Multiresolution Hypergraph Neural Network for Intelligent Fault Diagnosis

Xunshi Yan (闫循石), Yang Liu (刘洋), and Chen-An Zhang (张陈安)

Abstract—**Intelligent fault diagnosis has made significant progress, thanks to machine learning, particularly deep-learning algorithms. However, most machine-learning algorithms treat samples as independent and ignore the correlations between samples that contain valuable information for creating discriminative features. In recent years, graph neural networks have increased diagnostic performance by capturing the correlation between samples according to establishing the inherent structure of data, but they also suffer from two shortcomings. First, a simple graph only represents pairwise relationships of samples and cannot depict complex higher-order structures. Second, the generated graph structure is insufficient to characterize the data without an explicit structure. To address the above two issues, this article proposes a multiresolution hypergraph neural network, a novel algorithm that can discover higher-order complex relationships between samples, and mine the structure hidden in data by establishing and fusing hypergraph structures at multiple resolutions. Experiments are conducted on three datasets to demonstrate the effectiveness of the proposed algorithm.**

Index Terms—**Fault diagnosis, graph convolutional network (GCN), hypergraph, hypergraph neural network (HGNN), multiresolution.**

I. Introduction

WITH the development of artificial intelligence technology, intelligent fault diagnosis has become a hot topic in the engineering field. As a typical pattern recognition issue, fault types are identified employing signals captured by vibration, current, or other kinds of sensors installed on the rotating machinery or its components such as bearings and gearboxes. Due to the scarcity of fault samples, as well as their nonlinearity and nonstationarity, it is a great challenge to extract effective features of faults. Therefore, many researchers pay much attention to obtaining more information from fault signals to create discriminative features.

End-to-end learning [1], [2], multisensor fusion [3]–[5], and transfer learning [6], [7] techniques bring more information to the fault feature representation than manually designed features, but all regard the samples as independent and ignore the correlation between samples, which is an important factor in distinguishing fault types. In recent years, deep-learning-based graph neural networks (GNNs), such as CheyGNN [8], graph convolutional networks (GCNs) [9], GraphSAGE [10], and graph attention networks [11], have been developed successively and effectively capture the correlation between samples and improve the recognition performance of the data with graph structures.

GNNs have been applied in fault diagnosis of rotating machinery [12] and the related literature can be clustered into two categories. In the first category, each sample is treated as a node on the graph, and the fault diagnosis task is transformed into a node classification problem. Gao et al. [13] considered fault samples as nodes on a graph and directly classified faults in a semisupervised manner using GCNs. Li et al. [14] set up three graph receptive fields and used GCNs to fuse the three graphs to form fault representations. Yang et al. [15] proposed to modify GCNs into a multibranch structure, where each branch was trained using a graph generated from one sensor's data. The output of the branches was fused through a fully connected layer to achieve diagnosis results. To enhance the recognition accuracy, Zhou et al. [16] developed a dynamic graph model based on GCNs. A graph is manually constructed from samples, the GCN was trained for a specified number of epochs, and then the graph was reconstructed using the updated features. The process is repeated several times until the updated graph approximated the dataset's inherent structure.

In the second category of literature, each fault sample is considered a graph, and the fault diagnosis task is converted to a graph classification problem. Li et al. [17] transformed a sample of 1024 points into a graph consisting of 1024 nodes, where the weights of the edges were determined by the temporal distance between two nodes, and the graph isomorphic network was used for graph classification. Yang et al. [18] segmented each sample into pieces to build a spatial–temporal graph based on the frequency bands of the short-time Fourier transform spectrum and then applied the Laplacian transform to the graph to produce graph representation for classification. Zhang et al. [19] rearranged the fault signals into a 60×60 node matrix, considering the pairwise relationship between the central node and its eight neighbors, and then used GNNs for classification.

However, each edge in the graph connects two nodes, indicating pairwise relationships between nodes, and is unable to explain higher-order relationships or complex correlations

Manuscript received 18 July 2022; revised 12 September 2022; accepted 26 September 2022. Date of publication 5 October 2022; date of current version 20 October 2022. This work was supported in part by the National Science and Technology Major Project of China under Grant ZX069 and in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA17030100. The Associate Editor coordinating the review process was Dr. Siliang Lu. *(Corresponding author: Xunshi Yan.)*

Xunshi Yan is with the Institute of Nuclear and New Energy Technology, Tsinghua University, Beijing 100084, China (e-mail: yanxs@tsinghua.edu.cn).

Yang Liu is with the State Key Laboratory of High Temperature Gas Dynamics, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Engineering Science, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: liuyang2@imech.ac.cn).

Chen-An Zhang is with the State Key Laboratory of High Temperature Gas Dynamics, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China (e-mail: zhch_a@imech.ac.cn).

Digital Object Identifier 10.1109/TIM.2022.3212532

between multiple nodes. Furthermore, unlike social networks, citation networks, and so on, the data for fault diagnosis tasks has no explicit graph structure and the graphs are generated manually. The performance of GNNs is heavily dependent on whether the constructed graphs can truly capture the correlations hidden in data, however, the existing methods limit the ability of constructed graphs to represent the inherent structure in fault data.

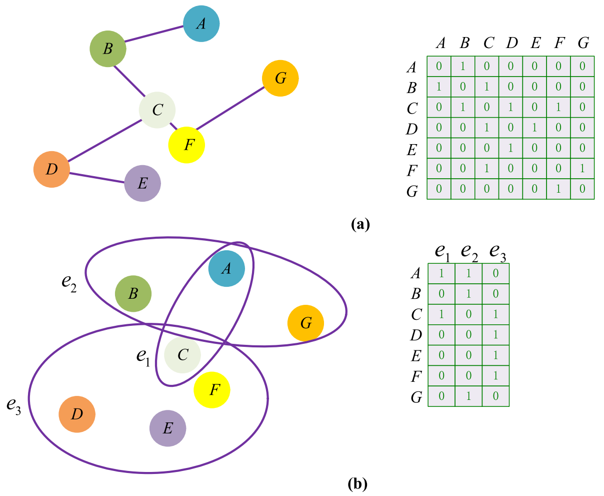
This article proposes a novel multiresolution hypergraph neural network (MrHGNN) to tackle the two aforementioned challenges. A hypergraph is adopted to build higher-order complex relationships between fault samples. A multiresolution hypergraph (MrHG) with the idea of coarse-to-fine descriptions is to approximate the potential structure in fault data, and a hypergraph neural network (HGNN) based on MrHG is designed for fault classification. The contributions of this article are primarily three folds.

1) Instead of a simple graph, a hypergraph is utilized to model the relationship between fault samples and mine higher-order correlations of multiple nodes, which is the first time that a hypergraph has been used in a mechanical fault diagnosis task to our best knowledge.
2) MrHG is proposed. A series of hypergraphs are constructed from samples with varying resolutions to describe the inherent structure of fault data from coarse to fine. The structure information in fault data can be represented by fusing the multiple hypergraphs, thus compensating for the inadequate capability of traditional graph generation methods.
3) A neural network on MrHG is designed to achieve fault classification based on the hypergraph learning mechanism, thus establishing a framework for hypergraph-based fault diagnosis research, which is validated on three datasets.

The rest of this article is mainly divided into four sections. Section II introduces the basic theory of the hypergraph and HGNN. The proposed algorithm is detailed in Section III. The experiments are presented in Section IV to verify the effectiveness of the proposed algorithm on three datasets. Section V concludes the whole paper and shows the future research directions.

II. Basic Theory of Hypergraph Neural Network

A. Hypergraph



	A	B	C	D	E	F	G
A	0	1	0	0	0	0	0
B	1	0	1	0	0	0	0
C	0	1	0	1	0	1	0
D	0	0	1	0	1	0	0
E	0	0	0	1	0	0	0
F	0	0	1	0	0	0	1
G	0	0	0	0	0	1	0

	e_1	e_2	e_3
A	1	1	0
B	0	1	0
C	1	0	1
D	0	0	1
E	0	0	1
F	0	0	1
G	0	1	0

Fig. 1. Comparison between a simple graph and a hypergraph. (a) Graph and adjacent matrix. (b) Hypergraph and incident matrix.

Hypergraph $G = (V, E, \mathbf{W})$ is a kind of data structure that represents multiple complex correlations and consists of nodes $V = \{v_1, v_2, v_3, \ldots, v_N\}$, hyperedges $E = \{e_1, e_2, e_3, \ldots, e_M\}$, and hyperedge weight $\mathbf{W}$. In contrast to a simple graph, each hyperedge can connect more than two nodes to model higher-order relationships, suggesting that these nodes as a subset share common properties. For example, considering the citation network as a simple graph, each paper is treated as a node on the graph, and each edge in the graph indicates that two papers have coauthors; however, if the citation network is depicted as a hypergraph, a hyperedge can connect all papers with the same coauthors. Hence, hypergraphs can describe complex higher-order relationships beyond pairwise correlation between two nodes and discover the common properties of nodes on the same hyperedge.

Similar to the adjacent matrix used to represent a simple graph, the incident matrix $\mathbf{H} \in R^{|V|\times|E|}$ describes the connectivity between nodes in the hypergraph. Each column and each row of $\mathbf{H}$ corresponds to one hyperedge and one node, respectively, with the element value set to 1 or 0, indicating whether the corresponding node lies on the corresponding hyperedge, as illustrated in

$$\mathbf{H}(v, e) = \begin{cases} 1, & \text{if } v \in e \\ 0, & \text{if } v \notin e. \end{cases} \tag{1}$$

The weight matrix $\mathbf{W} = \text{diag}(w_1, w_2, \ldots, w_{|E|}) \in R^{|E|\times|E|}$ is a diagonal matrix and represents the weight of each hyperedge. The larger the weight w_i, the greater the influence of the hyperedge e_i. The diagonal elements d_e and d_v of the hyperedge degree matrix $\mathbf{D}_E$ and node degree matrix $\mathbf{D}_V$ denote the number of nodes connected to each hyperedge and the number of hyperedges met in each node, respectively, as

$$d_e = \sum_{v\in V} \mathbf{H}(v, e) \tag{2a}$$

$$d_v = \sum_{e\in E} w_e * \mathbf{H}(e, v). \tag{2b}$$

A hypergraph is a generalized form of a simple graph. When all hyperedges satisfy $d_e = 2$, a hypergraph degenerates to a simple graph. Due to $d_e \geq 2$, hypergraphs can describe more complex relationships than simple graphs and generate powerful node representations based on the information aggregated from the nodes on the hyperedges. Fig. 1 depicts the comparison of a simple graph and a hypergraph as well as the meaning of the adjacent matrix and incident matrix.

B. Hypergraph Learning

Similar to graph learning, hypergraph learning is employed to solve three basic problems [20], including node classification, hyperedge prediction, and hypergraph classification. In this article, the task of fault diagnosis is treated as a node classification problem. Hypergraph learning is widely used in 3-D object recognition [21], action recognition [22], document classification [23], image retrieval [24], and medical image analysis [25].

Based on hypergraph cut theory, Zhou [26] first proposed a regularization framework for hypergraph node classification, which led to the development of hypergraph learning during the next ten years as

$$\arg\min_{f}\left\{R_{emp}(f)+\mu\Omega(f)\right\} \tag{3}$$

where $R_{emp}(f)$ is a supervised loss and $f(\cdot)$ is a classification function. The number $\mu > 0$ is the regularization parameter. $\Omega(f)$ is a regularizer on the hypergraph. Based on the normalized hypergraph cut, $\Omega(f)$ is defined as

$$\Omega(f) = \frac{1}{2}\sum_{e\in E}\sum_{\{u,v\}\in V}\frac{w(e)h(u,e)h(v,e)}{d_e}\left(\frac{f(u)}{\sqrt{d_v(u)}}-\frac{f(v)}{\sqrt{d_v(v)}}\right)^2. \tag{4}$$

Let $\Theta = \mathbf{D}_V^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_V^{-1/2}$ and $\Delta = \mathbf{I}-\Theta$ a simplified form of $\Omega(f)$ and is written as

$$\Omega(f) = f^T\Delta f \tag{5}$$

where $\Delta \in R^{N\times N}$ is named as hypergraph Laplacian as a positive semidefinite matrix. Perform eigendecomposition $\Delta = \Phi^T\Lambda\Phi$ and the orthonormal eigenvectors and eigenvalues are got as $\Phi = [\phi_1, \phi_2, \ldots, \phi_N]$ and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N)$.

The above approach has a high cost when dealing with large-scale hypergraphs because it involves the eigendecomposition of matrices. Inspired by the creation of graph convolution in GNNs, designing a low-cost hypergraph convolution and establishing a neural network becomes a solution to the hypergraph learning challenge. Similar to the definition of graph convolution, the spectral convolution on the hypergraph can be defined as

$$h * x = \Phi\left(\left(\Phi^T h\right)\odot\left(\Phi^T x\right)\right) = \Phi h(\Lambda)\Phi^T x \tag{6}$$

where $\odot$ represents the element-wise product and h is a hypergraph convolution filter.

Due to the high computational cost of (6) which still needs to eigendecomposition of a large matrix, according to the operation on graph convolution in [8], Chebyshev polynomials are used to parameterize $h(\Lambda)$ and the convolution of a signal on the hypergraph is modified as

$$h * x = \sum_{k=0}^{K}\theta_k T_k\left(\tilde{\Delta}\right)x \tag{7}$$

where $\tilde{\Delta} = (2/\lambda_{\max})\Delta - \mathbf{I}$ is scaled hypergraph Laplacian and $T_k(\tilde{\Delta})$ is the Chebyshev polynomials of order k.

Inspired by [9], Feng [27] further let $K = 1$ and simplified (7) as

$$\begin{aligned} h * x &= \theta_0 x - \theta_1\Theta x \\ &= \theta\mathbf{D}_V^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_E^{-1}\mathbf{H}^T\mathbf{D}_V^{-1/2}x \end{aligned} \tag{8}$$

where $\theta_1 = -(1/2)\theta$ and $\theta_0 = (1/2)\theta\mathbf{D}_V^{-1/2}\mathbf{H}\mathbf{D}_E^{-1}\mathbf{H}^T\mathbf{D}_V^{-1/2}$, both of which are replaced by single parameter θ to avoid overfitting. $\mathbf{W}$ is initialized as $\mathbf{I}$.

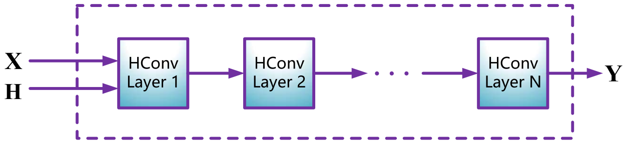


Fig. 2. HGNN. HConv represents hypergraph convolution.

Let the hypergraph node features as $\mathbf{X} \in R^{|V|\times|C_1|}$, the hypergraph convolution is reformed as

$$\mathbf{Y} = \sigma\left(\mathbf{D}_V^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_E^{-1}\mathbf{H}^T\mathbf{D}_V^{-1/2}\mathbf{X}\theta\right) \tag{9}$$

where $\mathbf{Y}$ is the updated node features after hypergraph convolution and θ is a learnable parameter. σ is an activation function.

C. HGNN

HGNN first proposed by [27] consists of a series of a hypergraph convolution layer (see Fig. 2), and each layer is denoted as

$$\mathbf{X}^{(l+1)} = \sigma\left(\mathbf{D}_V^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_V^{-1/2}\mathbf{X}^{(l)}\theta^{(l)}\right) \tag{10}$$

where $\mathbf{X}^{(l)} \in R^{|V|\times|C_1|}$ and $\mathbf{X}^{(l+1)} \in R^{|V|\times|C_2|}$ are the input and output of the lth hypergraph convolution layer. The learnable parameter is $\theta^{(l)} \in R^{|C_1|\times|C_2|}$. If the lth layer of HGNN is not the last layer, C_2 is preset as the hidden dimension.

The input of HGNN is initialized node features $\mathbf{X} \in R^{|V|\times d}$ and incident matrix $\mathbf{H} \in R^{|V|\times|E|}$, where d is the dimension of node features. Softmax is adopted as the activation function of the last layer in HGNN and the final output is $\mathbf{Y} \in R^{|V|\times C}$, where C is the number of fault types. The hidden dimension in each layer is defined in advance. The cross-entropy function is chosen as the loss function.

D. Discussion

In our research, hypergraph learning is applied to the fault diagnosis of rotating machinery, mainly for the following three reasons.

First, fault samples are cut from signals captured during the operation of the machines. Due to the continuity and periodicity of the signals, there are potential complex correlations between the samples. By constructing a hypergraph with signal pieces as nodes, the complex correlations can be described to fully mine the characteristic information between samples.

Second, compared with the problems with a large number of samples such as images and text, the cost of acquiring fault signals is high, resulting in the scarcity of fault samples. A crucial technical route is to extract additional information from limited fault samples. Most machine-learning algorithms, including convolutional neural networks (CNNs), obey the assumption of independent and identical distribution (IID), ignoring the correlations between data, which is helpful for improving diagnostic accuracy. Hypergraph learning can break the assumption of IID and capture the correlation between samples, thereby improving diagnostic performance.

Third, unlike graph-based methods, hyperedges in a hypergraph can connect more than two nodes and describe the high-order correlation between samples. The common attributes shared among multiple samples are extracted layer by layer through hypergraph learning to obtain a more effective feature representation.

III. Proposed Algorithm

In the fault diagnosis task, the fault samples are unstructured, making it challenging to use a single manually designed hypergraph to characterize the structure hidden in the data. To address the above issue, MrHG is proposed to model data structure between fault samples from coarse to fine, and a corresponding neural network based on MrHG is developed for fault identification.

A. Hypergraph Generation

In this study, the fault diagnosis task is considered a node classification problem, where each fault sample acts as a node on a hypergraph. Since there is no explicit structure among fault samples, the hypergraph should be generated from the samples first.

The signals gathered by the machine's sensors are split into a set of equal-length samples. Because the raw samples have a limited ability to discriminate faults and are prone to noise interference, they are processed by fast Fourier transform (FFT) as the initial node representations which are normalized with the minimum–maximum way to unify the feature value to [0,1].

To construct the hypergraph, it is critical to establish the connection between nodes to determine the hyperedges. For each sample, K-nearest neighbor (KNN) is employed to create hyperedges. The sample and its K neighboring samples in Euclidean space are grouped to form a hyperedge, resulting in one hypergraph containing N hyperedges. The incident matrix $\mathbf{H} \in R^{N\times N}$ can be written as follows:

$$\mathbf{H}_{ij} = \begin{cases} 1, & x_i \in KNN(x_j) \text{ or } i = j \\ 0, & \text{others} \end{cases} \tag{11}$$

where N is the number of samples in the dataset, and $KNN(x_j)$ represents the K samples closest to the central node sample x_j.

To simplify the problem, all hyperedges are assigned to the same weights, that is, $\mathbf{W} = \mathbf{I} \in R^{N\times N}$.

B. MrHG

To take benefit of the hypergraph and enhance its effectiveness, it is required to design a hypergraph that approximates the inherent structure hidden in the data without implicit connections between samples. On the premise of the determined node feature representation, the raw data $\mathbf{X} = \{x_1, x_2, x_3, \ldots, x_N\}$ is processed using the multiresolution features technique. The raw samples are resampled at various sampling rates as

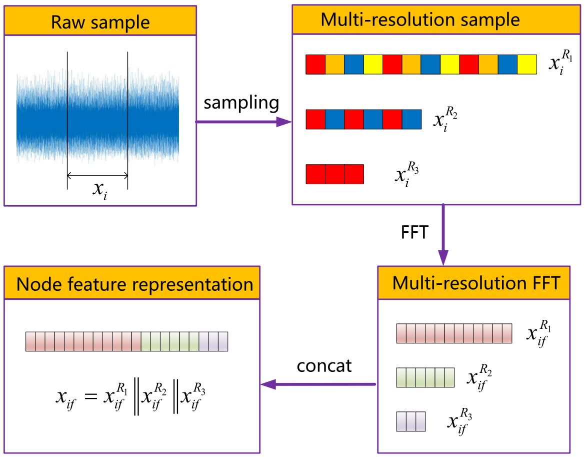


Fig. 3. Node feature representation with multiple resolutions.

$$\begin{cases} \mathbf{X}^{R_1} = \left\{x_1^{R_1}, x_2^{R_1}, x_3^{R_1}, \ldots, x_N^{R_1}\right\} \\ \mathbf{X}^{R_2} = \left\{x_1^{R_2}, x_2^{R_2}, x_3^{R_2}, \ldots, x_N^{R_2}\right\} \\ \vdots \\ \mathbf{X}^{R_s} = \left\{x_1^{R_s}, x_2^{R_s}, x_3^{R_s}, \ldots, x_N^{R_s}\right\} \end{cases} \tag{12}$$

where $R_1, R_2, \ldots, R_s$ are various sampling rates and R_1 denotes the basic sampling rate, $R_2 = (1/2)R_1, R_3 = (1/4)R_1, \ldots, R_s = (1/2^{s-1})R_1$. $\mathbf{X}^{R_1}$ is the raw dataset $\mathbf{X}$, and $\mathbf{X}^{R_p}$ is the resampling result of $\mathbf{X}$ under sampling rate R_p, then the raw dataset turns into s subdatasets $\{\mathbf{X}^{R_1}, \mathbf{X}^{R_2}, \mathbf{X}^{R_3}, \ldots, \mathbf{X}^{R_s}\}$.

Perform FFT on $\{\mathbf{X}^{R_1}, \mathbf{X}^{R_2}, \mathbf{X}^{R_3}, \ldots, \mathbf{X}^{R_s}\}$ and obtain

$$\begin{cases} \mathbf{X}_f^{R_1} = \left\{x_{1f}^{R_1}, x_{2f}^{R_1}, x_{3f}^{R_1}, \ldots, x_{Nf}^{R_1}\right\} \\ \mathbf{X}_f^{R_2} = \left\{x_{1f}^{R_2}, x_{2f}^{R_2}, x_{3f}^{R_2}, \ldots, x_{Nf}^{R_2}\right\} \\ \vdots \\ \mathbf{X}_f^{R_s} = \left\{x_{1f}^{R_s}, x_{2f}^{R_s}, x_{3f}^{R_s}, \ldots, x_{Nf}^{R_s}\right\} \end{cases} \tag{13}$$

where $x_{if}^{R_p}$ denotes the FFT feature representation of the ith sample under sampling rate R_p.

Define the node feature representation $\mathbf{X}_f = \{x_{if}, 1 \le i \le N\}$, where $x_{if} = x_{if}^{R_1} \| x_{if}^{R_2} \| x_{if}^{R_3} \ldots \| x_{if}^{R_s}$ and $\cdot\|\cdot$ is the concatenation operation. Here, the nodes are denoted by the union of multiresolution features (see Fig. 3).

Based on the referred hypergraph generation method, $\mathbf{X}_f^{R_1}, \mathbf{X}_f^{R_2}, \ldots, \mathbf{X}_f^{R_s}$ are transformed to s hypergraphs as subhypergraphs, respectively, and the corresponding incident matrices are $\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_s \in R^{N\times N}$. The s subhypergraphs are concatenated into one hypergraph as MrHG. The incident matrix of MrHG is

$$\mathbf{H} = \mathbf{H}_1 || \mathbf{H}_2 || \ldots || \mathbf{H}_s \tag{14}$$

where $\mathbf{H} \in R^{N\times Ns}$.

Fig. 4 expresses the procedure of the generation of MrHG. Since MrHG is made up of s subhypergraphs and Ns

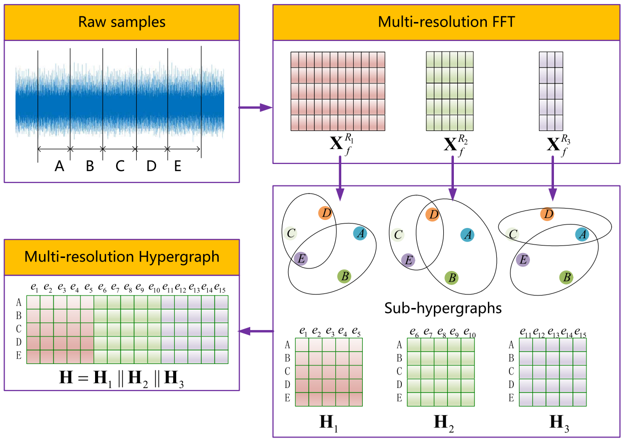

Raw samples
A B C D E
Multi-resolution FFT
$\mathbf{X}_f^{R_1}$
$\mathbf{X}_f^{R_2}$
$\mathbf{X}_f^{R_3}$
Multi-resolution Hypergraph
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12} e_{13} e_{14} e_{15}
$\mathbf{H} = \mathbf{H}_1 \| \mathbf{H}_2 \| \mathbf{H}_3$
Sub-hypergraphs
$\mathbf{H}_1$
$\mathbf{H}_2$
$\mathbf{H}_3$

Fig. 4. Generation of MrHG.

hyperedges that are built from raw signals of various resolutions, it depicts the correlation between samples from coarse to fine to approach the inherent structure hidden in the data.

C. MrHGNN

MrHGNN aims to implement MrHG learning by stacking a series of hypergraph convolution layers. In our study, we fix the number of layers of hypergraph convolution to 2. The inputs to MrHGNN are the incident matrix $\mathbf{H}$ and the node representation $\mathbf{X}_f$ with multiresolution features. In the first layer of MrHGNN, the learned parameter is $\theta^{(1)} \in R^{D\times h}$, and the activation function is the LeakyReLU function, where D is the node feature dimension and h is the output dimension defined as the hidden dimension. In the second layer, the learned parameter is $\theta^{(2)} \in R^{h\times C}$, where C is the number of fault types and Softmax is adopted as the activation function. The final output of MrHGNN is computed as (15)

$$\mathbf{X}^{(2)} = \text{Softmax}\big(\text{HConv}\big(\text{LeakyReLU}\big(\text{HConv}\big(\mathbf{X}_f, \mathbf{H}\big), \mathbf{H}\big)\big)\big) \tag{15}$$

where $\mathbf{X}^{(2)}$ is the final output of MrHGNN.

D. Pipeline

The pipeline of the proposed algorithm is depicted in Fig. 5 and the process is listed as follows.

1) Collect the fault signals acquired from the machine's sensors and cut them into N equal-length samples to form the dataset $\mathbf{X}$.
2) Resample $\mathbf{X}$ with different sampling rates $R_1, R_2, \ldots, R_s$, transform each sample into a group of subsamples with various resolutions, and form subdatasets $\{\mathbf{X}^{R_1}, \mathbf{X}^{R_2}, \mathbf{X}^{R_3}, \ldots, \mathbf{X}^{R_s}\}$.
3) Perform FFT on samples in $\{\mathbf{X}^{R_1}, \mathbf{X}^{R_2}, \mathbf{X}^{R_3}, \ldots, \mathbf{X}^{R_s}\}$, respectively, and obtain $\{\mathbf{X}_f^{R_1}, \mathbf{X}_f^{R_2}, \mathbf{X}_f^{R_3} \ldots, \mathbf{X}_f^{R_s}\}$. Node representation is $\mathbf{X}_f = \{x_{if}, 1 \leq i \leq N\}$.
4) Construct s hypergraphs from $\{\mathbf{X}_f^{R_1}, \mathbf{X}_f^{R_2}, \mathbf{X}_f^{R_3} \ldots, \mathbf{X}_f^{R_s}\}$ base on KNN principle and get the corresponding incident matrices $\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_s$. Form the incident matrix of MrHG as $\mathbf{H} = \mathbf{H}_1 || \mathbf{H}_2 || \ldots || \mathbf{H}_s$.
5) Build a two-layer MrHGNN, taking $\mathbf{X}_f$ and $\mathbf{H}$ as inputs.
6) Set a part of the samples for training and the other as testing samples, then train MrHGNN with hypergraph learning.
7) Test the trained model and get the corresponding fault types of testing samples.

E. Discussion

After exhibiting the principle and process of the algorithm, there are still some details worth discussing.

First, MrHGNN is a node classification algorithm in essence where each fault sample is viewed as a node on the hypergraph. The multiresolution feature vectors of all nodes on the hypergraph and the incident matrix of MrHG serve as the two inputs of MrHGNN. As a type of neural network, MrHGNN is composed of a series of hypergraph convolution layers, and the backpropagation method is used to optimize the parameters.

Second, from the perspective of spatial graph approaches, each hypergraph convolution layer implements a smooth operation on the hypergraph, which aggregates the information of the neighbor nodes on the hyperedge. The performance of MrHGNN will decline as the structure becomes deeper due to the consistency of all node features in oversmoothing. Therefore, the number of hypergraph convolution layers in MrHGNN is fixed to 2, which is consistent with ref [9], [27].

Third, there is only one matrix parameter to learn in each layer of MrHGNN. Regarding the time complexity analysis, according to ref [23], the computation upper bound of each hypergraph convolution layer is $O(2N^2M + 2M^2N)$, where N is the number of nodes and M represents the number of hyperedges. In MrHGNN, $M = Ns$, hence the computation upper bound turns to be $O(2N^3(s + s^2))$. In practice, the algorithm will run much faster because the matrix is sparse and the computation operations are matrix multiplication that can run on GPU.

IV. Experiments

In this section, a series of experiments are carried out to verify the effectiveness of the proposed algorithm.

A. Datasets

Three datasets including CWRU, Paderborn, and AMB-Y2 datasets are introduced. The first two datasets are often employed in many literature [28], while the third one is acquired from an industrial machine equipped with active magnetic bearings [13].

The CWRU dataset [29] is widely used in fault diagnosis, and the data is collected from a horizontal rotating machine (see Fig. 6). In this article, 10 data files are selected to create the dataset, which corresponds to 10 classes: 1) normal; 2) inner race-D0.007; 3) inner race-D0.014; 4) inner race-D0.021; 5) ball-D0.007; 6) ball-D0.014; 7) ball-D0.021; 8) outer race-D0.007; 9) outer race-D0.014; and 10) outer race-D0.021, for example, where the inner race-D0.007 represents that fault occurs in the inner race of the bearing and the fault diameter size is 0.007 In. Ninety-four training samples and 24 testing samples are cut from a file with no overlap between samples. The sample length is 1024 points and the sampling rate is 12k. The CWRU dataset setting is detailed in Table I.

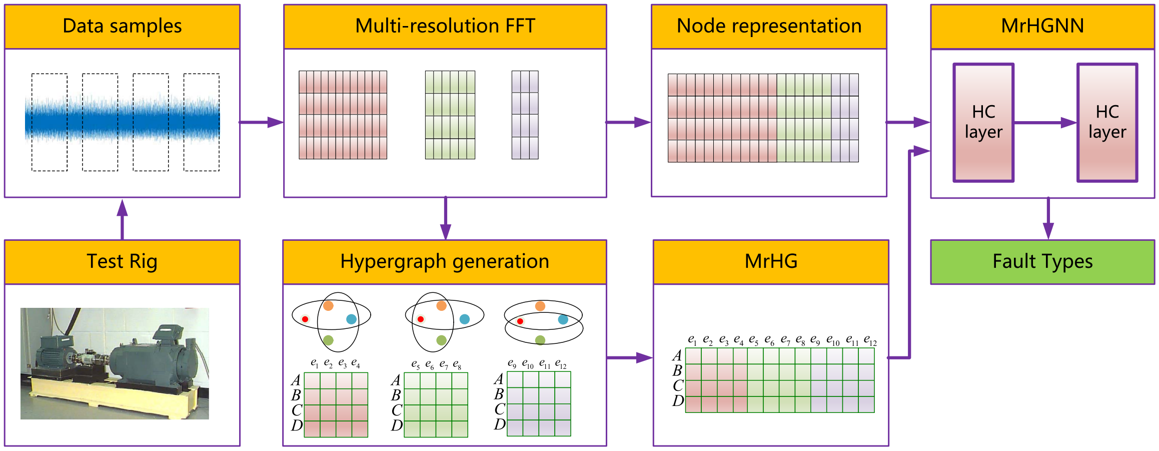


Fig. 5. Pipeline of the proposed algorithm.

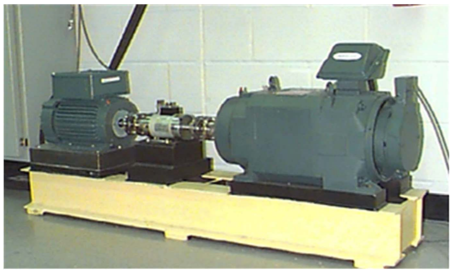


Fig. 6. Platform of the CWRU dataset [29].

TABLE I

CWRU DATASET SETTING

Fault Type	Data File	Training number	Testing number
normal	97	94	24
inner race-D0.007	105	94	24
inner race-D0.014	169	94	24
inner race-D0.021	209	94	24
ball-D0.007	118	94	24
ball-D0.014	185	94	24
ball-D0.021	222	94	24
outer race-D0.007	130	94	24
outer race-D0.014	197	94	24
outer race-D0.021	234	94	24

The Paderborn dataset [30] is obtained from a horizontal rotating machine (see Fig. 7) equipped with a variety of bearings and sensors, including artificial and natural faults under various working conditions. In our experiments, the dataset is set closer to engineering applications and the c1 current signals are selected instead of the vibration signal, making the dataset more difficult to distinguish. The fault types include normal, inner ring faults, and outer ring faults. In each fault type data, five bearing data are selected under the working condition of rotational speed 1500 rpm, load torque 0.1 N.m, and radial force 1000 N. Twenty tests are performed on each bearing, four of which are used for testing, and the other for training. The samples are cut from signals in turn and each sample has 5120 points with 1024 points overlapping the previous sample. The sampling rate is 64k. The collected dataset is marked as Paderborn-c1. The detail of the dataset is shown in Table II.

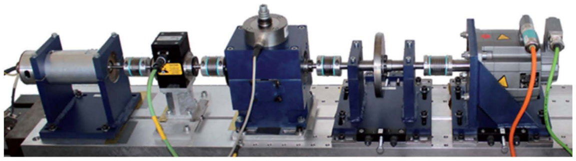


Fig. 7. Platform of the Paderborn dataset [30].

TABLE II

PADERBORN-C1 DATASET SETTING

Fault Type	Bearings for test	Training number	Testing number
normal	K001,K002,K003,K004,K005	4800	1200
inner ring fault	KI04,KI14,KI16,KI18,KI21	4800	1200
outer ring fault	KA04,KA15,KA16,KA22,KA30	4800	1200

TABLE III

AMB-Y2 DATASET SETTING

Fault Type	Training number	Testing number
normal	450	1190
rotor unbalance	450	1582
misalignment	450	838
rub-impact	450	370

The AMB-Y2 dataset is collected from a vertical machine supported with active magnetic bearings (as shown in Fig. 8). Four types of faults including normal, rotor unbalance, misalignment, and rub impact are captured from the Y2 displacement sensor. The sampling rate is 25k, and each sample contains 4096 points with 462 points overlapping. For each class, the number of samples for training is 450, and the test number is 1190, 1582, 838, and 370, respectively (see Table III).

B. Experimental Setting

Each experiment is run 10 times, with a random split of testing and training samples from the dataset. To assure the reliability of verification, the average accuracy with standard deviation is used as the final result.

Support vector machines (SVMs) with RBF kernel, AdaBoost, CNN, GCN, HGNN, and MrHGNN are introduced,

Fig. 8. Platform of the AMB-Y2 dataset [31].

TABLE IV

PERFORMANCE OF SIX ALGORITHMS ON THREE DATASETS

Algorithm/Accuracy(%)	CWRU	Paderborn-c1	AMB-Y2
SVM	99.71±0.62	75.91±5.38	95.37±0.55
AdaBoost	98.83±0.92	76.28±2.04	96.15±1.83
CNN	**99.96±0.12**	82.52±4.74	**99.26±0.19**
GCN	99.92±0.17	81.92±1.96	94.59±0.52
HGNN	**99.96±0.12**	85.32±5.73	96.20±0.31
MrHGNN	**99.96±0.12**	**89.25±4.43**	97.81±0.23

and all employ FFT features for a fair comparison. The CNN is composed of two convolution layers, two max-pooling layers, and three fully-connected layers. The latter three are graph learning algorithms and all are designed as a two-layer neural network structure with a hidden dimension of 128 and an adjacent node number of 10. The number of resolutions of MrHGNN is set to 4. The neural network-based algorithms are all trained over 1000 epochs.

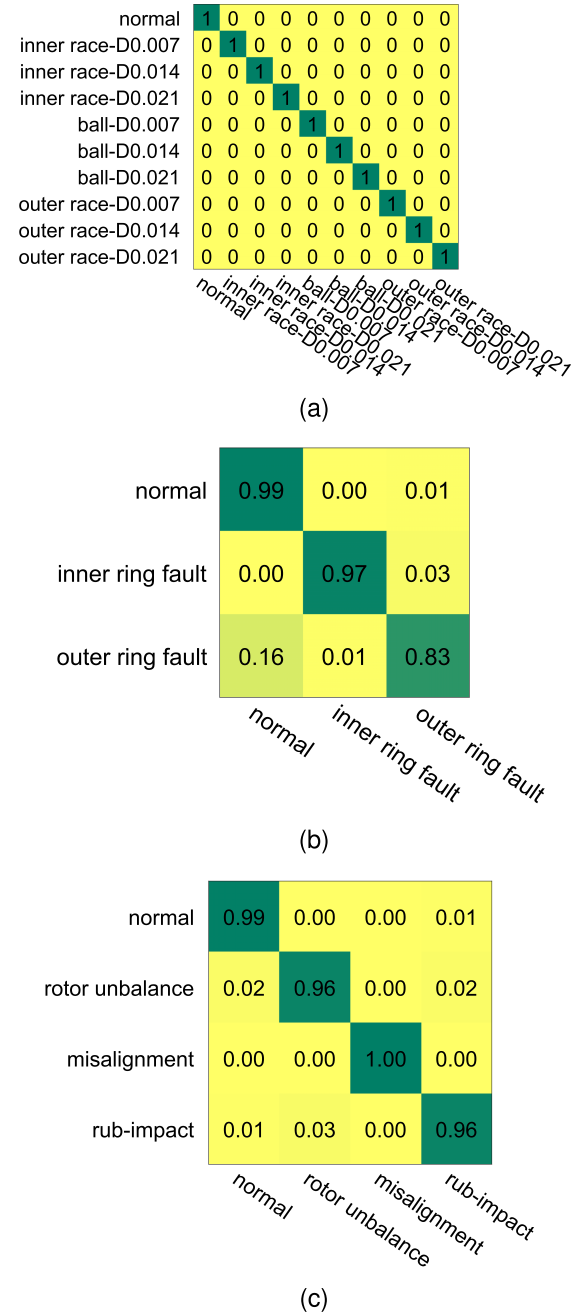
The CPU of computer is Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz, and the GPU is NVIDIA GeForce GTX 3090 Ti. The software is Pytorch 1.10.1 and THU-DeepHypergraph Toolbox [24], [27].

C. Comparison of Multiple Algorithms

Table IV shows the results of six algorithms applied to the three datasets. In general, MrHGNN outperforms the others on the three datasets.

On CWRU, all algorithms have achieved near-perfect recognition accuracy. The result of MrHGNN is comparable to that of other algorithms and close to 100%. Since CWRU is a benchmark in fault diagnosis research, it just demonstrates the feasibility of MrHGNN.

Paderborn-c1 is the most challenging of the three datasets since the current signal is used rather than the vibration signal and the training and test samples come from different tests. The graph-based algorithms beat both the traditional machine-learning algorithm SVM and deep-learning algorithm CNN because they take advantage of sample correlations in Paderborn-c1. By capturing high-order correlations between data, HGNN and MrHGNN achieve higher recognition accuracy than GCN. Furthermore, MrHGNN outperforms HGNN by about 4% since MrHG is closer to the inherent structure of the dataset than the general hypergraph.



	normal	inner race-D0.007	inner race-D0.014	inner race-D0.021	ball-D0.007	ball-D0.014	ball-D0.021	outer race-D0.007	outer race-D0.014	outer race-D0.021
normal	1	0	0	0	0	0	0	0	0	0
inner race-D0.007	0	1	0	0	0	0	0	0	0	0
inner race-D0.014	0	0	1	0	0	0	0	0	0	0
inner race-D0.021	0	0	0	1	0	0	0	0	0	0
ball-D0.007	0	0	0	0	1	0	0	0	0	0
ball-D0.014	0	0	0	0	0	1	0	0	0	0
ball-D0.021	0	0	0	0	0	0	1	0	0	0
outer race-D0.007	0	0	0	0	0	0	0	1	0	0
outer race-D0.014	0	0	0	0	0	0	0	0	1	0
outer race-D0.021	0	0	0	0	0	0	0	0	0	1

(a)

	normal	inner ring fault	outer ring fault
normal	0.99	0.00	0.01
inner ring fault	0.00	0.97	0.03
outer ring fault	0.16	0.01	0.83

(b)

	normal	rotor unbalance	misalignment	rub-impact
normal	0.99	0.00	0.00	0.01
rotor unbalance	0.02	0.96	0.00	0.02
misalignment	0.00	0.00	1.00	0.00
rub-impact	0.01	0.03	0.00	0.96

(c)

Fig. 9. Confusion matrices of MrHGNN verified on the three datasets. (a) CWRU. (b) Paderborn-c1. (c) AMB-Y2.

On AMB-Y2, GCN performs not better than SVM, indicating that the pairwise correlation has an insufficient influence on the final classification result in this dataset. Hence, compared with CNN, MrHGNN has a 1.45% lower recognition accuracy but still gets superior results in three graph learning algorithms.

The confusion matrices of MrHGNN performed on three datasets are depicted in Fig. 9.

As shown in Table V, because each hypergraph convolution layer has just one parameter matrix to learn and all operations

TABLE V

TRAINING TIME COST OF ALGORITHMS PERFORMED ON THE THREE DATASETS

Algorithm/Time(s)	CWRU	Paderborn-c1	AMB-Y2
SVM	0.11	573.14	5.42
AdaBoost	5.79	126.98	11.76
CNN	443.89	6399.07	1685.42
GCN	11.46	15.37	12.63
HGNN	17.75	41.89	31.78
MrHGNN	23.40	80.02	37.14

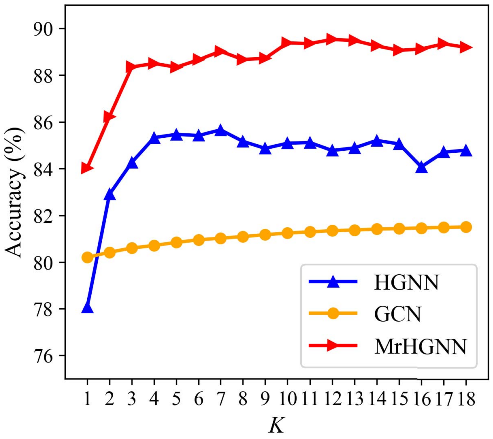


Fig. 10. Comparison of GCN, HGNN, and MrHGNN with a variation of K.

are matrix multiplications, MrHGNN is trained considerably faster than the CNN with the use of GPU. When there are more samples, MrHGNN takes less time to train than conventional machine-learning algorithms like SVM and AdaBoost.

It is worth noting that MrHGNN may achieve superior results by utilizing more discriminative node features or adding more training tricks. However, the purpose of the aforementioned series of experiments is to demonstrate that: 1) as a module or a learning mechanism, MrHGNN acts better than GCN and traditional shallow learning algorithms in many scenarios; 2) MrHG has great potential for finding the high-order correlation between samples and approaching the inherent structure of data, which helps to improve the performance; and 3) the training process of MrHGNN is much faster than that of CNN.

D. Influence of Adjacent Node Number K

The adjacent node number K is preset to generate a graph or hypergraph from unstructured data. A central node and its K neighbors build a hyperedge in a hypergraph to represent their shared properties, whereas pairwise relationships between a central node and its K neighbors are described in a simple graph. The performance of GCN, HGNN, and MrHGNN on Paderborn-c1 as K varies is illustrated in Fig. 10.

The three algorithms perform better as K increases. When compared to HGNN and MrHGNN, GCN's accuracy is less accurate and develops more slowly. The hypergraph will degenerate into a simple graph when K for HGNN and MrHGNN is equal to 1. With the increase of K, the hyperedge information becomes rich, and the common attribute information of the nodes on the same hyperedge can be expressed. It is important to note that when K exceeds a certain value, the performance of HGNN and MrHGNN hardly increases anymore or even decreases. It makes sense that if there are too many nodes located on a hyperedge, the common information of these nodes becomes reduced, weakening the information of the hyperedge feature after aggregation.

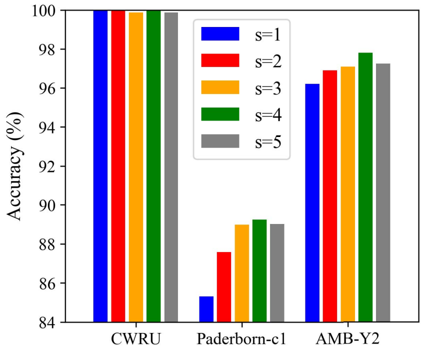


Fig. 11. Influence of the parameter s.

Compared to HGNN, MrHGNN achieves the maximum recognition accuracy at a larger K value. Low-resolution subhypergraphs can extract common correlation information at greater K because rich signal features are lost but rich common attributes remain among the low-resolution samples. As a result, even after HGNN reaches its maximal accuracy, MrHGNN's performance can still be enhanced.

E. Influence of the Resolution Number s

The resolution number s is also the parameter of MrHGNN. The larger s, the more detailed the data structure description. When $s = 1$, MrHGNN degenerates into HGNN. With the increase of s, both the dimension of the node features and the hyperedge number of MrHG gradually increase. Taking the CWRU dataset as an example, when $s = 1$, the dimension of node representation is 512, and the corresponding MrHG contains 1180 hyperedges. When $s = 5$, the node representation dimension increases to 992, and the number of hyperedges reaches up to 5900.

As depicted in Fig. 11, with the increase of s, the result shows a gradual increase in recognition accuracy for MrHGNN on the three datasets. However, as s further increases, both the node feature dimension and the number of hyperedges have become larger even exceeding the sample number, resulting in overfitting so that the accuracy is maintained at a certain level and even decreased slightly. Therefore, it is necessary to select an appropriate s when applying MrHGNN and find a balance between the algorithm performance and computational consumption. It should be noted that the choice of s should ensure that the low-resolution samples meet the requirements of the Sampling Theorem to prevent undersampling.

F. Influence of Subhypergraphs Overlapping

MrHG is made up of subhypergraphs constructed by samples at various resolutions. If the structure of subhypergraphs

TABLE VI
OVERLAP RATIO OF THE THREE DATASETS

Overlap ratio(%)	CWRU	Paderborn-c1	AMB-Y2
$\eta(\mathbf{H}_1, \mathbf{H}_2)$	5.00	0	74.01
$\eta(\mathbf{H}_1, \mathbf{H}_3)$	0.68	0	58.06
$\eta(\mathbf{H}_1, \mathbf{H}_4)$	0	0	41.45
$\eta(\mathbf{H}_2, \mathbf{H}_3)$	3.56	0	63.43
$\eta(\mathbf{H}_2, \mathbf{H}_4)$	1.10	0	44.34
$\eta(\mathbf{H}_3, \mathbf{H}_4)$	1.61	0	51.96

has a large similarity, hyperedges in subhypergraphs will partially overlap, which reduces MrHG's ability to express the inherent structure of the data.

To measure the similarity of two subhypergraphs, define $\eta(\mathbf{H}_a, \mathbf{H}_b)$ as the overlap ratio between two subhypergraphs as

$$\eta(\mathbf{H}_a, \mathbf{H}_b) = \frac{|\mathbf{H}_a \bigcap \mathbf{H}_b|}{|\mathbf{H}_a \bigcup \mathbf{H}_b|} \tag{16}$$

where $|\mathbf{H}_a \bigcap \mathbf{H}_b|$ denotes the number of the same hyperedges in $\mathbf{H}_a$ and $\mathbf{H}_b$, and $|\mathbf{H}_a \bigcup \mathbf{H}_b|$ represents the union of hyperedges in $\mathbf{H}_a$ and $\mathbf{H}_b$.

From Table VI, it can be seen that all the overlap ratio in Paderborn-c1 is 0, indicating that different subhypergraphs have distinct structures for describing the information of the data. Hence, the improvement of MrHGNN relative to HGNN is greater than that of other datasets. While the overlap ratio of AMB-Y2 and CWRU is larger, even reaching 0.74, which means that MrHG contains a large number of repeated hyperedges, reducing the effectiveness of MrHGNN.

G. Effect of Noise on the Performance of MrHGNN

To demonstrate that MrHGNN is robust, we manually introduce random noises to the data and then investigate if the model's performance is significantly affected. The amount of additional noise is gauged using the signal-to-noise ratio (SNR). The noise decreases as the SNR increases. To evaluate the algorithm's anti-interference capability, the parameter accuracy retain ratio (ARR) is defined as

$$\text{ARR} = \frac{\text{accuracy}_{\text{noise}}}{\text{accuracy}_{\text{noiseless}}} \tag{17}$$

where $\text{accuracy}_{\text{noise}}$ and $\text{accuracy}_{\text{noiseless}}$ stand for the algorithm's accuracy rate with or without noise, respectively. The value of the ARR shows that, in contrast to the case where noise is not manually added, the algorithm can maintain its recognition performance ratio in the presence of noise. The algorithm's antinoise ability increases with the ARR value. Fig. 12 displays the performance of MrHGNN on three datasets under various levels of noises.

The ARR of MrHGNN is above 98% when the SNR is higher than 30 dB, keeping a perfect antinoise capacity. The algorithm's performance on the CWRU dataset declines significantly when the SNR is less than 30 dB. Owing to the fact that the CWRU dataset contains fewer samples of each type, it causes the intraclass dispersion of the samples to worsen when noise is severe. In contrast, in the Paderborn-c1 and AMB-Y2 datasets, MrHGNN can still maintain the ARR greater than 80% even when the SNR = 10 dB, fully demonstrating the potent anti-interference ability.

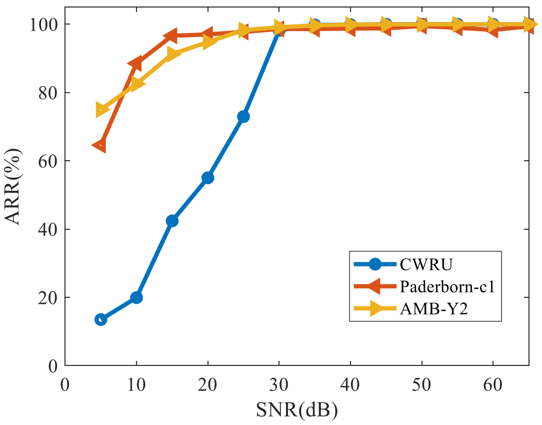


Fig. 12. Performance of MrHGNN on three datasets under various levels of noises.

H. Conclusion

From the above series of experiments, the following conclusions can be drawn.

1) Compared to the simple graph and hypergraph, MrHG approximates the potential data structure by combining multiple subhypergraphs under various resolutions. Therefore, MrHGNN can handle unstructured data with strong discriminative power.
2) MrHG is constructed based on the KNN principle. The number of nodes in each hyperedge is determined by the K value. The hyperedge information grows richer as K increases, and the common attributes can be extracted from more nodes. However, when K reaches a certain value, it leads to a reduction of the efficient information propagated from nodes to the hyperedge.
3) MrHG consists of subhypergraphs with multiple resolutions. The description ability of MrHG will be improved by increasing the resolution number s. However, s is too large to cause time cost and the occurrence of overfitting.
4) The multiple subhypergraphs in MrHG describe the structure of the data from coarse to fine. When there are fewer overlapping hyperedges between subhypergraphs, MrHGNN outperforms HGNN by a substantial margin. On the other hand, the more overlapping the subhypergraphs act, the less the improvement of MrHGNN.
5) MrHGNN exhibits robustness to noises and can maintain comparable performance even in the presence of severe noises.

V. DISCUSSION

This article proposes a new fault diagnosis algorithm MrHGNN. MrHG is generated to incorporate the relationship between more than two nodes and the potential information hidden in the unstructured data is mined from coarse to fine. Fault type is recognized according to MrHG learning in a deep neural network way and a series of experiments demonstrate the superior performance of our proposed.

To improve the effectiveness of fault diagnosis in the future, we will continue to explore constructing a hypergraph beyond the KNN mechanism and adaptively learn the correlation between the samples to approach the inherent structure of data.

REFERENCES

[1] Y. Zhang, T. Zhou, X. Huang, L. Cao, and Q. Zhou, “Fault diagnosis of rotating machinery based on recurrent neural networks,” *Measurement*, vol. 171, Feb. 2021, Art. no. 108774.

[2] R. Liu, F. Wang, B. Yang, and S. J. Qin, “Multiscale kernel based residual convolutional neural network for motor fault diagnosis under nonstationary conditions,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 3797–3806, Jun. 2020.

[3] X. Yan, C.-A. Zhang, and Y. Liu, “Multi-branch convolutional neural network with generalized shaft orbit for fault diagnosis of active magnetic bearing-rotor system,” *Measurement*, vol. 171, Feb. 2021, Art. no. 108778.

[4] J. Wang, P. Fu, L. Zhang, R. X. Gao, and R. Zhao, “Multilevel information fusion for induction motor fault diagnosis,” *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 5, pp. 2139–2150, Oct. 2019.

[5] H. Wang, S. Li, L. Song, L. Cui, and P. Wang, “An enhanced intelligent diagnosis method based on multi-sensor image fusion via improved deep learning network,” *IEEE Trans. Instrum. Meas.*, vol. 69, no. 6, pp. 2648–2657, Jun. 2020.

[6] R. Yan, F. Shen, C. Sun, and X. Chen, “Knowledge transfer for rotary machine fault diagnosis,” *IEEE Sensors J.*, vol. 20, no. 15, pp. 8374–8393, Aug. 2020.

[7] W. Li et al., “A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges,” *Mech. Syst. Signal Process.*, vol. 167, Mar. 2022, Art. no. 108487.

[8] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1–9.

[9] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[10] W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[11] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–12.

[12] T. Li, Z. Zhou, S. Li, C. Sun, R. Yan, and X. Chen, “The emerging graph neural networks for intelligent fault diagnostics and prognostics: A guideline and a benchmark study,” *Mech. Syst. Signal Process.*, vol. 168, Apr. 2022, Art. no. 108653.

[13] Y. Gao, M. Chen, and D. Yu, “Semi-supervised graph convolutional network and its application in intelligent fault diagnosis of rotating machinery,” *Measurement*, vol. 186, Dec. 2021, Art. no. 110084.

[14] T. Li, Z. Zhao, C. Sun, R. Yan, and X. Chen, “Multireceptive field graph convolutional networks for machine fault diagnosis,” *IEEE Trans. Ind. Electron.*, vol. 68, no. 12, pp. 12739–12749, Dec. 2021.

[15] C. Yang, J. Liu, K. Zhou, X. Jiang, and X. Zeng, “An improved multi-channel graph convolutional network and its applications for rotating machinery diagnosis,” *Measurement*, vol. 190, Feb. 2022, Art. no. 110720.

[16] K. Zhou, C. Yang, J. Liu, and Q. Xu, “Dynamic graph-based feature learning with few edges considering noisy samples for rotating machinery fault diagnosis,” *IEEE Trans. Ind. Electron.*, vol. 69, no. 10, pp. 10595–10604, Oct. 2022.

[17] C. Li, L. Mo, and R. Yan, “Fault diagnosis of rolling bearing based on WHVG and GCN,” *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.

[18] C. Yang, K. Zhou, and J. Liu, “SuperGraph: Spatial–temporal graph-based feature extraction for rotating machinery diagnosis,” *IEEE Trans. Ind. Electron.*, vol. 69, no. 4, pp. 4167–4176, Apr. 2022.

[19] D. Zhang, E. Stewart, M. Entezami, C. Roberts, and D. Yu, “Intelligent acoustic-based fault diagnosis of roller bearings using a deep graph convolutional network,” *Measurement*, vol. 156, May 2020, Art. no. 107585.

[20] Y. Gao, Z. Zhang, H. Lin, X. Zhao, S. Du, and C. Zou, “Hypergraph learning: Methods and practices,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 5, pp. 2548–2566, May 2022.

[21] L. Su, Y. Gao, X. Zhao, H. Wan, M. Gu, and J. Sun, “Vertex-weighted hypergraph learning for multi-view object classification,” in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2779–2785.

[22] X. Hao, J. Li, Y. Guo, T. Jiang, and M. Yu, “Hypergraph neural network for skeleton-based action recognition,” *IEEE Trans. Image Process.*, vol. 30, pp. 2263–2275, 2021.

[23] S. Bai, F. Zhang, and P. H. S. Torr, “Hypergraph convolution and hypergraph attention,” *Pattern Recognit.*, vol. 110, Feb. 2021, Art. no. 107637.

[24] Y. Gao, M. Wang, D. Tao, R. Ji, and Q. Dai, “3-D object retrieval and recognition with hypergraph analysis,” *IEEE Trans. Image Process.*, vol. 21, no. 9, pp. 4290–4303, Sep. 2012.

[25] D. Di, J. Zhang, F. Lei, Q. Tian, and Y. Gao, “Big-hypergraph factorization neural network for survival prediction from whole slide image,” *IEEE Trans. Image Process.*, vol. 31, pp. 1149–1160, 2022.

[26] D. Zhou, J. Huang, and B. Schölkopf, “Learning with hypergraphs: Clustering, classification, and embedding,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1–8.

[27] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, “Hypergraph neural networks,” in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 1–8.

[28] J. Jiao et al., “A comprehensive review on convolutional neural network in machine fault diagnosis,” *Neurocomputing*, vol. 417, pp. 36–63, Dec. 2020.

[29] [Online]. Available: https://csegroups.case.edu/bearingdatacenter/pages/welcome-case-western-reserve-university-bearing-data-center-website

[30] C. Lessmeier, J. Kimotho, D. Zimmer, and W. Sextro, “Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification,” in *Proc. PHM Soc. Eur. Conf.*, 2016, pp. 1–17.

[31] X. Yan, Z. Sun, J. Zhao, Z. Shi, and C.-A. Zhang, “Fault diagnosis of rotating machinery equipped with multiple sensors using space-time fragments,” *J. Sound Vib.*, vol. 456, pp. 49–64, Sep. 2019.

Xunshi Yan received the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2013.

He is currently an Associate Research Professor with the Institute of Nuclear and New Energy Technology, Tsinghua University. His research interests include machine learning, fault diagnosis, and active magnetic bearing.

Yang Liu received the B.S. degree in aircraft design and engineering from Northwestern Polytechnical University, Xi’an, China. He is currently pursuing the Ph.D. degree in general mechanics and fundamentals of mechanics with the School of Engineering Science, University of Chinese Academy of Sciences, Beijing, China, and the Institute of Mechanics, Chinese Academy of Sciences, Beijing.

His primary research interest includes flush air data sensing systems and physics-informed neural networks.

Chen-An Zhang received the Ph.D. degree in fluid–solid interaction and control from Northwestern Polytechnical University, Xi’an, China, in 2010.

He is currently a Professor with the Institute of Mechanics, Chinese Academy of Sciences, Beijing, China. His research interests include aerodynamics, aeroelasticity, and fault diagnosis.