Research Article

Bearing Fault Diagnosis Based on Multiscale Convolutional Neural Network Using Data Augmentation

Seungmin Han, Seokju Oh, and Jongpil Jeong

Department of Smart Factory Convergence, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea

Correspondence should be addressed to Jongpil Jeong; jpjeong@skku.edu

Received 25 December 2020; Revised 21 January 2021; Accepted 2 February 2021; Published 22 February 2021

Academic Editor: Jose-Luis Casteleiro-Roca

Copyright © 2021 Seungmin Han et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Bearings are one of the most important parts of a rotating machine. Bearing failure can lead to mechanical failure, financial loss, and even personal injury. In recent years, various deep learning techniques have been used to diagnose bearing faults in rotating machines. However, deep learning technology has a data imbalance problem because it requires huge amounts of data. To solve this problem, we used data augmentation techniques. In addition, Convolutional Neural Network, one of the deep learning models, is a method capable of performing feature learning without prior knowledge. However, since conventional fault diagnosis based on CNN can only extract single-scale features, not only useful information may be lost but also domain shift problems may occur. In this paper, we proposed a Multiscale Convolutional Neural Network (MSCNN) to extract more powerful and differentiated features from raw signals. MSCNN can learn more powerful feature expression than conventional CNN through multiscale convolution operation and reduce the number of parameters and training time. The proposed model proved better results and validated the effectiveness of the model compared to 2D-CNN and 1D-CNN.

1. Introduction

The development of the IoT and industrial applications is rapidly improving the intelligence of equipment in the modern industry. As a result, mechanical equipment is becoming increasingly sophisticated and complex. Machinery failure can cause significant financial loss as well as human casualties. The rotating machine is one of the most widely used machines in the industry [1]. Rolling bearings are very essential components in rotating machines [2, 3]. Therefore, diagnosing bearing failure is very important.

Recently, data-based fault diagnosis [4–6] is drawing a lot of attention to the researchers due to the development of computers and GPUs. Traditional model-based diagnostic methods [7–11] are not efficient for learning nonlinear data. In addition, in the feature extraction step [12–15], there is a large difference in the result value depending on the skill of the expert. Machine learning methods such as Support Vector Machine (SVM) [16, 17], Principal Component Analysis (PCA) [18, 19], and artificial neural network (ANN) [20] have been used frequently. However, traditional machine learning [21–23] has also difficulty handling complex data.

On the other hand, data-driven diagnostics can effectively and accurately express the characteristics of big data or complex input data. With the advent of deep learning, it is possible to train neural networks through very deep continuous layers. Deep learning [24, 25] is widely applied in various fields such as image processing and image generation. DNN (Deep Neural Network) [26, 27] is a structure composed of many layers and can automatically extract deep features. Jia et al. [28] performed rolling bearing failure diagnosis via DNN and said that characteristics could be collected from raw signals. Xu et al. [29] used PCA to reduce the size of these features. Eren et al. [30] built an error detection system using 1D-CNN (One-Dimension Convolutional Neural Network). Deng et al. [31] proposed the MSIQDE algorithm based on making use of the merits of the Mexh wavelet function. Shao et al. [32] proposed the Convolutional Deep Faith Network (CDBN) for rolling bearings. Thus, deep learning technologies have the ability to overcome the shortcomings inherent in traditional machine learning methods.

Among deep learning models, CNNs are one of the great ways to perform feature learning without prior knowledge. CNNs are suitable for feature learning because they can pass

signals periodically. However, CNNs have some drawbacks. Unlike methods like SVM or PCA, it requires a lot of training samples. Also, the filter size of each convolutional layer is fixed, so you cannot get various information. In addition, since a CNN with a general structure can only extract single scale features, useful information may be lost and domain shift problems may occur. Therefore, we proposed an improved CNN called MSCNN with different filter sizes at each convolution. This allows us to extract useful information in frequency domains with different resolutions. In addition, more powerful feature expressions can be learned than conventional CNNs, and the number of parameters and training time can be reduced [33, 34]. Also, training deep learning requires a lot of data. If there is not enough training data, it is difficult to expect good results. Data on rolling bearings are not always enough under real condition. In this paper, we have increased the amount of data by applying permutation and time-warping techniques.

This paper is configured as follows: Neural networks and the background of the proposed model are described in Section 2. Experiment and results are provided in Section 3. Finally, Section 4 presents a conclusion.

2. Background

2.1. Artificial Neural Network (ANN). The structure used in artificial neural networks [35] today was proposed by Frank Rosenblatt in 1958. Rosenblatt proposed a linear classifier called perceptron, which was a linear classifier structure that outputs 1 if the value is greater than 0 and -1 if it is less than 0 by adding the product of the inputs and weights and applying the activation function. Neurons have multiple inputs and one output. When each input is multiplied by a weight, the weight is multiplied by the next input, and the larger the weight, the more information is conveyed. A bias is added to the sum of the input value and the weight, and this bias represents the sensitivity of the neuron. Figure 1 describes the architecture of perceptron:

$$u = \sum_{i=1}^n (w_i \times x_i) + b, \quad (1)$$

where w represents the weight, x represents the input of neuron, and b represents the bias.

It was expected that the perceptron could create artificial intelligence like a real human, but in 1969, Minsky and Papert [36] proved the mathematical limitations of the perceptron, and people's expectations dwindled. According to them, the perceptron is only a simple linear classifier and cannot perform XOR classification. In other words, simple problems can be solved, but complex problems cannot be solved. Figure 2 describes the problem of OR, AND, and XOR.

In 1986, Rumelhart et al. [37] proposed a multilayered perceptron that overcomes the limitation of linear classifiers by adding a hidden layer. It proved that the XOR problem can be solved by using a concept of multilayered perceptron.

FIGURE 1: Architecture of perceptron.

MLP has a structure that is similar to a single perceptron, but by making the input/output characteristics of the intermediate layer and each unit nonlinear, it overcame the shortcomings of single perceptron by improving the network capability. In MLP, as the number of layers increases, the characteristics of decision regions formed by perceptron become more advanced. In Figure 3, we visualized the architecture of MLP.

2.2. Convolutional Neural Network (CNN). CNN was developed by Lecun and Bengio [38] in the 1990s as a neural network structure that classifies handwritten numbers and received great attention. It is one of the most popular deep learning algorithms. CNN is a model that reduces the number of parameters using convolution using spatial relations. The goal of extracting hidden features from the data is to learn several feature filters in the input data and then perform operations between the feature filters and the input data. Since the vibration signal is a time series vibration signal, 1D-CNN [39] was used. CNN mainly consists of input, convolution layer, pooling layer, fully connected layer, and output. The basic structure of 1D-CNN is represented in Figure 4.

The convolutional layer is a layer that learns the feature values of the input data and consists of multiple feature maps. Neurons in each feature map are connected to the local area of the previous functional map through a set of weights. This set of weights is called the convolution kernel. The result after performing convolution on the input feature map and the convolution kernel is passed to the activation function to form the next feature map layer. Functional maps are computed through weight sharing, reducing model complexity, and making network training easier. The forward propagation of the convolutional layer is as follows:

$$x_j^n = f \left(\sum_{i \in F_j} x_i^{n-1} \times k_{ij}^n + b_j^n \right), \quad (2)$$

where x_j^l is the output of the layer l , M_j is the selected feature map, x_j^{l-1} is the output of the layer $l-1$, k_{ij}^l is the weight of layer l , and b_j^l is the bias of layer l .

The pooling layer is usually placed between successive convolutional layers. It is about reducing the dimensionality

FIGURE 2: Problem of ANN.

FIGURE 3: Architecture of multilayered perceptron (MLP).

FIGURE 4: Architecture of One-Dimensional Convolutional Neural Network (1D-CNN).

FIGURE 5: Structure of maxpooling layer.

FIGURE 6: Framework of MSCNN.

FIGURE 7: Original signal (blue) and generated signal by time-warping (red).

FIGURE 8: Original signal (blue) and generated signal by time-warping (red).

FIGURE 9: Original signal (green) and generated signal by permutation (pink).

FIGURE 10: Original signal (green) and generated signal by permutation (pink).

TABLE 1: System specification.

Hardware environment	Software environment
CPU: Intel Core i7-8700 K, 3.7 GHz, six-core twelve threads, 16 GB	Windows TensorFlow 2.0 framework and Python 3.7
GPU: Geforce RTX 2080ti	

of the convolutional layer to do the extraction. The pooling layer uses feature vector values in feature maps for subsampling, so the most commonly used pooling methods are maxpooling and average pooling. In this paper, we used a maxpooling method that performs better in one-dimensional time series operations. The structure of the

maximum pooling layer is shown in Figure 5 and is as follows:

$$p_i^{h+1}(j) = \max_{(j-1)w+1 \leq t \leq jw} (h_i^n(t)), \quad (3)$$

FIGURE 11: Bearing simulator of CWRU.

TABLE 2: Description of data—CWRU.

Training samples	Validation samples	Test samples	Fault types	Fault diameters	Class ID
1600	192	240	Normal	—	0
1600	192	240	Ball	0.007	1
1600	192	240	Ball	0.014	2
1600	192	240	Ball	0.021	3
1600	192	240	IR	0.007	4
1600	192	240	IR	0.014	5
1600	192	240	IR	0.021	6
1600	192	240	OR	0.007	7
1600	192	240	OR	0.014	8
1600	192	240	Outer	0.021	9

TABLE 3: Result of accuracy test—CWRU.

Model	MSCNN (%)	1D-CNN (%)	2D-CNN (%)	MSCNN (%) (no augmentation)
Train accuracy	99.88	99.31	99.27	98.56
Test accuracy	99.79	99.22	98.50	98.35

where q_i^l is the output of the t th neuron in the i th feature map of the layer l , $t \in [(j-1)W+1, jW]$, W is the width of the pooled area, and $P_i^{l+1}(j)$ is the pooled value of the corresponding neuron in the layer $l+1$.

2.3. Multiscale Convolutional Neural Network (MSCNN). Since the input value of CNN is usually a raw signal, poor results can be obtained regardless of hyperparameter changes if there is insufficient useful information. A con-

volution is the most important method to analyze the signal, and the size of the convolution filter in 1D-CNN has a great influence on the performance. In 1D-CNN, the size of the convolution filter is a hyperparameter. Since the convolution layer uses a convolution filter of a fixed size, laying out the size of the convolution filter is a very difficult problem.

Also, there are also some issues with the classification. First, a large size of convolutional filter has a good resolution because it focuses on a low-frequency region but tends to ignore high-frequency information. Conversely, a small size of convolutional filter focuses on the frequency band but has a lower resolution. Second, if a convolution filter of the same size is used, other discriminatory features cannot be properly extracted.

To address this problem, researchers proposed a multiscale convolution neural network. Multiscale convolution extracts features from a vibration signal using several convolution filters of various scales. Our framework of the proposed model is described in Figure 6.

We have used three convolution filters with different widths in the convolution layer to extract features from the original data through two convolution layers and one maxpooling layer and obtain three different feature maps and then concatenate them. The 1×1 convolution reduces the depth and width of the networks without increasing computation resources. This structure made it possible to extract other distinct features from the original signal. After concatenation, the classification part consisted of two fully connected layers and a softmax layer. The softmax function is designed as the number of labels in two different datasets.

The proposed multiscale feature extraction utilizes three convolution filters with different widths in the convolution layer to extract features from the original data through two convolution layers and one maximum

FIGURE 12: Loss and accuracy curves—CWRU.

pooling layer and obtain three different feature maps, then connect them. This structure allows both low- and high-frequency information to be obtained from the original signal. The convolution layer of MSCNN is as follows [40]:

$$C_d^t = \max(0, \text{Concat}(c_1^t, c_2^t, c_3^t)), \quad (4)$$

where C_d^t represents the output feature map of the t th convolution layer of MSCNN with depth $d=3$ and c_1^t, c_2^t, c_3^t represents the feature maps after convolutions of MSCNN and can be seen as

$$\begin{cases} c_1^t = f_1^t * C^{[t-1]} + b_1^t, \\ c_2^t = f_2^t * C^{[t-1]} + b_2^t, \\ c_3^t = f_3^t * C^{[t-1]} + b_3^t. \end{cases} \quad (5)$$

Three convolution filters f_d^t are convolved with a $C^{[t-1]}$ feature map. b_d^t is the bias added to the feature map of the t th convolution layers. Each convolution layer is combined into a concatenation layer after convolution operations.

2.4. Data Augmentation. Deep learning increases the expressiveness of the model by increasing the parameters by stacking a lot of hidden layers. A huge amount of

training data is necessary to properly train a lot of parameters. However, it is not easy to extract a lot of data under real working conditions. In addition, the data should keep the quality high and varied enough to reflect reality. If a deep learning model without enough training data is performed to train the parameters, the overfitting problem usually occurs. Therefore, by increasing the absolute amount of data even in a small data set area through a data augmentation technique [41, 42], we acquired new data by applying artificial changes to the data. Data augmentation can process unexplored input and improve the generalization effect of deep learning models. The important thing about data augmentation is to meet your domain knowledge to keep your existing labels when generating new data. It also does not change the data label with minor changes. Data augmentation technology is often used for images, but data augmentation technology is applied to time series data.

In this paper, we have used two data augmentation techniques. Both techniques are made by the fact that a slight change in the action point can keep the label. Firstly, a time-warping technique changes the position of time samples by smoothly distorting the time interval between samples. In Figures 7 and 8, we visualized the original signal and the generated signal by time-warping using subplot and scatter plot.

Second, the permutation is a technique that randomly changes the location of an event. To confuse the location of the input data, it is a technique to create a new window

FIGURE 13: Confusion matrix of the proposed model (a), 1D-CNN (b), 2D-CNN (c), and proposed model without data augmentation (d)—CWRU.

by splitting the data into segments with the same length and then randomly changing the next segment. Figures 9 and 10 are diagrams visualizing the original data and the data generated by the permutation technique. Both methods have slight changes to the data, but no significant changes to the labels.

3. Experiment and Results

3.1. Experimental Configuration. Keras library was used with Tensorflow backend. We compared our proposed model with 2 CNN models. The system specification for the experiment is shown in Table 1.

3.2. Simulation Case 1: CWRU Bearing Dataset. To evaluate the performance of the proposed MSCNN, a bearing dataset from Case Western Reserve University Bearing Data Center was used, and the fault test bench is given in Figure 11.

The CWRU bearing dataset provides vibration signals generated by the simulator in normal and fail conditions. The 2hp electric motor, torque converter, and dynamometer are the main components, and the vibration signals were collected from the drive end, fan end, and accelerometer mounted in the housing. We used motor drive end bearing data sampled at 0hp, 1hp, 2hp, and 3hp. Each defect type is divided into 0.007, 0.014, and 0.021 inches, so there are 10 states in the data set. We divided the size

FIGURE 14: Bearing simulator of IMS.

TABLE 4: Description of data—IMS.

Training samples	Validation samples	Test samples	Fault types	Class ID
1500	192	240	Normal	0
1500	192	240	IR	1
1500	192	240	Roller element	2
1500	192	240	OR	3

of each signal segment by 400. Each state has 1600 samples, total of 10 states, so there are 16000 data. We divided 80% of the total data as training data, 20% as test data, and 20% of the remaining training data as validation data. The detailed description of data is in Table 2.

Two types of CNN models were compared to confirm the effectiveness of the proposed model. Both models are built as commonly used. Table 3 shows the results of an accuracy comparison of the proposed MSCNN with 1D-CNN, 2D-CNN, and the proposed model without increasing data.

Also, Figure 12 is a graph that visualizes the loss curve and accuracy curve for each model. In the loss curve, it can be seen that the proposed model is fine but settles more quickly. In addition, the accuracy curve shows that the proposed model is rapidly increasing in accuracy.

According to the result, the proposed model showed a better accuracy compared with others. The proposed model showed high accuracy of up to 1.4% and at least 0.53% in tests. There seems to be a slight difference, but this is significant because the 1% difference is not a small

number as the accuracy increases. In addition, there is a significant difference in accuracy even when data augmentation techniques are not used. This proves that splitting the convolution kernel into multiple scales works. In addition, we created a confusion matrix for each model for the reliability of the experiment.

As a second indicator, we used the confusion matrix. Although accuracy is the most intuitive metric, performance can be skewed on unbalanced datasets. There are four concepts you need to understand for the confusion matrix. The formula utilized for the confusion matrix is as below.

First, precision is the ratio of true to what the model classifies as true. Precision is expressed as

$$\text{Precision} = \frac{|TP|}{|TP| + |FP|}. \quad (6)$$

Recall is the proportion of true that the predicted model is true. The recall is expressed as

$$\text{Recall} = \frac{|TP|}{|TP| + |FN|}. \quad (7)$$

Accuracy is the most intuitive metric and is expressed as follows:

$$\text{Accuracy} = \frac{|TP| + |TN|}{|TP| + |FP| + |FN| + |TN|}. \quad (8)$$

FIGURE 15: Loss and accuracy curves—IMS.

TABLE 5: Result of accuracy test—IMS.

Model	MSCNN (%)	1D-CNN (%)	2D-CNN (%)	MSCNN (%) (no augmentation)
Train accuracy	99.64	99.33	99.03	98.57
Test accuracy	99.51	99.25	99.01	98.14

The F1 score is the harmonic mean, which allows you to accurately evaluate the performance of your model if your data labels are unbalanced. The F1 score can be expressed as

$$\text{F1 score} = 2 \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (9)$$

Figure 13 shows the results of the confusion matrix for each model. As a result of the experiment, as in the accuracy analysis, the proposed model showed the best performance, followed by 1D-CNN and 2D-CNN. Since the time series data is the input data, the performance of 1D-CNN appears to be better than that of 2D-CNN performance, followed by 1D-CNN and 2D-CNN. Since the time series data is the input data, the performance of 1D-CNN appears to be better than that of 2D-CNN.

3.3. Simulation Case 2: IMS Bearing Dataset. To add reliability to the evaluation of the proposed model, we used the bearing dataset provided by the Center for Intelligent Maintenance System (IMS), and the fault test bench is given in Figure 14.

Four rolling bearings were installed and operated with a rotational speed of 2000 rpm and a radial load of 6000 lbs. The raw signal was received by two accelerometers arranged vertically and horizontally. Therefore, there are a total of 4 types, including 3 fault states and normal states. In order to see the difference in the results, we conducted an experiment with the CWRU data set presented above and the number of samples made the same. The detailed description of data is in Table 4.

We compared the two types of CNN models and the proposed model as in the previous experiment. Table 5 shows the results of comparing the accuracy of the proposed model and a typical CNN model using IMS data. According to the results, the proposed model showed better accuracy compared to other models. The proposed model showed maximum 1.37%, minimum 0.26%, and higher accuracy in the test.

Also, Figure 15 is a graph that visualizes the loss curve and accuracy curve for each model. From the loss curve, it is shown that the proposed model is fine, similar to the CWRU experiment results, but settles more quickly. In addition, the accuracy curve shows that the proposed model is rapidly increasing in accuracy. The results of the confusion matrix for each model are shown in Figure 16.

FIGURE 16: Confusion matrix of proposed model (a), 1D-CNN (b), 2D-CNN (c), and proposed model without data augmentation (d)—IMS.

4. Conclusion

Diagnosis of bearing failures is very important in the industry. Knowing bearing failure in advance can reduce downtime, prevent financial losses, and prevent failures in advance. Raw vibration signals were collected from CWRU and IMS bearing data sets. However, in the real world, there is not always enough data, and without enough data, the deep learning model performs very poorly. Therefore, in this paper, data was generated using data augmentation techniques that are good for application to two types of time series data. Experimentation has shown that data augmentation techniques significantly improve accuracy. In addition, in order to overcome the shortcomings of CNN, we proposed a model that minimizes the information lost through each different convolution filter by configuring the con-

volution layer in multiscale and reduces parameters and training time. The proposed model not only extracts useful information better from the frequency domain with different resolutions than the conventional CNN but also enables more powerful feature expression learning. The proposed model showed better performance than the existing 1D-CNN and 2D-CNN. Future research will apply to not only bearing data but also data from other fields widely used in the industry. In addition, we plan to improve the structure of the model more efficient.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request (jpjeong@skku.edu).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the “Ministry of Science, ICT (MSIT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2018-0-01417) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) and the Smart Factory Technological R&D Program S2727115 funded by the Ministry of SMEs and Startups (MSS, Korea).

References

- [1] Q. Gao, C. Duan, H. Fan, and Q. Meng, “Rotating machine fault diagnosis using empirical mode decomposition,” *Mechanical Systems and Signal Processing*, vol. 22, no. 5, pp. 1072–1081, 2008.
- [2] B. Li, M. Y. Chow, Y. Tipsuwan, and J. C. Hung, “Neural-network-based motor rolling bearing fault diagnosis,” *IEEE Transactions on Industrial Electronics*, vol. 47, no. 5, pp. 1060–1069, 2000.
- [3] Z. K. Peng, W. Tse Peter, and F. L. Chu, “A comparison study of improved Hilbert-Huang transform and wavelet transform: application to fault diagnosis for rolling bearing,” *Mechanical Systems and Signal Processing*, vol. 19, no. 5, pp. 974–988, 2005.
- [4] P. M. Frank, S. X. Ding, and T. Marcu, “Model-based fault diagnosis in technical processes,” *Transactions of the Institute of Measurement and Control*, vol. 22, no. 1, pp. 57–101, 2016.
- [5] S. Simani, C. Fantuzzi, and R. J. Patton, “Model-based fault diagnosis techniques,” *Model-Based Fault Diagnosis in Dynamic Systems Using Identification Techniques*, pp. 19–60, 2003.
- [6] W. Hongm, C. Tian-you, J. L. Ding, and M. Brown, “Data driven fault diagnosis and fault tolerant control: some advances and possible new directions,” *Acta Automatica Sinica*, vol. 35, no. 6, pp. 739–747, 2009.
- [7] S. X. Ding, *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*, Springer Science & Business Media, 2008.
- [8] R. Isermann, “Model-based fault-detection and diagnosis - status and applications,” *Annual Reviews in Control*, vol. 29, no. 1, pp. 71–85, 2005.
- [9] P. M. Frank and X. Ding, “Frequency domain approach to optimally robust residual generation and evaluation for model-based fault diagnosis,” *Automatica*, vol. 30, no. 5, pp. 789–804, 1994.
- [10] A. K. Jalan and A. R. Mohanty, “Model based fault diagnosis of a rotor-bearing system for misalignment and unbalance under steady-state condition,” *Journal of Sound and Vibration*, vol. 327, no. 3–5, pp. 604–622, 2009.
- [11] R. Isermann, “Model-based fault detection and diagnosis - status and applications,” *IFAC Proceedings Volumes*, vol. 37, no. 6, pp. 49–60, 2004.
- [12] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, “Feature extraction,” *Foundations and Applications*, vol. 207, 2008.
- [13] H. Liu and H. Motoda, *Feature Extraction, Construction and Selection: A Data Mining Perspective*, vol. 453, Springer Science & Business Media, 1998.
- [14] B.-C. Kuo and D. A. Landgrebe, “Nonparametric weighted feature extraction for classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 5, pp. 1096–1105, 2004.
- [15] J. Mao and A. K. Jain, “Artificial neural networks for feature extraction and multivariate data projection,” *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 296–317, 1995.
- [16] A. Widodo and B.-S. Yang, “Support vector machine in machine condition monitoring and fault diagnosis,” *Mechanical Systems and Signal Processing*, vol. 21, no. 6, pp. 2560–2574, 2007.
- [17] J. A. K. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.
- [18] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1–3, pp. 37–52, 1987.
- [19] R. Bro and A. K. Smilde, “Principal component analysis,” *Analytical Methods*, vol. 6, no. 9, pp. 2812–2831, 2014.
- [20] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*, MIT press, 1995.
- [21] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, “Machine learning,” *Neural and Statistical Classification*, vol. 13, no. 1994, pp. 1–298, 1994.
- [22] M. I. Jordan and T. M. Mitchell, “Machine learning: trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015.
- [23] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT press, 2012.
- [24] I. Goodfellow, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1, no. 2, 2016, MIT press, Cambridge, 2016.
- [25] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [26] A. Canziani, A. Paszke, and E. Culurciello, *An Analysis of Deep Neural Network Models for Practical Applications*, 2016.
- [27] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, “Evaluating the visualization of what a deep neural network has learned,” *IEEE transactions on neural networks and learning systems*, vol. 28, no. 11, pp. 2660–2673, 2016.
- [28] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, “Deep neural networks: a promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data,” *Mechanical Systems and Signal Processing*, vol. 72, pp. 303–315, 2016.
- [29] F. Xu, G. Guohua, X. Kong, P. Wang, and K. Ren, “Object tracking based on two-dimensional PCA,” *Optical Review*, vol. 23, no. 2, pp. 231–243, 2016.
- [30] L. Eren, T. Ince, and S. Kiranyaz, “A generic intelligent bearing fault diagnosis system using compact adaptive 1D CNN classifier,” *Journal of Signal Processing Systems*, vol. 91, no. 2, pp. 179–189, 2019.
- [31] W. Deng, H. Liu, J. Xu, H. Zhao, and Y. Song, “An improved quantum-inspired differential evolution algorithm for deep belief network,” *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 7319–7327, 2020.
- [32] H. Shao, H. Jiang, H. Zhang, and T. Liang, “Electric locomotive bearing fault diagnosis using a novel convolutional deep belief

- network,” *IEEE Transactions on Industrial Electronics*, vol. 65, no. 3, pp. 2727–2736, 2017.
- [33] Y. Song, D. Wu, W. Deng et al., “MPPCEDE: multi-population parallel co-evolutionary differential evolution for parameter optimization,” *Energy Conversion and Management*, vol. 228, article 113661, 2020.
 - [34] W. Deng, J. Xu, Y. Song, and H. Zhao, “Differential evolution algorithm with wavelet basis function and optimal mutation strategy for complex optimization problem,” *Applied Soft Computing*, vol. 100, article 106724, 2020.
 - [35] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain,” *Psychological review*, vol. 65, no. 6, p. 386, 1958.
 - [36] M. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*, MIT press, 2017.
 - [37] D. E. Rumelhart, G. E. Hinton, and J. L. McClelland, “A general framework for parallel distributed processing,” *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1, no. 26, pp. 45–76, 1986.
 - [38] Y. LeCun and Y. Bengio, “Convolutional networks for images, speech, and time series,” *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10, p. 1995, 1995.
 - [39] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, “Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks,” *Journal of Sound and Vibration*, vol. 388, pp. 154–170, 2017.
 - [40] H. T. Mustafa, J. Yang, and M. Zareapoor, “Multi-scale convolutional neural network for multi-focus image fusion,” *Image and Vision Computing*, vol. 85, pp. 26–35, 2019.
 - [41] J. Salamon and J. P. Bello, “Deep convolutional neural networks and data augmentation for environmental sound classification,” *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.
 - [42] T. T. Um, F. M. J. Pfister, D. Pichler et al., “Data augmentation of wearable sensor data for Parkinson’s disease monitoring using convolutional neural networks,” in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 216–220, ICMI '17: Proceedings of the 19th ACM International, 2017.