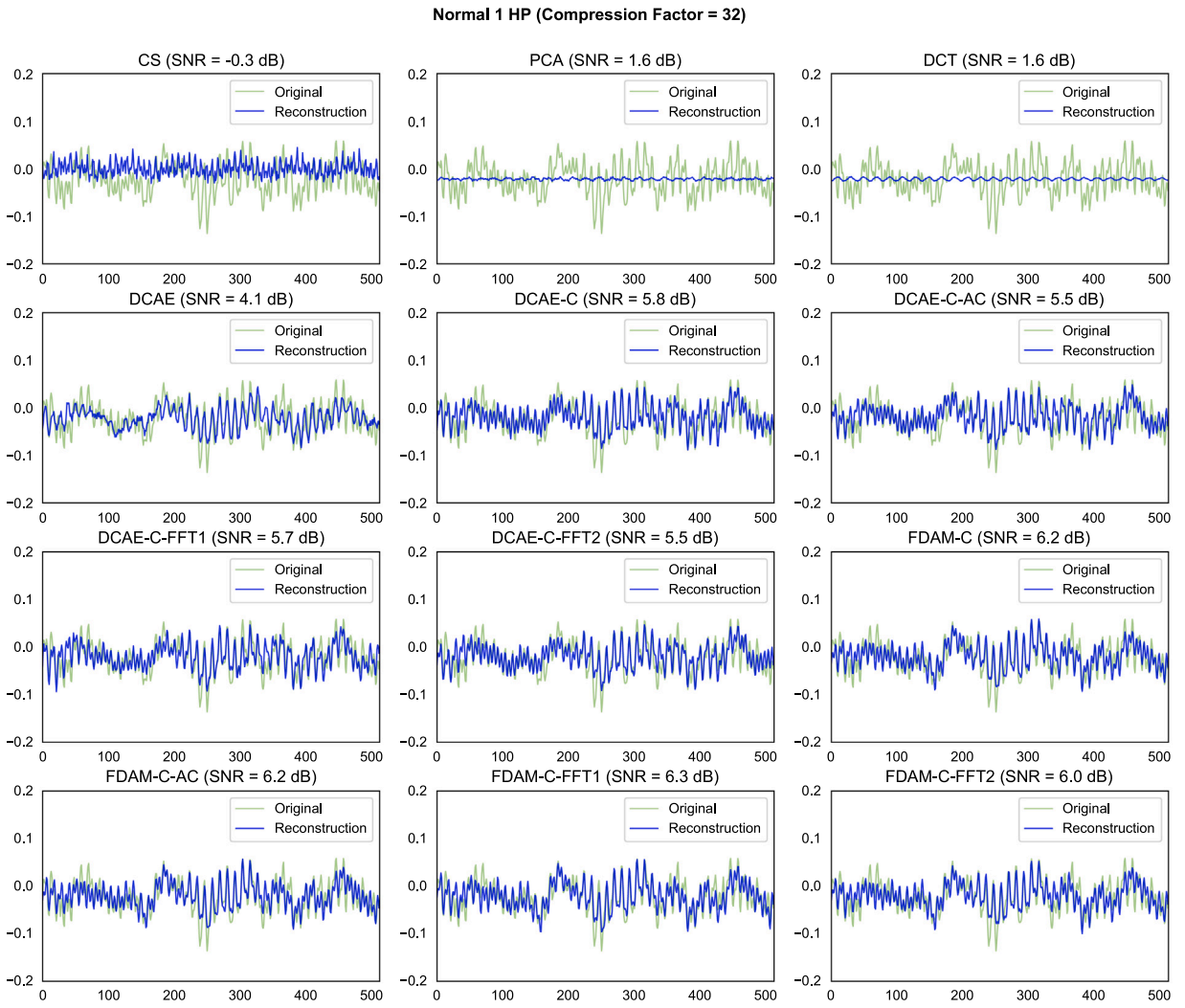

Fig. 9. Reconstructions of normal bearing data with a 1 HP load with a 32:1 compression ratio.

and the error bars in Fig. 12. While a lower learning rate and more epochs may have resolved this issue, it implies that for a given set of hyperparameters, DCAE training may be less stable or reproducible than alternative methods. This performance dropoff and variability suggests that the standard, MSE-based DCAE model is not an obvious choice for PHM compression applications.

4.1.2. Performance impacts of PCC loss

The first of the proposed loss functions, the PCC loss term substantially improves the reconstructions across all operating conditions, as the DCAE-C model results indicate in Fig. 12. DCAE-C produces consistently higher SNR than the baseline DCAE—reaching at 84% higher SNR at the 8:1 and 16:1 ratios—while providing 77%, 47%, 25%, and 230% condition-wise improvements at the 32:1 ratio across normal, ball faults, inner race faults, and outer race faults, respectively. Furthermore, the variation of the DCAE-C results is consistently smaller than the baseline DCAE, proving better model stability. The results can be attributed to the structural focus of PCC, which provides a second perspective (in addition to averaged reconstruction quality evaluated by MSE loss) and further constrains network parameter optimization during training. Notably, while DCAE underperforms non-DL approaches for high compression rates and faulty signals, DCAE-C continues to surpass the non-DL methods in these scenarios, an outcome consistent with the observation that PCC may be better when signals could differ in amplitude, as is the case between normal and faulty signals. Thus, the proposed PCC loss term quantifiably enhances the signal reconstructions over the MSE-only DCAE model, and these gains trend upward as the compression rate increases.

4.1.3. Performance impacts of FDAM architecture

The proposed FDAM model architecture also achieves higher levels of SNR than the baseline DCAE model in all cases and proves capable of further increasing the performance over the DCAE-C method. At the 4:1 compression rate, FDAM-C outperforms the

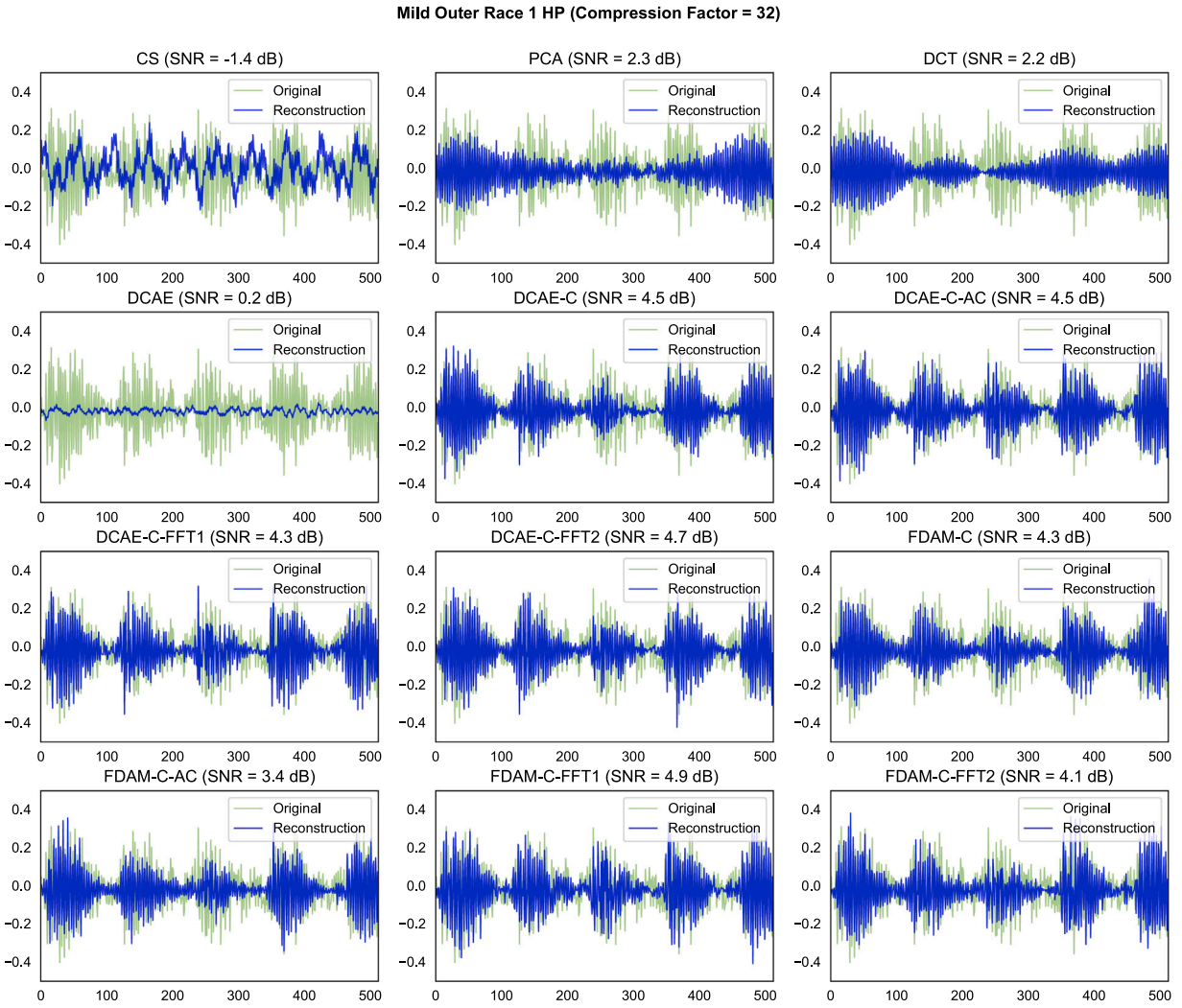

Fig. 10. Reconstructions of outer race fault bearing data with a 1 HP load with a 32:1 compression ratio.

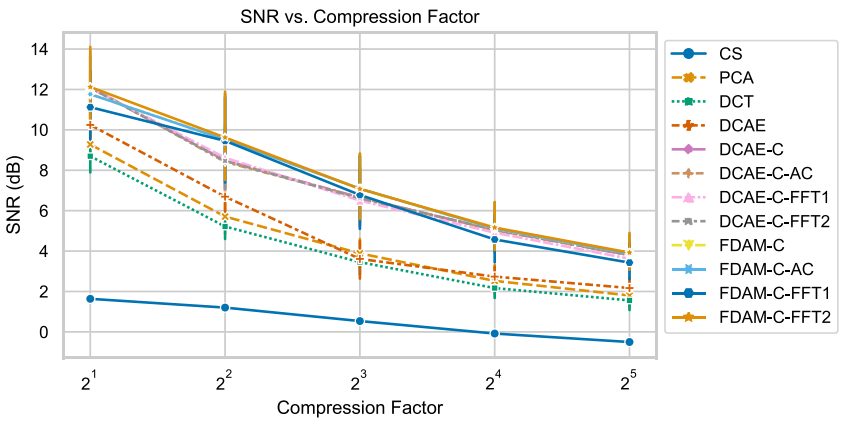

Fig. 11. Comparing reconstruction SNR across all five compression factors for the bearing data set.

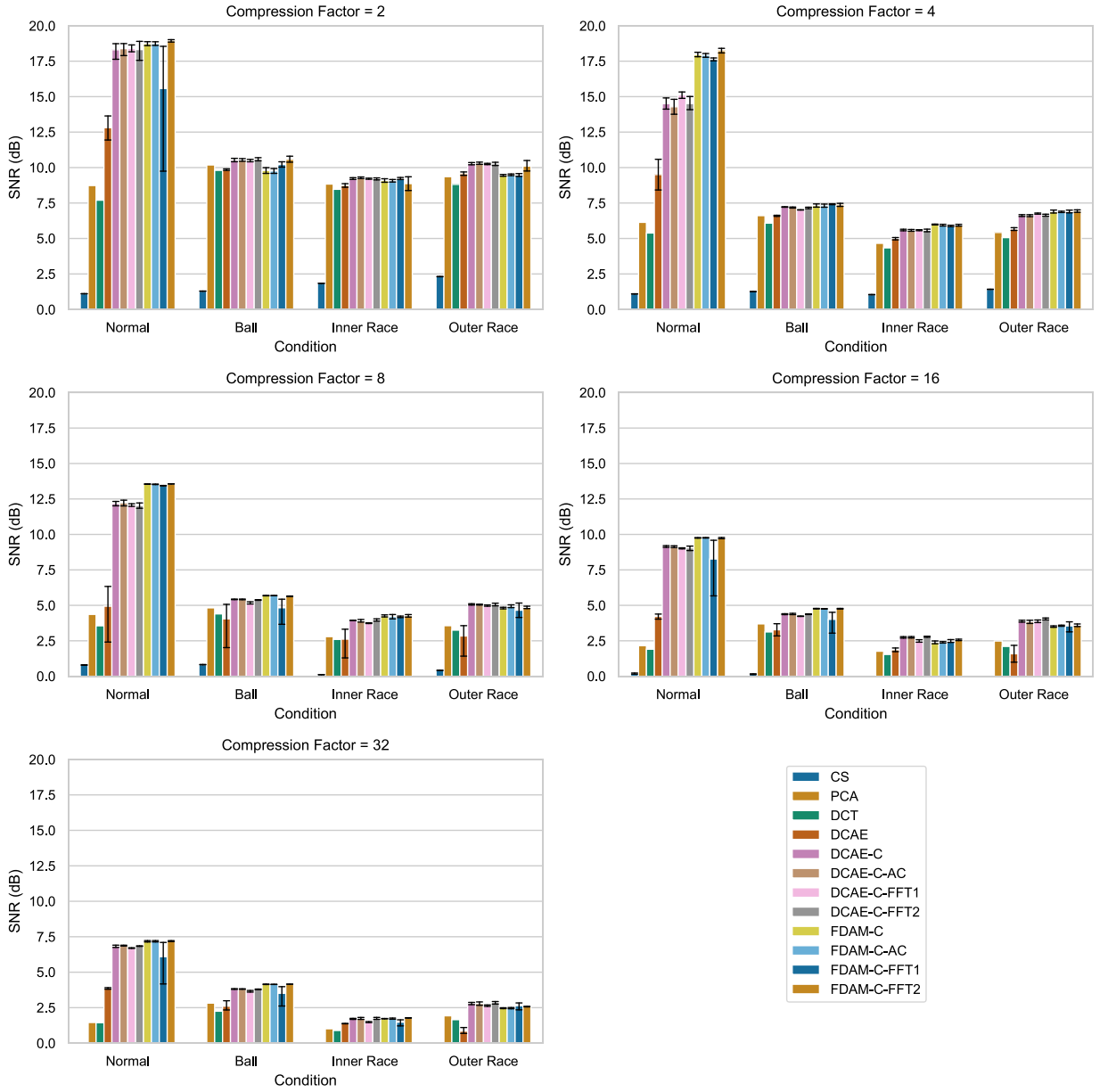

Fig. 12. Class-wise reconstruction SNR across all five compression factors for the bearing data set.

baseline DCAE by 43%, 16 points higher than the advantage yielded by DCAE-C. Similarly, FDAM-C reaches almost double the SNR of DCAE when compressing signals at 8:1, and that 96% jump is a 12-point improvement over DCAE-C. As the compression rate continues to increase, the impact of FDAM-C becomes particularly apparent with respect to reconstructing normal signals—when using a 16:1 rate, the SNR increases by 131% over the baseline DCAE, 14 percentage points higher than the 117% increase in SNR given by DCAE-C, and the 32:1 rate places FDAM-C a comfortable nine points above the DCAE-C model. From a compression ratio standpoint, FDAM-C allows signals to be compressed with the same fidelity as DCAE, but with half the required signal length or less (e.g., compressing the vibration signals with FDAM-C at 32:1 is comparable to compressing them with DCAE at 8:1.) Fig. 12 reflects this discussion, with the FDAM-C models appreciably exceeding the DCAE-C model, particularly for the 4:1 compression factor. This behavior matches the theoretical motivations of FDAM. By using separate AEs for each operating condition, FDAM learns to properly reconstruct each discrete condition without needing to simultaneously learn information only relevant to the other signal types. The original signals in Figs. 8–10 clearly show that the difference between normal and faulty signals (see Figs. 9 vs. 10) is greater than the differences among the three faulty signals (e.g., see Figs. 8 vs. 10). Therefore, the reconstruction of normal signals would stand

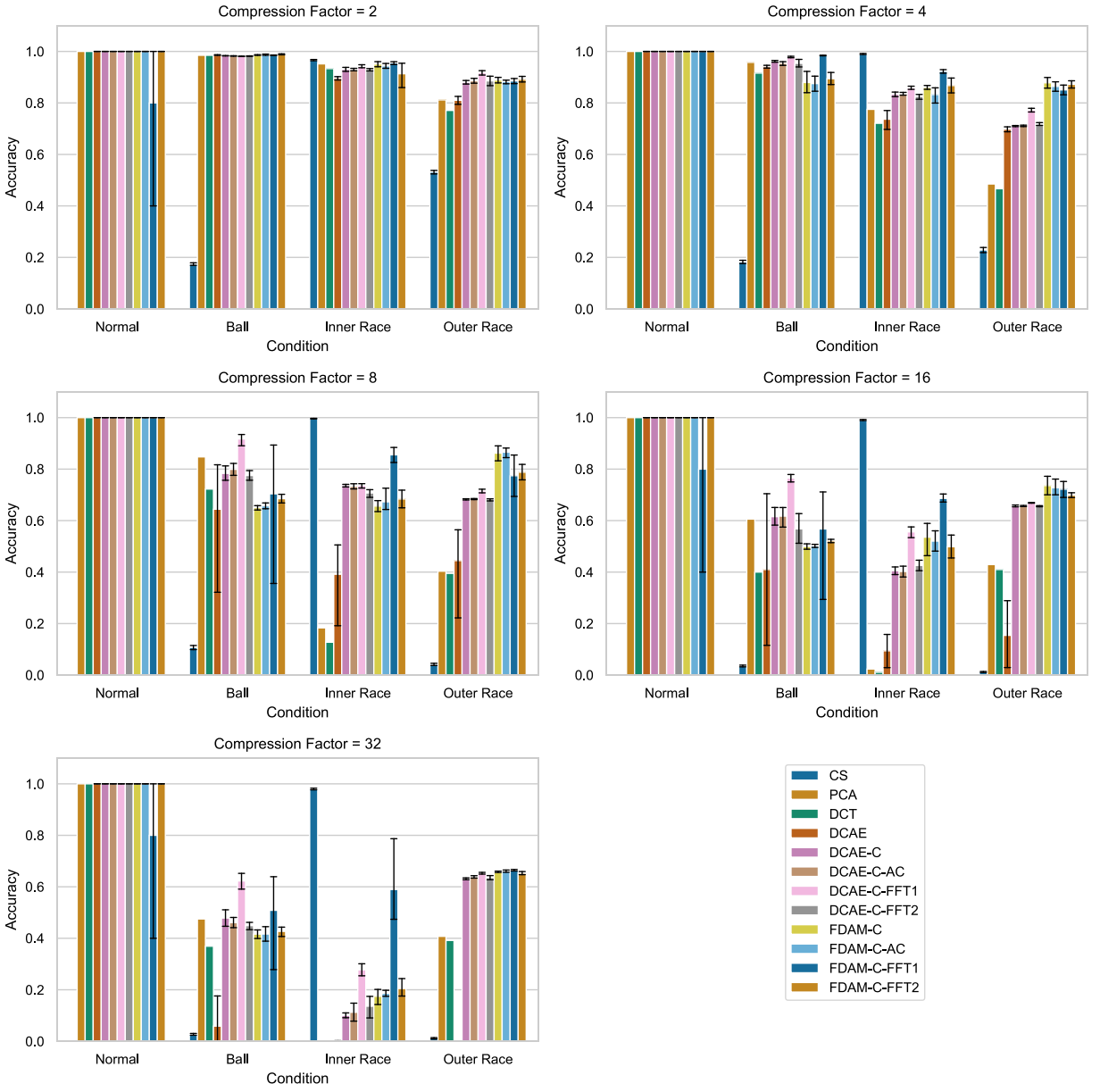

Fig. 13. Post-reconstruction bearing fault classification accuracy. CS achieves high accuracy on inner race faults and low accuracy elsewhere simply because it generated a degenerate representation that exclusively looks like this single fault to the classifier.

the most to gain from FDAM, since this would be the most different task. The experimental results confirm this reasoning and indicate the measurable advantages of the proposed FDAM architecture.

4.1.4. Performance impacts of additional PI loss terms

When combined with PCC, the proposed PI frequency-focused loss terms also demonstrate positive impacts on the reconstruction fidelity for DCAE and FDAM, particularly through the FFT2 term with real and imaginary loss. The addition of AC loss to create DCAE-C-AC and FDAM-C-AC does not produce significant changes in SNR over DCAE-C and FDAM-C, respectively (see Tables 2–5), with improvement over the baseline holding at 75% to 85% for compression factors of 8:1 and up. One plausible explanation is that the MSE and PCC loss terms already provided similar time-domain-based information to the optimization algorithm, so AC does not generate a novel enough perspective to noticeably alter the reconstructions. Interestingly, while AC simply falls short of increasing SNR, FFT1 (magnitude-only loss) tends to both reduce the reconstruction SNR and increase variability, as evidenced by Fig. 12. In fact, the addition of the FFT1 term to FDAM-C at 32:1 drops the SNR increase over the baseline DCAE from 79% to 58%. This

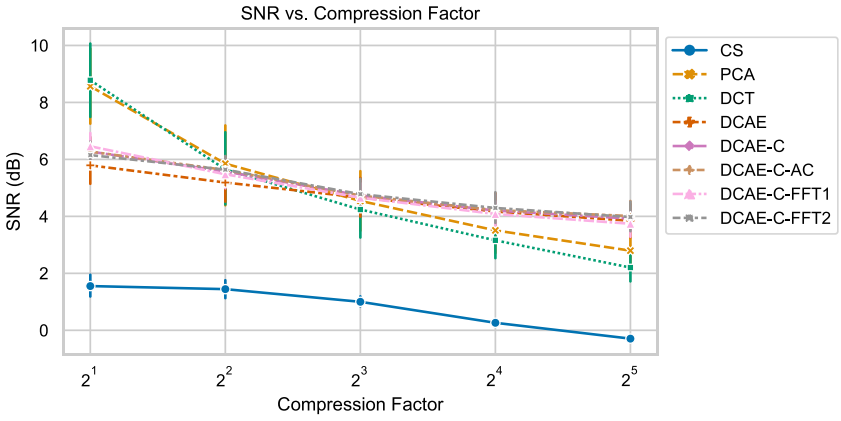

Fig. 14. Comparison of SNR across compression factors for the milling data set.

is consistent with the understanding that FFT1's lack of phase information results in an ambiguous optimization objective when attempting to reconstruct input signals. Moreover, rather than being negligible, this property actively hinders the model's ability to converge to a good solution, leading to decreased SNR and more inconsistent results. If this reasoning is correct, models using FFT2 would be expected to outperform those using FFT1, since the FFT2 term includes magnitude and phase via the real and imaginary parts of the FFT, and this is what the results illustrate. In many configurations, DCAE-C-FFT2 and FDAM-C-FFT2 surpass the SNR of DCAE-C-FFT1 and FDAM-C-FFT1, and FDAM-C-FFT2 provides the most consistent and significant advantages, specifically in the reconstruction of normal signals (see Table 2). FDAM-C-FFT2 yielded the highest SNR for normal signal reconstruction across all tested models for 2:1, 4:1, 8:1, and 32:1 compression ratios. Overall, FDAM-C-FFT2 performs 89% better than the baseline DCAE model when signals are compressed at 16:1, two percentage points above FDAM-C and 20 points beyond FDAM-C-FFT1. Similar differences appear at the 32:1 ratio, with FDAM-C-FFT2 achieving 81% higher SNR than the baseline, once again two points higher than FDAM-C and more than 20 points higher than FDAM-C-FFT1. Therefore, the proposed FFT2 loss term separates itself as a useful addition when reconstructing PHM vibration signals.

4.1.5. Post-reconstruction classification accuracy

In practical applications, the usefulness of the reconstructed machine signals will often be determined by how well they can be used to assess machine health and condition. Therefore, the operating condition of reconstructions should be identifiable. Fig. 13 shows the accuracy of the fault classifier when applied to reconstructions of the signals from each operating condition under the increasing compression factors. Even at high compression rates, the classifier can identify normal signals with near 100% accuracy. Most DCAE and FDAM models with advanced loss functions outperform the non-DL methods and the baseline DCAE model. In particular, the DCAE-C-FFT1 model distinguishes itself by consistently having higher post-reconstruction accuracy than DCAE-C that does not frequency-informed loss terms. Its superiority over DCAE-C-FFT2, which outperforms FFT1 in the reconstruction task, can be attributed to the change in task, which is now classification. Its FDAM counterpart, FDAM-C-FFT1, demonstrates some instability as it is the only DL model to struggle to produce reconstructions preserving the condition information. Among the non-DL methods, CS appears to produce reconstructions that are effective for the inner race fault condition, but the extremely poor performance elsewhere would indicate that CS in fact only generates signals that look like inner race faults, regardless of the actual input signal condition. While most methods only result in classification accuracies between 40% and 70% among the fault conditions at a compression ratio over 4:1, the results seem to indicate that the classifier is able to correctly distinguish between normal and faulty conditions when compressing and reconstructing the signal with the DCAE and FDAM models with advanced loss functions, although this could be susceptible to false negatives. Moreover, an informed choice of a PI loss term, e.g., adding FFT1 to DCAE-C in this case, could make a substantial positive impact on the post-reconstruction classification accuracy, a vital part of the ability to monitor the machine condition from compressed data.

4.2. Milling case study

To evaluate the algorithms over the milling data set, 185 model instances are created from five trials of five DCAE configurations across the five compression factors (125 DCAE instances), PCA across the five factors, DCT across the five factors, and 10 trials of CS across five factors (50 CS instances). Results show better performance for PCA and DCT at low compression factors, but demonstrate the advantage of DCAE-based compression at higher ratios (see Fig. 14 and Table 6). DCT and PCA achieve a ~35% increase in SNR over the DL methods at the lowest compression rate of 2:1. However, this advantage quickly drops to around 4% at the 4:1 rate. At the highest rate of 32:1, DCAE-C, DCAE-C-AC, and DCAE-C-FFT2 outperform PCA by over 40% in SNR. A possible reason why PCA and DCT sustain excellent performance at low compression rates is that the spindle vibration signal shows relatively uniform patterns across different combinations of feed rate and depth of cut, compared to the vibration signal from different bearing conditions. Thus

Table 2
Bearing data set model performance over normal operating condition in SNR dB.

Condition = Normal					
Model	Compression factor				
	2	4	8	16	32
CS	1.11 ± 0.01	1.09 ± 0.02	0.79 ± 0.03	0.19 ± 0.07	−0.31 ± 0.04
PCA	8.73	6.14	4.36	2.15	1.45
DCT	7.70	5.39	3.56	1.91	1.44
DCAE	12.81 ± 1.09	9.51 ± 1.40	4.94 ± 2.77	4.22 ± 0.25	3.86 ± 0.07
DCAE-C	18.31 ± 0.73	14.51 ± 0.48	12.17 ± 0.19	9.16 ± 0.08	6.84 ± 0.10
DCAE-C-AC	18.38 ± 0.54	14.29 ± 0.67	12.23 ± 0.26	9.15 ± 0.08	6.87 ± 0.03
DCAE-C-FFT1	18.41 ± 0.32	15.12 ± 0.28	12.08 ± 0.11	9.02 ± 0.05	6.70 ± 0.03
DCAE-C-FFT2	18.32 ± 0.87	14.51 ± 0.59	12.03 ± 0.24	9.05 ± 0.21	6.84 ± 0.03
FDAM-C	18.74 ± 0.18	17.96 ± 0.20	13.55 ± 0.02	9.76 ± 0.03	7.18 ± 0.06
FDAM-C-AC	18.74 ± 0.17	17.92 ± 0.18	13.53 ± 0.02	**9.76 ± 0.02**	7.18 ± 0.06
FDAM-C-FFT1	15.57 ± 6.50	17.62 ± 0.13	13.44 ± 0.02	8.27 ± 2.90	6.09 ± 2.16
FDAM-C-FFT2	**18.93 ± 0.11**	**18.24 ± 0.22**	**13.56 ± 0.01**	9.74 ± 0.06	**7.19 ± 0.04**

Table 3
Bearing data set model performance over ball fault condition in SNR dB.

Condition = Ball fault					
Model	Compression factor				
	2	4	8	16	32
CS	1.29 ± 0.01	1.26 ± 0.01	0.82 ± 0.02	0.15 ± 0.05	−0.36 ± 0.04
PCA	10.18	6.61	4.82	3.69	2.82
DCT	9.80	6.08	4.40	3.12	2.25
DCAE	9.86 ± 0.07	6.61 ± 0.05	4.05 ± 2.26	3.27 ± 0.58	2.60 ± 0.46
DCAE-C	10.52 ± 0.15	7.22 ± 0.02	5.42 ± 0.03	4.38 ± 0.04	3.81 ± 0.03
DCAE-C-AC	10.53 ± 0.12	7.18 ± 0.05	5.42 ± 0.03	4.39 ± 0.06	3.81 ± 0.03
DCAE-C-FFT1	10.49 ± 0.11	7.02 ± 0.01	5.17 ± 0.10	4.23 ± 0.02	3.66 ± 0.07
DCAE-C-FFT2	**10.59 ± 0.14**	7.16 ± 0.07	5.38 ± 0.03	4.37 ± 0.03	3.79 ± 0.03
FDAM-C	9.80 ± 0.24	7.32 ± 0.15	5.69 ± 0.02	**4.76 ± 0.03**	4.15 ± 0.02
FDAM-C-AC	9.76 ± 0.22	7.31 ± 0.14	**5.69 ± 0.01**	4.75 ± 0.02	4.14 ± 0.01
FDAM-C-FFT1	10.21 ± 0.25	7.42 ± 0.04	4.83 ± 1.28	4.01 ± 1.09	3.51 ± 1.00
FDAM-C-FFT2	10.58 ± 0.27	**7.37 ± 0.14**	5.63 ± 0.03	**4.76 ± 0.03**	**4.16 ± 0.00**

Table 4
Bearing data set model performance over inner race fault condition in SNR dB.

Condition = Inner race fault					
Model	Compression factor				
	2	4	8	16	32
CS	1.84 ± 0.01	1.04 ± 0.02	0.11 ± 0.01	−0.41 ± 0.01	−0.67 ± 0.02
PCA	8.84	4.66	2.79	1.77	1.01
DCT	8.47	4.34	2.60	1.54	0.89
DCAE	8.74 ± 0.16	5.00 ± 0.10	2.62 ± 1.47	1.86 ± 0.17	1.37 ± 0.02
DCAE-C	9.23 ± 0.08	5.60 ± 0.07	3.93 ± 0.02	2.74 ± 0.07	1.71 ± 0.05
DCAE-C-AC	**9.28 ± 0.07**	5.57 ± 0.08	3.91 ± 0.11	2.74 ± 0.06	1.74 ± 0.09
DCAE-C-FFT1	9.20 ± 0.05	5.58 ± 0.04	3.75 ± 0.04	2.49 ± 0.11	1.48 ± 0.05
DCAE-C-FFT2	9.20 ± 0.09	5.56 ± 0.11	3.97 ± 0.10	**2.79 ± 0.04**	1.75 ± 0.09
FDAM-C	9.09 ± 0.17	**5.99 ± 0.04**	**4.26 ± 0.10**	2.38 ± 0.12	1.72 ± 0.02
FDAM-C-AC	9.08 ± 0.12	5.95 ± 0.08	4.21 ± 0.19	2.39 ± 0.06	1.72 ± 0.05
FDAM-C-FFT1	9.23 ± 0.09	5.88 ± 0.06	4.19 ± 0.07	2.47 ± 0.13	1.47 ± 0.30
FDAM-C-FFT2	8.87 ± 0.65	5.94 ± 0.08	4.26 ± 0.11	2.56 ± 0.07	**1.78 ± 0.02**

at low compression rates, the linear ML techniques can easily learn a way to preserve critical signal characteristics, while the DCAE methods might complicate the compression process. However, at high compression rates, linear ML techniques would lose too many signal details to accurately reconstruct the original signal, while the DCAE methods can preserve more details through a nonlinear transformation approach.

While not showing as dramatic of increases as on the bearing data set, DCAE-C increases the SNR over the baseline DCAE by 8.3%, 7.3%, 1.5%, 0.71%, and 2.1%, at the five compression rates, respectively. As with the bearing data, DCAE-C-FFT1 falls on the lower performance end of the DCAE spectrum. DCAE-C-AC and DCAE-C-FFT2 slightly outperform the DCAE-C, starting from of compression factor of 4:1. Together with the results on the bearing data set, the milling data results reinforce that PCC loss

Table 5
Bearing data set model performance over outer race fault condition in SNR dB.

Condition = Outer race fault					
Model	Compression factor				
	2	4	8	16	32
CS	2.31 ± 0.01	1.42 ± 0.01	0.41 ± 0.01	−0.26 ± 0.03	−0.67 ± 0.04
PCA	9.36	5.42	3.56	2.48	1.93
DCT	8.81	5.06	3.25	2.10	1.65
DCAE	9.57 ± 0.16	5.66 ± 0.12	2.84 ± 1.59	1.59 ± 0.79	0.85 ± 0.28
DCAE-C	10.28 ± 0.10	6.62 ± 0.08	**5.07 ± 0.06**	3.88 ± 0.08	2.81 ± 0.10
DCAE-C-AC	**10.31 ± 0.10**	6.61 ± 0.09	5.04 ± 0.04	3.85 ± 0.15	2.80 ± 0.16
DCAE-C-FFT1	10.25 ± 0.06	6.75 ± 0.05	4.99 ± 0.06	3.89 ± 0.12	2.64 ± 0.07
DCAE-C-FFT2	10.26 ± 0.15	6.63 ± 0.09	5.07 ± 0.12	**4.04 ± 0.09**	**2.86 ± 0.12**
FDAM-C	9.44 ± 0.08	6.90 ± 0.13	4.81 ± 0.07	3.52 ± 0.07	2.46 ± 0.04
FDAM-C-AC	9.49 ± 0.08	6.88 ± 0.06	4.94 ± 0.12	3.56 ± 0.05	2.46 ± 0.06
FDAM-C-FFT1	9.48 ± 0.13	6.90 ± 0.12	4.66 ± 0.69	3.55 ± 0.46	2.63 ± 0.32
FDAM-C-FFT2	10.10 ± 0.47	**6.93 ± 0.12**	4.85 ± 0.11	3.60 ± 0.12	2.57 ± 0.03

Table 6
Milling data set model performance in SNR dB.

Milling case study					
Model	Compression factor				
	2	4	8	16	32
CS	1.55 ± 1.20	1.45 ± 1.04	1.00 ± 0.57	0.26 ± 0.11	−0.29 ± 0.14
PCA	8.57 ± 1.66	**5.85 ± 1.65**	4.54 ± 1.30	3.51 ± 0.95	2.80 ± 0.68
DCT	**8.78 ± 1.61**	5.65 ± 1.59	4.24 ± 1.25	3.16 ± 0.76	2.20 ± 0.60
DCAE	5.79 ± 1.59	5.19 ± 1.61	4.66 ± 1.59	4.17 ± 1.52	3.84 ± 1.45
DCAE-C	6.27 ± 1.12	5.57 ± 1.12	4.73 ± 1.15	4.20 ± 1.17	3.92 ± 1.13
DCAE-C-AC	6.27 ± 1.11	5.63 ± 1.13	4.73 ± 1.15	4.22 ± 1.18	**4.01 ± 1.19**
DCAE-C-FFT1	6.46 ± 1.16	5.48 ± 1.14	4.66 ± 1.15	4.08 ± 1.13	3.74 ± 1.08
DCAE-C-FFT2	6.15 ± 1.15	5.64 ± 1.19	**4.78 ± 1.17**	**4.30 ± 1.25**	3.98 ± 1.18

Table 7
Performance in SNR dB for imbalanced bearing data set with 70% normal data. The relatively high standard deviations are not a reflection of true stochasticity, but of the model's varying ability to compress the different operating conditions.

Imbalanced bearing case study					
Model	Compression factor				
	2	4	8	16	32
CS	1.63 ± 0.46	1.19 ± 0.14	0.54 ± 0.30	−0.11 ± 0.26	−0.50 ± 0.20
PCA	**10.53 ± 4.14**	6.54 ± 3.32	4.44 ± 2.39	3.27 ± 2.28	2.50 ± 2.17
DCT	10.27 ± 4.14	6.14 ± 3.02	4.02 ± 2.21	2.86 ± 1.93	2.07 ± 1.68
DCAE	7.03 ± 2.74	5.42 ± 3.29	3.75 ± 3.42	3.01 ± 2.81	2.49 ± 2.31
DCAE-C	8.14 ± 3.25	6.28 ± 3.73	5.06 ± 3.32	4.07 ± 2.77	3.10 ± 2.25
DCAE-C-FFT2	8.01 ± 3.17	6.26 ± 3.65	5.03 ± 3.25	4.07 ± 2.72	2.95 ± 2.26
FDAM-C	6.89 ± 3.68	**6.58 ± 4.08**	5.55 ± 3.71	4.45 ± 3.09	3.45 ± 2.33
FDAM-C-FFT2	6.76 ± 3.53	6.54 ± 3.86	**5.56 ± 3.78**	**4.47 ± 3.09**	**3.47 ± 2.30**

consistently improves compression performance; additional PI loss terms have the potential to further improve results, with level of impact governed by the characteristics of the underlying signals.

4.3. Performance on imbalanced classes

In most condition monitoring applications, perfectly balanced normal and faulty data would not be available. Some situations permit the use of data augmentation techniques to artificially increase the number of faulty samples. For example, using a smaller window step when preprocessing fault signals than for normal signals will increase the relative amount of minority-class data, although such samples will be more correlated than the more widely spaced normal samples. However, when data augmentation is not possible, special attention must be given to designing and training DCAE-based models. In the unbalanced case, the network is inclined to learn how to accurately characterize the samples from the majority condition, leading to a deteriorated performance on the minority conditions. FDAM seeks to mitigate this by using separate DCAEs for the majority and minority class(es). While this may enable the DCAEs to focus on a single cohesive condition, care must be taken to avoid overfitting when working with

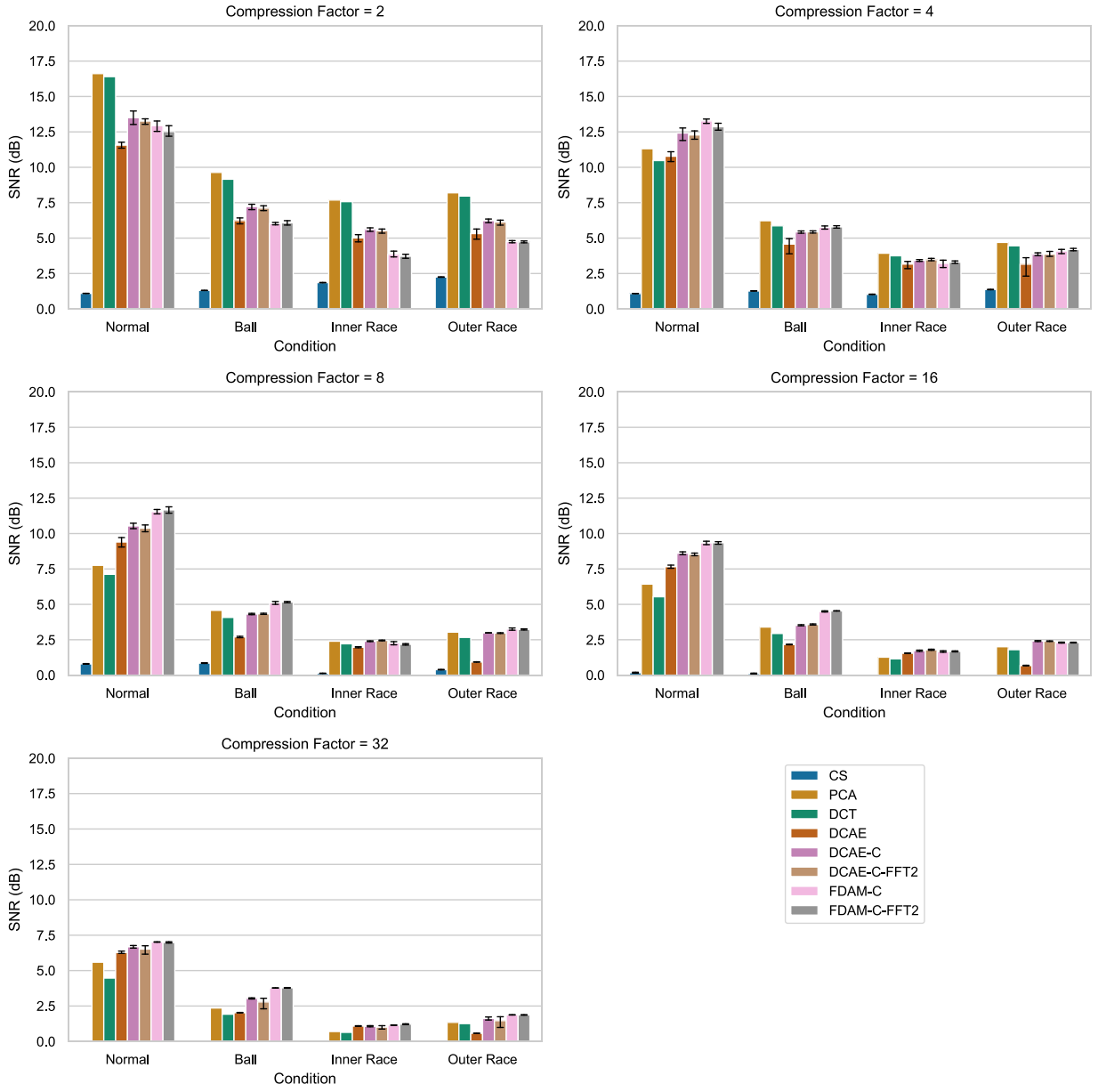

Fig. 15. SNR dB vs. compression factor for traditional and DL approaches on 70% normal/30% fault imbalanced bearing data set.

small-sample-size minority classes. Reducing the number of CNN layers or adopting dropout (i.e., randomly masking layer outputs to limit the capacity of the NN during training) can help regularize the network training. To explore method performance, CS, PCA, DCT, DCAE, DCAE-C, DCAE-C-FFT2, FDAM-C, and FDAM-C-FFT2 are tested on an imbalanced data set consisting of 70% bearing normal data and 30% bearing faulty data, by reducing the number of fault samples from 36 000 to approximately 5100. To mitigate overfitting, all DCAE models use four (instead of five) convolution layers and add a dropout rate of 0.2.

Fig. 15 and Table 7 survey the results of these experiments. Similar to the milling data results, PCA and DCT achieve the best performance at the 2:1 compression rate, but quickly drop below the DCAE-based methods at higher rates. As seen in Fig. 15, the DCAE-C noticeably outperforms DCAE across nearly all operating conditions and compression rates. The FDAM networks appear to produce higher SNR on normal and ball faulty conditions at all compression factors—inner race and outer race fault results are more comparable to the single-AE DCAE methods. These results indicate that the addition of PCC and other loss terms can improve performance on imbalanced data when compared to traditional compression and baseline DCAE methods, and future studies can clarify the role of PI loss terms in this scenario.

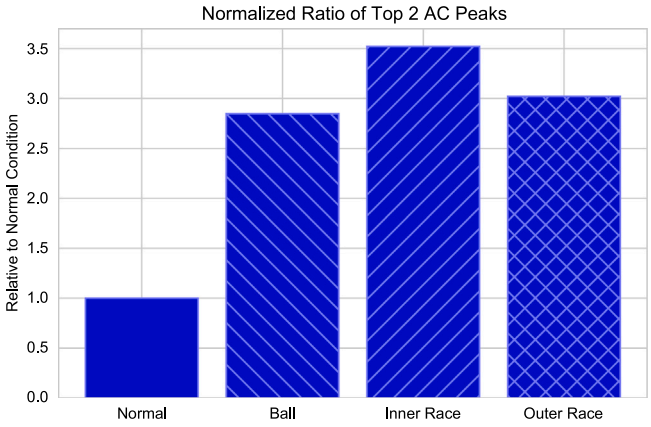

Fig. 16. Mean autocorrelation peak ratios as a proportion of the peak ratio for normal signals. Compare with Fig. 12 to see how the AC peak ratio inversely mirrors the SNR dB on the operating conditions.

4.4. Operating condition compressibility

For nearly all the DCAE models, the SNR results obtained from the bearing data set reveal a consistent pattern. Sorted in descending order gives the ranking of normal, ball fault, outer race fault, and inner race fault. A key consideration is whether this ordering of performance is caused by some aspect of DCAE models that make them less effective at outer race and inner race faulty data, or if it is a result of inherent unpredictability (i.e., noise) in the vibration data of these faults. Although signal noise information is not available for the bearing data set, relative noise magnitudes between the operating conditions can be estimated using autocorrelation (AC). The peak AC value of a signal with additive white gaussian noise (AWGN) occurs at a lag of zero. At this lag, the AC represents the combined power of the signal and noise. As the lag increases, the noise becomes uncorrelated, and its power no longer affects the magnitude of the AC. Thus, at nonzero lags, the AC represents a scaled version of only the signal power. Consequently, the ratio between the top two AC peaks produces a qualitative metric representing the noise content of the signal:

$$R = \frac{r_{xx}(l_1)}{r_{xx}(l_2)} \tag{12}$$

where l_1 and l_2 are the locations (i.e., lags) of the first and second largest values of the AC.

Fig. 16 shows the mean top-two peak ratio across the testing samples of each operating condition, normalized by the peak ratio of the normal condition. Significantly, the values of the peak ratio, which represent an estimate of the relative noise in the signal, inversely follow the pattern of SNR results shown in Fig. 12. This is expected since as the noise content increases, the algorithms are unable to compress the noise, causing the SNR of the reconstructed signal to decrease. Therefore, it is reasonable to assume that the relative differences in SNR among operating conditions could be attributable to the underlying signal noise, and are not indicative of any model's structural inability to represent the information in these signals.

5. Conclusion

This study demonstrates how incorporating local structure and physics-informed (PI) loss terms in addition to standard Mean Squared Error (MSE) loss can improve the PHM sensing signal compression performance by Deep Convolutional Autoencoder (DCAE). These domain-informed terms, e.g., FFT real and imaginary component loss, can improve network results by constraining the weight optimization space. In addition, Fault Division Autoencoder Multiplexing (FDAM) is proposed to further improve the performance of preserving characteristics of heterogeneous data across different machine conditions. Two case studies with representative high-velocity vibration signals typical of rotating machinery demonstrate that proposed methods noticeably outperform traditional non-DL data compression methods, and all DCAE models with PI loss terms outperform the baseline DCAE structure. Several key findings and recommendations can be outlined in light of the results:

- The inclusion of PCC loss, which was first quantitatively examined for PHM machine condition monitoring in this study, can improve reconstruction SNR by over 80%;
- PI loss terms can successfully provide secondary perspectives during model training, with the proposed real and imaginary FFT loss yielding additional SNR improvements over the baseline;
- The novel FDAM architecture can produce 10 to 15 percentage point SNR improvements over the corresponding DCAE architecture;
- Combinations of these techniques can allow data to be compressed up to two to four times smaller with the same level of reconstruction fidelity than if compressed using the baseline DCAE model.

Furthermore, the autocorrelation-based analysis provides insight into the differences in reconstruction performance across the operating conditions, indicating that the reconstruction performance is inversely proportional to the noise level of the sensing signal. The future research will further explore the effects of data imbalance on the DCAE and FDAM architectures, consider fusing the DCAE methods with classical compression techniques to improve their performance at high compression rates, and demonstrate the feasibility of this solution by implementing a complete IoT pipeline for cloud-based, physics-informed DL-enabled machine condition monitoring and PHM. Integrating improved compression algorithms into smart manufacturing infrastructure is an enabling technology for sending high-volume, high-velocity Big Data from connected factories to the cloud.

CRediT authorship contribution statement

Matthew Russell: Methodology, Software, Validation, Formal analysis, Data curation, Writing – original draft, Visualization. **Peng Wang:** Conceptualization, Methodology, Writing – review & editing.

Declaration of competing interest

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

Acknowledgments

This work was supported by National Science Foundation, United States grant number 2015889. We would thank the University of Kentucky Center for Computation Sciences and Information Technology Services Research Computing for their support and use of the Lipscomb Compute Cluster and associated research computing resources.

References

[1] S. Li, L.D. Xu, X. Wang, Compressed sensing signal and data acquisition in wireless sensor networks and Internet of Things, IEEE Trans. Ind. Inf. 9 (2013) 2177–2186.
[2] F. Civerchia, S. Bocchino, C. Salvadori, E. Rossi, L. Maggiani, M. Petracca, Industrial Internet of Things monitoring solution for advanced predictive maintenance applications, J. Ind. Inf. Integr. 7 (2017) 4–12.
[3] J. Wang, Y. Ma, L. Zhang, R.X. Gao, D. Wu, Deep learning for smart manufacturing: Methods and applications, J. Manuf. Syst. 48 (2018) 144–156.
[4] F. Ni, J. Zhang, M.N. Noori, Deep learning for data anomaly detection and data compression of a long-span suspension bridge, Comput.-Aided Civ. Infrastruct. Eng. (2019) 1–9.
[5] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.
[6] L. Theis, W. Shi, A. Cunningham, F. Huszár, Lossy image compression with compressive autoencoders, in: ICLR 2017, 2017.
[7] D. Del Testa, M. Rossi, Lightweight lossy compression of biometric patterns via denoising autoencoders, IEEE Signal Process. Lett. 22 (2015) 2304–2308.
[8] O. Yildirim, R.S. Tan, U.R. Acharya, An efficient compression of ECG signals using deep convolutional autoencoders, Cogn. Syst. Res. 52 (2018) 198–211.
[9] R.K. Mahendran, P. Velusamy, P. R, S. J, P. Pandian, An efficient priority-based convolutional auto-encoder approach to electrocardiogam signal compression in Internet of Things based healthcare system, Trans. Emerg. Telecommun. Technol. 32 (2020) 1–15.
[10] M. Sakurada, T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data, 2014, pp. 4–11.
[11] D.Y. Oh, I.D. Yun, Residual error based anomaly detection using auto-encoder in SMD machine sound, Sensors 18 (2018).
[12] G. Jiang, H. He, P. Xie, Y. Tang, Stacked multilevel-denoising autoencoders: A new representation learning approach for wind turbine gearbox fault diagnosis, IEEE Trans. Instrum. Meas. 66 (2017) 2391–2402.
[13] M. Li, Z. Wang, Deep learning for high-dimensional reliability analysis, Mech. Syst. Signal Process. 139 (2020).
[14] L. Ren, Y. Sun, J. Cui, L. Zhang, Bearing remaining useful life prediction based on deep autoencoder and deep neural networks, J. Manuf. Syst. 48 (2018) 71–77.
[15] H. Liu, J. Zhou, Y. Zheng, W. Jiang, Y. Zhang, Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders, ISA Trans. 77 (2018) 167–178.
[16] Z. Meng, Z. Zhan, J. Li, Z. Pan, An enhancement denoising autoencoder for rolling bearing fault diagnosis, Measurement 130 (2018) 448–454.
[17] X. Zhao, M. Jia, M. Lin, Deep Laplacian Auto-encoder and its application into imbalanced fault diagnosis of rotating machinery, Measurement 152 (2020).
[18] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio, Contractive auto-encoders: Explicit invariance during feature extraction, in: ICML 2011, 2011.
[19] A. Trilla, D. Miralles, V. Fernández, Pushing distributed vibration analysis to the edge with a low-resolution companding autoencoder: Industrial IoT for PHM, in: Annual Conference of the PHM Society, Vol. 12, 2020, p. 9.
[20] H. Shao, H. Jiang, Y. Lin, X. Li, A novel method for intelligent fault diagnosis of rolling bearings using ensemble deep autoencoders, Mech. Syst. Signal Process. 102 (2018) 278–297.
[21] H. Shao, H. Jiang, H. Zhao, F. Wang, A novel deep autoencoder feature learning method for rotating machinery fault diagnosis, Mech. Syst. Signal Process. 95 (2017) 187–204.
[22] R. Rai, C.K. Sahu, Driven by data or derived through physics? A review of hybrid physics guided machine learning techniques with cyber-physical system (CPS) focus, IEEE Access 8 (2020) 71050–71073.
[23] J. Ling, A. Kurzawski, J. Templeton, Reynolds averaged turbulence modeling using deep neural networks with embedded invariance, J. Fluid Mech. 807 (2016) 155–166.
[24] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[25] A. Dourado, F.A.C. Viana, Physics-informed neural networks for corrosion-fatigue prognosis, in: Annual Conference of the PHM Society, Vol. 11, 2019.
[26] Y.A. Yucesan, F.A.C. Viana, Wind turbine main bearing fatigue life estimation with physics-informed neural networks, in: Annual Conference of the PHM Society, Vol. 11, 2019.
[27] S. Bai, J. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018, arXiv preprint arXiv:1803.01271.

[28] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, 2015, arXiv:1511.06434. URL https://arxiv.org/pdf/1511.06434.pdf.
[29] B. Li, M.-Y. Chow, Y. Tipsuwan, J.C. Hung, Neural-network-based motor rolling bearing fault diagnosis, IEEE Trans. Ind. Electron. 47 (2000) 1060–1069.
[30] Bearing Data Center, Case Western Reserve University, https://csegroups.case.edu/bearingdatacenter/home (n.d.).
[31] A. Agogino, K. Goebel, Milling Data Set, BEST Lab, UC Berkeley, 2007, NASA Ames Prognostics Data Repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#milling).
[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035, http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.
[33] W. Falcon, et al., PyTorch Lightning, Vol. 3, GitHub, 2019, https://github.com/PyTorchLightning/pytorch-lightning.
[34] Neptune, Manage all your model building metadata in a single place, 2021, https://neptune.ai/.
[35] L.F. Polania, R.E. Carrillo, M. Blanco-Velasco, K.E. Barner, Compressed sensing based method for ECG compression, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, pp. 761–764.
[36] F. Castells, P. Laguna, L. Sörnmo, A. Bollman, J.M. Roig, Principal component analysis in ECG signal processing, EURASIP J. Adv. Signal Process. 2007 (2007) 1–21.
[37] K.R. Rao, P. Yip, Discrete Cosine Transform: Algorithms and Applications, Elsevier Science, 2014, p. 512.
[38] E.J. Candès, M.B. Wakin, An introduction to compressive sampling, IEEE Signal Process. Mag. March (2008) 21–30.