International Journal on Engineering Technologies and Informatics

On the Use of Convolutional Neural Network for Bearing Fault Diagnosis with Small Data

Research Article

Volume 4 Issue 3- 2023

Author Details

Issam Abu-Mahfouz and Amit Banerjee*

School of Science, Engineering and Technology, USA

*Corresponding author

Issam Abu-Mahfouz, School of Science, Engineering and Technology, Penn State Harrisburg, Middletown, PA 17057, USA

Article History

Received: April 28, 2023 Accepted: May 16, 2023 Published: May 18, 2023

Abstract

Rolling element bearings are commonly used in supporting rotor components and assemblies in rotating machinery. Bearing defects can lead to undesirable vibrations, noise, or machine failure. Modern predictive maintenance techniques are increasingly adopting artificial intelligence (AI) techniques for bearing fault diagnosis. In this work, wavelet analysis is used to process vibration signals from three bearing cases. These are: no fault, inner race fault, and ball fault under varying rotating speed. Small set of the wavelet scalogram images are fed into a Convolutional Neural Network (CNN) model for the classification of bearing defects into four classes based on operating speeds. Even with small sample size for training, we have been to achieve classification accuracies of around 90% with a specific combination of CNN parameters.

Keywords: Convolutional neural network, Deep learning; Fault diagnosis, Rolling element bearing, Vibration signals, Wavelet analysis

Abbreviation: AI: Artificial Intelligence; CNN: Convolutional Neural Network; DL: Deep Learning; ML: Machine Learning; EFMF: Energy-Fluctuated Multiscale Feature; PSR: Phase Space Reconstruction; BP: Backpropagation; DBN: Deep Belief Network; DAE : Deep Auto-Encoder; MC-CNN: Multi-Scale Cascade Convolutional Neural Network; OEDTN: Optimal Ensemble Deep Transfer Network; DTNs: Deep Transfer Networks; CWT: Continuous Wavelet Transform; DNN: Deep Neural Network; ANN: Artificial Neural Network; RPN: Region Proposal Network; SVM: Support Vector Machines

Introduction

Rolling bearings are crucial components of all types of rotating machinery equipment. Bearing fault diagnosis has been a subject of great importance in machine condition monitoring, predictive maintenance, and machine failure prevention and analysis. It has received considerable attention from researchers in both academia and the industry. In recent years, machine learning and deep learning-based intelligent fault diagnosis methods of rolling bearings have been widely and successfully developed. Many researchers have attempted to bridge the gap between established fault diagnosis expert rules and artificial intelligence (AI) methods. Deep Learning (DL) is a branch of Machine Learning (ML), with a deep multilayered architecture that has multiple levels of data abstraction and feature extraction. DL algorithms often require large computation resources. However, the low cost and high computational ability of modern computer processors accelerated the development and deployment of DL algorithms. Convolutional neural network (CNN) is one of the popular DL models that can perform feature learning and pattern classification automatically. In the following a review of previous work in the field of bearing fault diagnosis using DL is presented.

Z. Chen, et al. [1] employed three deep neural network models (Deep Boltzmann Machines, Deep Belief Networks and Stacked Auto-Encoders) to evaluate the performance of deep learning models for rolling bearing fault diagnosis. They discussed four preprocessing schemes including time domain, frequency domain and time-frequency domain features extracted from vibration signals. The accuracy achieved by Deep Boltzmann Machines, Deep Belief Networks and Stacked Auto-Encoders are highly reliable and applicable in fault diagnosis of rolling bearing. They also concluded that deeper architecture of deep neural network does not necessarily lead to better results. A novel energy-fluctuated multiscale feature (EFMF) mining method for spindle bearing fault diagnosis was presented in [2]. In this technique, the wavelet packet (WP) energy is first rebuilt into a 2-D image space by the phase space reconstruction (PSR), to reconstruct a local relationship of the WP nodes and reveal the energy fluctuation in a new WP phase space. Then these acquired WP images are used as the input to the deep CNN, where the backpropagation (BP) algorithm is employed to fine-tune the architecture by minimizing the log-likelihood function. Adaptive nonlinear signal decomposition and unsupervised feature learning method by using convolutional restricted Boltzmann machine model were proposed in [3]. This method automatically captures shift-invariant patterns hidden in the original signal and decompose the original signal into several fault-related sub-components, i.e., transient impulses signal, could be likely extracted. Subsequently maximizing kurtosis is applied to select optimally latent fault component. In [4], the authors proposed a novel domain adaptation method for rolling bearing fault diagnosis based on deep convolutional neural network architecture.

They showed that their domain adaptation methods were able to learn the domain-invariant features under two different motor loads and performed successful bearing faults diagnosis in the new domain that is different from the one with data available for training. Attention mechanism is introduced to assist the deep network to locate informative data segments and to extract the discriminative features as inputs to assist a deep network and to visualize the learned diagnosis knowledge [5]. The study aimed at bridging the gap between well-established fault signal processing approaches and deep learning methods. An ensemble deep learning diagnosis method based on multi-objective optimization was proposed in [6]. In this method, the Convolution Residual Network (CRN), Deep Belief Network (DBN) and Deep Auto-Encoder (DAE) were weighted and integrated to realize the effective diagnosis of rotor and bearing faults for rotating machinery. A literature review of the applications of three popular DL algorithms for bearing fault diagnosis was discussed in [7]. These are the Autoencoder, the Restricted Boltzmann Machine, and the Convolutional Neural Network. A hybrid deep signal processing method for bearing fault diagnosis is presented in [8] that incorporates vibration analysis techniques into deep learning. This is performed for automatic raw vibration signal processing, feature extraction, and bearing fault diagnosis. In [9], an improved CNN named multi-scale cascade convolutional neural network (MC-CNN) is proposed for the classification information enhancement of input. This was achieved by adding a new layer before convolutional layers to construct a new signal by concatenating the signals convolved by original input and kernels of different lengths. They also added a convolutional layer with kernels of small size and a pooling layer after the multi-scale cascade layer to reduce the neurons produced by the multi-scale signal. This technique was applied to pattern classification of bearing vibration signal under normal and noise environments. The kernels act as filters of different resolutions to distinguish the different faults signals in the frequency domain. Feature matching method based on multi-kernel maximum mean discrepancies between source domain and target domain is adopted to get enough labeled target domain signals in [10].

These rolling bearing signals are then converted to multi-dimensional graph samples before being fed into the CNN model. Generalization was improved, under variable operating conditions, by combining model-based transfer learning with feature-based transfer learning to initialize and optimize the CNN. A systematic and comprehensive review of the existing literature on bearing fault diagnostics with deep learning (DL) algorithms can be found in [11]. An optimal ensemble deep transfer network (OEDTN) for rolling bearing fault diagnosis with unlabeled data was proposed in [12]. The proposed method firstly takes advantage of parameter transfer learning, domain adaptation and ensemble learning. Different kernel MMDs are first used to construct multiple diverse deep transfer networks (DTNs) for feature adaptation. Then, parameter transfer was applied to optimally initialize these DTNs. The last step involved the use of ensemble learning to combine these DTNs to acquire the results. A comprehensive metric was designed to guide the particle swarm optimization to assign suitable voting weights for each DTN. Yanwei Xu, et al. [13] proposed an intelligent diagnosis method of rolling bearing fault based on an improved convolution neural network and light gradient boosting machine. In this method, the generalization ability of the model is improved by replacing the full connection layer with the global average pooling layer. The extracted features are classified by a light gradient boosting machine.

A 1D-CNN network architecture was proposed to effectively improve the accuracy of the diagnosis of rolling bearing, and the number of convolution kernels decreases with the reduction of the convolution kernel size [14]. The method directly used original bearing vibration data without preprocessing. A dropout layer was added to the 1D-CNN model to enhance its generalization. Many DL and CNN based applications have also been reported in [15,16]. A comprehensive review of the developments made in rotating bearing fault diagnosis during the past decade is presented in [17]. This review discussed various signal processing techniques, classical machine learning approaches, and deep learning algorithms used for bearing fault diagnosis. It also highlighted the available public datasets that have been widely used in bearing fault diagnosis experiments.

However, due to the required computation time and large data required for training, these methods may cause extended latency time when they are applied to machine fault diagnosis at the edge for real-time machine condition monitoring applications. The goal of this paper is to evaluate the feasibility of the implementing CNN with small data availability. The remaining sections of this paper are arranged, as follows. Section 2 presents a summary of the experimental procedure. Section 3 describes the CNN method, and its implementation is this work. Section 4 presents results and discussion. Finally, Section 5 presents the conclusion of this paper.

Experimental Procedure

A schematic of the experimental setup is shown in Figure 1. Three ball bearing conditions are considered. These are no fault, inner race fault, and ball fault. The rotor is run at 10 speeds (from 500rpm to 1400rpm) in increments of 100rpm. At each speed vibration signals are acquired using a a PCB accelerometer (model PCB 302A) which is moutned on the outboard test bearing as indicated in Figure 1. The sampling rate is 10,000 samples/sec. A radial load of about 2000lb was kept constant during all tests. A small unbalance mass is added to the rotor to introduce sustained periodic vibation excitation. Figure 2 shows sample raw vibration data and its corresponding frequency spectrum obtaine using the FFT algorithm. The raw time vibration signals are also analyzed using the continuous wavelet transform (CWT) techniques. Figures 3 presents sample scalograms.

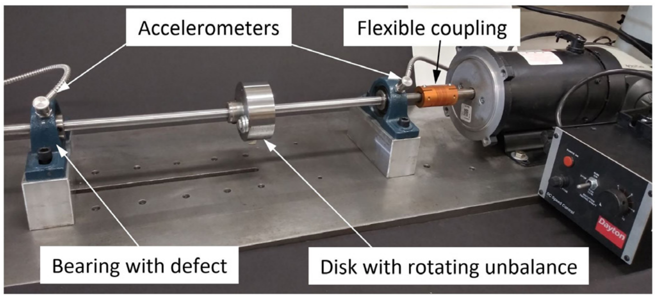

Figure 1: Experiment setup.

Wavelet transform is being increasingly used as a signal processing technique [18]. Continuous wavelet transform (CWT) analysis was performed using the MATLAB's wavelet toolbox. After experimenting with several wavelet transform functions, it was decided to use the 'Mexican hat' and the coiflet wavelet function. MATLAB's CWT can be considered as a filter that scales a mother wavelet function along the time axis. At each scale, the CWT function will superimpose the scaled mother wavelet wave form over a segment of the signal under analysis.

The similarities and differences between the form of the wavelet wave and the signal being analyzed are determined by the CWT

$$C(a,T)=\int \frac{1}{\sqrt{a}}\psi\left(\frac{t-T}{a}\right)x(t)dt \qquad (1)$$

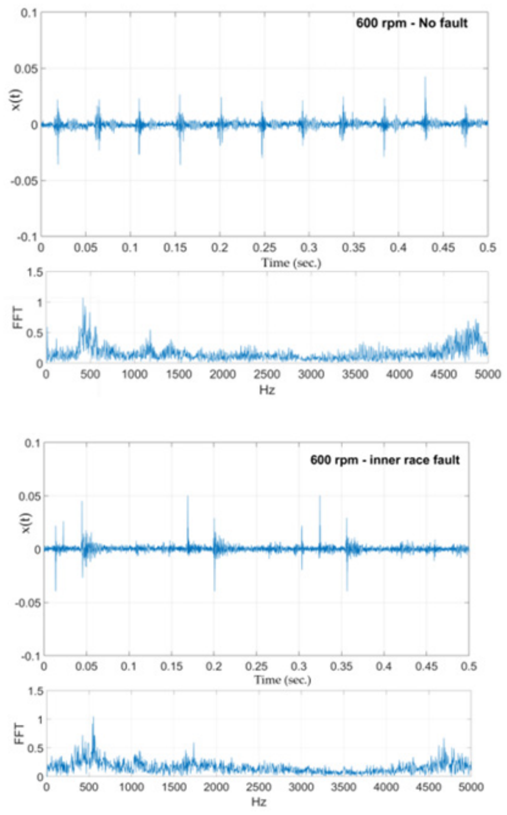

Figure 2: Vibration time signatures and corresponding FFT spectrum for two bearing cases.

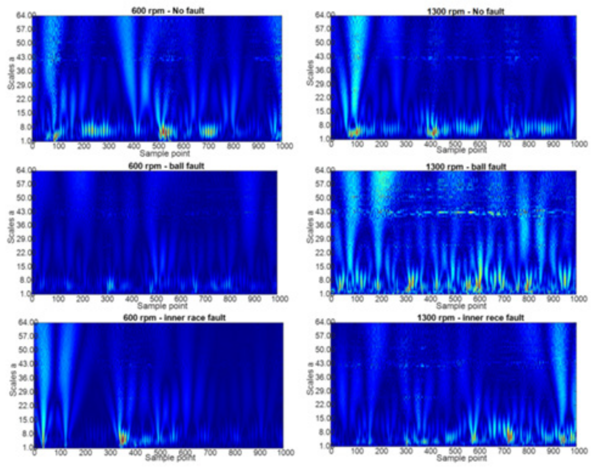

Figure 3: Sample wavelet scalogram images for the three bearing conditions at 600 rpm (left) and at 1300 rpm (right).

Where ψ(t) represents the CWT mother wavelet function which is shifted in time by T and dilated or contracted by a factor and then correlated with the vibration signal represented by x(t). The MATLAB toolbox gives the option to show the result of the CWT as a scalogram image. Several sample images for the vibration signals obtained from different bearing conditions are shown in Figure 3. The vertical axis represents the dilation or contraction of the mother wavelet function with large dilations at the top and smaller contractions at the bottom of the image. The horizontal axis in Figure 3 is labeled as the sample points in the acquired vibration signature. Each image represents only 1000 sampled points. Bright areas represent strong correlations between the vibration signal and the wavelet wave for a particular scaling while darker regions represent poor signal agreement. In the scalogram images, high scale values (top) of a scalogram represent low frequency content and low scale values (bottom) of a scalogram represent signal components with high frequency.

Method

Deep neural network (DNN) is a subtype of artificial neural network (ANN) with multiple hidden layers of units embedded between the input and output layers which has been widely used in the field of image processing and classification. Unlike ANNs which use features extracted from the images as features, DNNs use the images themselves and can self-extract features. Deep neural networks can be trained to learn such features at different levels of abstraction which may not be explicitly visible to the human eye even in a 2D image [18]. Deep Convolutional Neural Network (ConvNet or CNN) is a deep neural network which can take in an input image, assign importance (learnable weights and biases) to various aspects of the image and be able to design a classifier to distinguish images from one another. The preprocessing required in a CNN is much lower as compared to other classification algorithms. With enough training, CNNs have the ability to learn the characteristics of these filters in order to produce classification based on image labels. A convolutional neural network was shown to obtain good classification in detecting optical defects in [19]. It has been shown that CNNs can achieve an error rate of 7% for automated defect detection compared to manual visual inspection method in steel specimens in [20]. A CNN has been used to detect damage in gearboxes in [21] with classification accuracies exceeding 99%. Convolutional neural networks (CNN) are a subset of deep learning neural networks developed specifically for image processing and classification.

The advantage of CNN is that it can automatically detect features from an image map. The features are used to train the classifier. The architecture of a CNN is different from an artificial neural network (ANN) in that every CNN uses three types of layers – convolution layer, pooling layer and a fully connected layer. ANN on the other hand has an input layer, one or many hidden layers and an output layer. In a CNN there are usually multiple stages of convolutional and pooling layers. A sample image is fed to the convolutional layer which uses linear and nonlinear functions to create a feature map from the input image by a kernel operation. The kernel is a two-dimensional matrix that when applied to the input image produces regions of high or low activations in the input – these are the self-extracted features. Usually there are multiple kernels, and the convolution layer produces as many feature maps as the number of kernels used. The activation function, usually nonlinear functions such as the hyperbolic tan (tanh) or the sigmoid function, or piecewise linear activation functions such as the rectified linear unit (ReLu) are used to create a fully activated feature map.

The activation output y^l_j of the jth particular feature map in the convolutional layer l, after adding the bias and passing through an activation function, is given by equation 2 in [22] and reproduced here.

$$y_j^l = \Phi\left(b_j^l + \sum_{i\in M_j^l} y_i^{l-1} * k_{ij}^l\right) \qquad (2)$$

where Φ is the activation function, b^l_j is the scalar bias for *l* layer, M^l_j is the selected feature map i in the (*l* – 1) layer which is summed over by the feature map *j* in the *l* layer, *denotes the convolutional operator that convolutes the activation y^{l-1}_j of the preceding layer and k^l_{ij} is the kernel used to perform that convolutional operation.

After each convolution layer, the extracted feature sets pass through the pooling layer. The pooling layer is essentially a downsampling operation that reduces the in-plane dimensionality of the feature map. The activation output d^l_h after downsampling the feature map j into a feature map h in a layer l is given by equation 3 in [22] and repre- oduced here.

$$d_h^l = \Psi\left(a_j^l, N^l\right) \tag{3}$$

where Ψ is the downsampling function such as maximum or mean function that down samples by a factor of N^l and a^l_j is the convoluted (and activated) feature map to be downsampled. The downsampling factor should be selected to ensure there is no loss of information in the feature maps due to dimensionality reduction. As the original input passes through successive stages of convolution and pooling, the network learns to efficiently represent the input image with relatively high-quality features of manageable dimensions. The output of the final pooling layer is flattened to create a one-dimensional array which is then connected to one or more fully connected layers. These are dense layers in which inputs and outputs are connected through a learnable weight (similar to an ANN). These are mapped to class probabilities in the classification task. The final fully connected layer has the same number of output nodes as the classes, and each fully connected layer is followed by an activation function such as ReLu or tanh. The activation at the last fully connected layer (output nodes) is different from the what is applied to the output of the convolution layer – it is usually a multi-class classification task activation such as softmax function which normalizes the real-valued output from the last layer to class probabilities.

Region-based CNN or RCNN is a variant of CNN that uses bounding boxes over different regions of the image and applies convolution independently to those regions [23]. They are especially useful when the number of occurrences of objects of interest across images used for training is not fixed with varying spatial locations and aspect ratios. RCNNs use a selective search method to extract region proposals (a set of 2000 regions). RCNNs' image segmentation capabilities have been shown to be better than traditional CNN, although it does tend to be slower and requires more storage than traditional CNN does. Fast-RCNN [24] and Faster-RCNN [25] significantly improve the efficiency of RCNN and reduce storage requirements by not having to feed multiple region proposals to the convolutional network every time. Instead, the convolution operation is done only once per image and a feature map is generated from it. Faster-RCNN uses a trainable region proposal network (RPN) instead of category-independent region proposals such as selective search or constrained parametric min-cuts that RCNNs use. The RPN is simply a neural network that proposes multiple objects that are available within a particular image.

Vibration signals are processed using wavelet transform and converting them to 2D images of size 514 x 649 pixels with three color channels. The image is first resized to 82 x 110 x 3 and then the three channels are combined into yield a single 2D image. These are used as inputs to the CNN and RCNN used in this work. The deep neural network toolbox for MATLAB described in [26] is used in this work. The network architecture for the CNN is described below and shown in Figure 4.

- The first layer following the input layer is a convolutional layer with 12 or 6 feature maps of kernel size 11 × 11 and 15 x 15. For a kernel size of 11 x 11, the result is a convoluted image of size 72 x 100, and for the 15 x 15 kernel, the result is a convoluted image of size 68 x 96. This is followed by a mean-pooling layer of size 2 × 2 resulting in pooled activations at 36 x 50 or 34 x 48.

- The next layer is a convolutional layer with 12, 6 or 3 feature maps of 5 × 5 resulting in convoluted images of size 32 x 46 for pooled activated images of size 36 x 50 or convoluted image of size 30 x 44 for pooled activated images of size 34 x 48 after the first layer. This is followed by a 2 × 2 mean-pooling layer resulting in pooled activations at 16 x 23 or 17 x 24.

- This is followed by a third convolutional layer with 6 feature maps of 7 x 8 or 8 x 9 resulting in convoluted images of size 10 x 16 for all cases. This is followed by a size 2 x 2 max-pooling layer resulting in pooled activations of size 5 x 8. This vector is flattened so that input to the classification part of the network is of size 40 x 1.

- The output layer contains 4 neurons corresponding to the 4 different fault states. These are low-speed + no-defect (0), high-speed + no-defect (1), low-speed + defect (2), and high-speed + defect (3). The class label assignments are shown in Table 1. All the layers are fully connected. ReLu and tanh functions is used as activation functions. Stochastic gradient descent method is used to train the network with a learning rate of 1. The batch size is 5 and the training was carried out for 50 epochs.

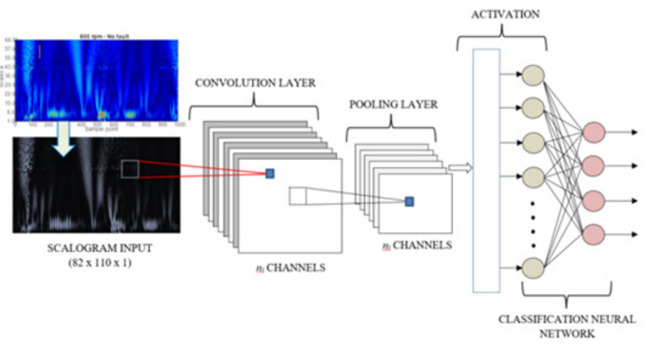

Figure 4: A schematic of the CNN algorithm for bearing fault diagnosis.

The architecture of RCNN is shown in Fig. 5. The detector first generates region proposals using Edge Boxes [27], the proposed regions Figure 5 are cropped out of the image and resized. The CNN described above is used to classify the cropped and resized regions. The region proposal bounding boxes are refined by a Support Vector Machines (SVM) that is trained using CNN features. Both trainRCNNObjectDetector() and trainFastRCNNObjectDetector() functions in the computer vision toolbox of MATLAB are used in this study. While the former is show in training and detection, both allow customized region proposals.

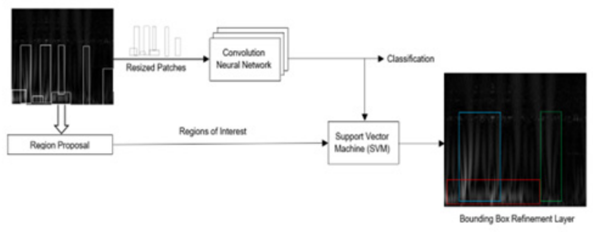
Resized Patches
Convolution Neural Network
Classification
Region Proposal
Regions of Interest
Support Vector Machine (SVM)
Bounding Box Refinement Layer

Figure 5: Architecture of the RCNN algorithm.

The following datasets are used:

- Dataset 1: no overlap between successive signals (both Mexican Hat and Coiflet 4 wavelets) for all three conditions with 180 scalogram images

- Dataset 2: 50% overlap between successive signals (both Mexican Hat and Coiflet wavelets) for all three conditions with 120 scalogram images

- Dataset 3: 25% overlap between successive signals (both Mexican Hat and Coiflet wavelets) for all three conditions with 120 scalogram images.

- Dataset 4: All Mexican Hat wavelet scalograms (for all three conditions – seeded ball fault, seeded inner race fault and no fault) with non-overlapping and overlapping signals with 210 images.

- Dataset 5: All Coiflet wavelet scalograms (for all three conditions – seeded ball fault, seeded inner race fault and no fault) with non-overlapping and overlapping signals with 210 images.

- Dataset 6: The entire dataset of 420 images is also used as a benchmark for comparison.

70% of the images in each set are used for training and 30% for validation (70-30 split). A three-fold cross-validation technique is used to create the training and validation subsets. The folds were first scanned to ensure that all samples from all four classes are evenly distributed in the training set. The classification results are analyzed using the following metrics – Accuracy, Precision, Recall and F-score. Accuracy is defined as the ratio of the total number of correctly assigned samples to the total number of samples in the validation subset. For example, in dataset 1 where 126 images are used for training and 54 for validation, if 5 out of the 54 images are incorrectly identified, the accuracy of the classifier is 0.91. Precision is used to measure correctness of a classifier and is sometimes called the positive predictive value while Recall is used to measure the completeness (or the sensitivity in binary classification). These two measures are calculated on each of the four classes one at a time. Precision on a certain class is the ratio of the number of samples in the testing set that are correctly classified as belonging to that class to all of the samples that are classified as belonging to that class. Recall is the ratio of correctly classified samples of that class to the total number of samples for that class. Class-wise average values of precision and recall are used to calculate the F-score as shown below.

$$\left.\begin{aligned} &\text{Precision} = \frac{TP}{TP+FP} \\ &\text{Recall} = \frac{TP}{TP+FN} \\ &\text{F-Score } F_1 = \frac{2}{\text{Recall}^{-1}+\text{Precision}^{-1}} = \frac{TP}{TP+0.5(FP+FN)} \end{aligned}\right\} \quad (4)$$

TP (True Positives) Samples correctly classified as belonging to a certain class

FP (False Positives) Samples incorrectly classified as belonging to a certain class

FN (False Negatives) Samples that actually belong to a certain class but incorrectly classified as belonging to another class

To calculate accuracy, we use the total misclassifications (total FN or FP); precision, recall and F-score are based on TP, FP and FN for the median class with respect to misclassifications as calculated from a confusion matrix.

Results and Discussion

In this work, classification problem as a 6-class problem where the differentiators are operating speed (high speeds above 1000rpm and low below 1000rpm) and type of fault (seeded ball defect, seeded inner race fault and no fault). The class labels are shown in Table 1. Different combinations of kernel size, number of feature maps (# of kernels) in the convolutional layer and the type of pooling strategy for the CNN are listed in Table 2. Accuracy, precision, recall and F-score for the different parameter combinations for the six datasets are reported in Tables 3-8. It was observed that three combinations of feature map numbers and kernel size over the three layers produced the best results (Combinations # 2, 10 and 12) It is also very clear that decreasing the number of feature maps affected the classification accuracy, which makes intuitive sense. Since all combinations resulted in the same number in the flattened classification layer (40 neurons), we were able to overcome the problem reported in [22] where increasing the number of neurons in the classification layer, the network seems to lose its sensitivity to distinguish between classes. In the same work, it was also observed that when the number of feature maps in the first convolution layer was increased, while decreasing the number of feature maps in the second layer, the classification performance increased. We observe a similar trend but not to the extent as reported in [22]. The balance between number of feature maps and kernel size is important so as to avoid under-training and over-training the network. It was also observed that the choice of activation methodology (ReLu or tanh) did not produce any significant difference in classification performance.

The computational load on the network is less with ReLu activation since not all neurons are fired at the same time – which has implications in training time of the network. The Coiflet wavelet scalograms (dataset #5 in Table 7) seems to have a better discriminating ability than Mexican Hat wavelet scalogram images (dataset #4 in Table 6). A Coiflet filter acts as band-pass filter as well as a high-pass or low-pass filter. Unlike the Mexican hat-based filter which acts only as a pass-band filter, the Coiflet-based filter allows for selecting horizontal, vertical or diagonals details of the original image. The Coiflet wavelets have more spatial information than Mexican Hat wavelet but have been shown to do poorly in the presence of a high amount of noise [28]. Results in our work compare favorably to results on larger dataset (CWRU Bearing dataset, 48,000 samples) using Scalograms and Switchable Normalization-based CNN [29] and slightly better results using the classical CNN as reported on the same CWRU Bearing dataset [30].

For the RCNN, we have used the best parameter combination resulting from the independent runs of the CNN (CNN parameter combination #12). Regions are extracted using Edge Boxes [27] and these regions are stitched together to create a warped region which is then fed to the CNN instead of the original scalogram. Both the regular RCNN object detector and the Fast RCNN object detector are used separately. The validation results for the six datasets are shown in Table 9. In most cases, we see a relatively high classification accuracy by 20-50 epochs of the max 50 epochs used in the study. The benchmark dataset (dataset # 6) which has 294 training images and 126 validation images performed only marginally better than dataset # 1 which had no overlap, implying that using the overlapping bounding scalograms had little incremental effect on classification accuracy of the algorithm. The region-based information in Coiflet wavelets is also richer and more discriminating compared the Mexican Hat wavelets. The run-time for the Object Detector algorithm (run separately from the CNN) is also given in Table 9 for simulations using MATLAB deep neural learning toolbox on Intel i7-8650 CPU @ 1.9 GHz with 4 cores and 8 logical processers. The implementation on a GPU or dedicated multi-core processors will be significantly faster.

Table 1: Class labels for different combinations of operating speeds and defect conditions.

#	Operating Speed (rpm)	Defect	Class label
1	500	No defect	0
2	600	No defect	0
3	700	No defect	0
4	800	No defect	0
5	900	No defect	0
6	1000	No defect	1
7	1100	No defect	1

8	1200	No defect	1
9	1300	No defect	1
10	1400	No defect	1
11	500	Inner race defect	2
12	600	Inner race defect	2
13	700	Inner race defect	2
14	800	Inner race defect	2
15	900	Inner race defect	2
16	1000	Inner race defect	3
17	1100	Inner race defect	3
18	1200	Inner race defect	3
19	1300	Inner race defect	3
20	1400	Inner race defect	3
21	500	Ball defect	4
22	600	Ball defect	4
23	700	Ball defect	4
24	800	Ball defect	4
25	900	Ball defect	4
26	1000	Ball defect	5
27	1100	Ball defect	5
28	1200	Ball defect	5
29	1300	Ball defect	5
30	1400	Ball defect	5

Table 2: Different feature learning parameter combinations (three sets of convolution and pooling layers and two different activation functions) tested.

CNN Combina-tion	**Convolution Layer 1**		**Convolution Layer 2**		**Convolution Layer 3**		**Activa-tion**
	# of ker-nels	Kernel size	# of ker-nels	Kernel size	# of ker-nels	Kernel size	
1	12	11 x 11	12	5 x 5	6	7 x 8	ReLu
2	12	11 x 11	6	5 x 5	6	7 x 8	ReLu
3	12	11 x 11	3	5 x 5	6	7 x 8	ReLu
4	6	11 x 11	12	5 x 5	6	7 x 8	tanh
5	6	11 x 11	6	5 x 5	6	7 x 8	tanh
6	6	11 x 11	3	5 x 5	6	7 x 8	tanh
7	12	15 x 15	12	5 x 5	6	8 x 9	ReLu
8	12	15 x 15	6	5 x 5	6	8 x 9	ReLu
9	12	15 x 15	3	5 x 5	6	8 x 9	ReLu
10	6	15 x 15	12	5 x 5	6	8 x 9	tanh
11	6	15 x 15	6	5 x 5	6	8 x 9	tanh
12	6	15 x 15	3	5 x 5	6	8 x 9	tanh

Table 3: Classification performance of different feature learning parameter combinations for dataset 1 (180 samples – 126 training samples and 54 validation samples).

CNN Combination	Accuracy	Preci-sion	Recall	F-score
1	0.778	0.840	0.913	0.875
2	0.926	0.943	0.980	0.962
3	0.815	0.880	0.917	0.898
4	0.815	0.898	0.898	0.898
5	0.852	0.885	0.958	0.920
6	0.778	0.823	0.933	0.875
7	0.741	0.833	0.869	0.851
8	0.833	0.882	0.937	0.909
9	0.852	0.868	0.979	0.920
10	0.907	0.942	0.961	0.951
11	0.852	0.902	0.939	0.920
12	0.926	0.962	0.961	0.962

Table 4: Classification performance of different feature learning parameter combinations for dataset 2 (120 samples – 84 training samples and 36 validation samples).

CNN Combina-tion	**Accuracy**	**Precision**	**Recall**	**F-score**
1	0.750	0.794	0.931	0.857
2	0.861	0.886	0.969	0.925
3	0.778	0.848	0.903	0.875
4	0.722	0.788	0.896	0.839
5	0.778	0.848	0.903	0.875
6	0.667	0.727	0.889	0.800
7	0.722	0.788	0.897	0.839
8	0.722	0.788	0.897	0.839
9	0.667	0.750	0.857	0.800
10	0.806	0.853	0.935	0.892
11	0.750	0.818	0.900	0.857
12	0.889	0.914	0.970	0.941

Table 5: Classification performance of different feature learning parameter combinations for dataset 3 (120 samples – 84 training samples and 36 validation samples).

CNN Combina-tion	**Accuracy**	**Precision**	**Recall**	**F-score**
1	0.694	0.781	0.862	0.820
2	0.777	0.823	0.933	0.875
3	0.667	0.774	0.827	0.800
4	0.583	0.700	0.778	0.737
5	0.694	0.806	0.833	0.820
6	0.639	0.697	0.885	0.780
7	0.667	0.750	0.857	0.800
8	0.583	0.656	0.84	0.737
9	0.555	0.645	0.800	0.714
10	0.750	0.818	0.900	0.857
11	0.694	0.781	0.862	0.820
12	0.806	0.853	0.935	0.892

Table 6: Classification performance of different feature learning parameter combinations for dataset 4 (210 Mexican Hat wavelet scalogram samples – 147 training samples and 63 validation samples).

CNN Combination	Accuracy	Precision	Recall	F-score
1	0.714	0.789	0.882	0.833
2	0.809	0.864	0.927	0.895
3	0.762	0.857	0.873	0.865
4	0.714	0.804	0.865	0.833
5	0.730	0.793	0.902	0.844
6	0.667	0.750	0.857	0.800
7	0.698	0.772	0.880	0.822
8	0.730	0.807	0.885	0.844
9	0.714	0.804	0.865	0.833
10	0.809	0.864	0.927	0.895
11	0.762	0.842	0.889	0.865
12	0.841	0.883	0.946	0.914

Table 7: Classification performance of different feature learning parameter combinations for dataset 5 (210 coiflet wavelet scalogram samples – 147 training samples and 63 validation samples).

CNN Combination	Accuracy	Precision	Recall	F-score
1	0.714	0.750	0.937	0.833
2	0.857	0.885	0.964	0.923
3	0.810	0.850	0.944	0.895
4	0.762	0.828	0.906	0.865
5	0.810	0.850	0.944	0.895
6	0.746	0.797	0.922	0.855
7	0.730	0.767	0.939	0.844
8	0.778	0.803	0.961	0.875
9	0.762	0.800	0.941	0.865
10	0.825	0.867	0.945	0.904
11	0.841	0.869	0.964	0.914
12	0.873	0.902	0.965	0.932

Table 8: Classification performance of different feature learning parameter combinations for dataset 6 (420 samples – 294 training samples and 126 validation samples).

CNN Combination	Accuracy	Precision	Recall	F-score
1	0.770	0.822	0.924	0.870
2	0.921	0.943	0.975	0.959
3	0.825	0.867	0.945	0.904
4	0.794	0.833	0.943	0.885
5	0.857	0.900	0.947	0.923
6	0.786	0.853	0.908	0.880
7	0.754	0.812	0.913	0.860
8	0.825	0.874	0.937	0.904
9	0.849	0.899	0.939	0.918
10	0.897	0.934	0.958	0.946
11	0.833	0.861	0.936	0.909
12	0.913	0.935	0.975	0.954

Table 9: Classification performance using RCNN for dataset 1 (180 samples – 126 training samples and 54 validation samples) for CNN parameter combination # 12.

Dataset	Object Detector	Accuracy	Preci-sion	Recall	F-score	Run Time (s)
1	RCNN	0.944	0.962	0.981	0.971	182
1	Fast-RCNN	0.926	0.943	0.981	0.962	153
2	RCNN	0.889	0.914	0.970	0.941	122
2	Fast-RCNN	0.889	0.914	0.970	0.941	93
3	RCNN	0.861	0.911	0.968	0.925	136
3	Fast-RCNN	0.833	0.857	0.940	0.909	102
4	RCNN	0.833	0.882	1.000	0.909	282
4	Fast-RCNN	0.833	0.833	0.937	0.909	253
5	RCNN	0.889	0.918	0.966	0.941	298
5	Fast-RCNN	0.873	0.902	0.965	0.932	264
6	RCNN	0.929	0.943	0.983	0.963	620
6	Fast-RCNN	0.920	0.943	0.975	0.959	587

Conclusions

Continuous monitoring of the condition of rolling element bearings to avoid machine failure is of great significance. In recent years Deep Learning (DL) has been very attractive approach in the field of Machine Condition Monitoring (MCM). Most machine health monitoring applications use neural networks with features extracted from sensor data. Deep learning neural networks developed specifically for 2D image classification are well-suited for pattern recognition and classification using wavelet images instead of features extracted from raw data. In this work vibration signals are collected and analyzed for bearings with no fault, seeded inner race fault, and seeded ball fault. Multiple datasets were generated by applying two wavelet filters – Mexican Hat and coiflet wavelets with overlapping and non-overlapping sections of the vibration signal time-series. Convolutional Neural Networks (CNN) and Region-based CNNs (RCNNs) with two Object Detector algorithms were trained using wavelet scalogram images. The main objective is to investigate the classification of operating parameters (rotational speed and type of fault) into 6 classes. We have achieved a classification accuracy in excess of 90% in some cases (with F-score in excess of 90% in most cases).

Results of this work showed that wavelet images and CNN are promising tools for intelligent fault detection. In future research, we will continue to explore the use of DL with small data, and we will strive to improve the prediction rate for fault detection in rotating machinery. In addition to using Mexican Hat wavelets and coiflet wavelets, the network can be trained with other wavelet images such as Haar wavelets. With multiple types of inputs, an ensemble CNN will be developed for combining classification predictions from different models.

References

1. Zhiqiang Chen, Shengcai Deng, Xudong Chen, Chuan Li, René-Vinicio Sanchez, et al. (2017) Deep neural networks-based rolling bearing fault diagnosis. Microelectronics Reliability 75: 327-333.
2. Xiaoxi Ding, Qingbo He (2017) Energy-Fluctuated Multiscale Feature Learning with Deep ConvNet for Intelligent Spindle Bearing Fault Diagnosis. IEEE transactions on instrumentation and measurements 66(8).
3. Longting Chen, Guanghua Xu, Yi Wang, Jianhua Wang (2018) Detection of weak transient signals based on unsu-pervised learning for bearing fault diagnosis. Neurocomputing 314: 445-457.
4. Xiang Li, Wei Zhang, Qian Ding, Jian-Qiao Sun (2019) Multi-Layer domain adaptation method for rolling bearing fault diagnosis. Signal Processing 157: 180-197.
5. Xiang Li, Wei Zhang, Qian Ding (2019) Understanding and improving deep learning-based rolling bearing fault diagnosis with attention mechanism. Signal Processing 161: 136-154.
6. Sai Ma, Fulei Chua (2019) Ensemble deep learning-based fault diagnosis of rotor bearing systems. Computers in Industry 105: 143-152.
7. Duy-Tang Hoang, Hee-Jun Kang (2019) A survey on Deep Learning based bearing fault diagnosis. Neurocomputing 335: 327-335.
8. Miao He, David He (2020) A new hybrid deep signal processing approach for bearing fault diagnosis using vibration signals. Neurocomputing 396: 542-555.
9. Wenyi Huang, Junsheng Cheng, Yu Yang, Gaoyuan Guo (2019) An improved deep convolutional neural network with mul-ti-scale information for bearing fault diagnosis. Neurocomputing 359: 77-92.
10. Changchang Che, Huawei Wang, Qiang Fu and Xiaomei Ni (2019) Deep transfer learning for rolling bearing fault diagnosis under variable operating conditions. Advances in Mechanical Engineering 11(12): 1-11.
11. Shen Zhang, Shibo Zhang, Bingnan Wang, Thomas G Habetler (2020) Deep Learning Algorithms for Bearing Fault Diagnostics A Comprehensive Review. IEEEAccess, Multidisciplinary Rapid Review 8.
12. Xingqiu Li, Hongkai Jiang, Ruixin Wang, Maogui Niu (2021) Rolling bearing fault diagnosis using optimal ensemble deep transfer Network. Knowledge-Based Systems 213: 106695.
13. Yanwei Xu, Weiwei Cai, Liuyang Wang, Tancheng Xie (2021) Intelligent Diagnosis of Rolling Bearing Fault Based on Improved Convolutional Neural Network and LightGBM. Shock and Vibration 8.
14. Chih-Cheng Chen, Zhen Liu, Guangsong Yang, Chia-Chun Wu, Qiubo Ye (2021) An Improved Fault Diagnosis Using 1D-Convolutional Neural Network Model. Electronics 10(1): 59.
15. Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, et al. (2019) Deep Learning and its Applications to Machine Health Monitoring. Mechanical Systems and Signal Processing 115: 213-237.
16. Samir Khan, Takehisa Yairi (2018) A Review on the Application of Deep Learning in System Health Management. Mechanical Systems and Signal Processing 107: 241-265.
17. Shiza Mushtaq, MM Manjurul Islam, Muhammad Sohaib (2021) Deep Learning Aided Data-Driven Fault Diagnosis of Rotatory Machine: A Comprehensive Review. Energies 14(16): 5150.
18. Lecun Y, Bengio Y, Hinton G (2015) Deep Learning. Nature 521: 436-444.
19. Weimer D, Scholz-Reiter B, Shpitalni M (2016) Design of deep convolutional neural network architectures for automated feature extraction in

industrial inspection. CIRP Annals-Manufacturing Technology 65(1): 417-420.

20. Masci J, Meier U, Ciresan D, Schmidhuber J, Fricout G (2012) Steel defect classification with max-pooling convolutional neural networks. In 2012 International Joint Conference on Neural Networks (IJCNN), p: 1-6.

21. M Zhao, M Kang, B Tang, Michael Pecht (2018) Deep residual networks with dynamically weighted wavelet coefficients for fault Diagnosis of planetary gearboxes. IEEE Transactions on Industrial Electronics 65 (5): 4290-4300.

22. Peng Wang, Ananya, Ruqiang Yan, Robert X Gao (2017) Virtualization and Deep Recognition for System Fault Classification. Journal of Manufacturing Systems 44: 310-316.

23. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik (2014) Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. Proc IEEE Conf Comput Vis Pattern Recognit (CVPR), pp: 580-587.

24. Ross Girshick (2015) Fast R-CNN, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp: 1440-1448.

25. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans Pattern Anal. Mach Intell 39(6): 1137-1149.

26. Phil Kim (2017) MATLAB Deep Learning: With Machine Learning Neural Networks and Artificial Intelligence. Apress Media.

27. Lawrence Zitnick, Piotr Dollár (2014) Edge boxes: Locating Object Proposals from Edge, European Conference on Computer Vision. Springer Cham.

28. Diego Suárez, Ángel Salazar (2009) Comparative Study of Pattern Correlation Using Mexican Hat and Coiflet Wavelet-based Filtering. Optics communications 282(21): 4203-4209.

29. Dhiraj Neupane, Yunsu Kim, Jongwon Seok (2021) Bearing Fault Detection Using Scalogram and Switchable Normalization-Based CNN (SN-CNN). IEEE Access 9: 88151-88166.

30. Mingxuan Liang, Pei Cao, J Tang (2021) Rolling Bearing Fault Diagnosis based on Feature Fusion with Parallel Convolutional Neural Network. Int J Adv Manuf Techn 112: 819-831.