South China University of Technology

《统计学习与数据科学导论课程设计》 课程 论 文

(2023 -2024 学年第 2 学期)

论文题目

学生姓名：徐维石

提交日期： 2024 年 8 月 23 日 学生签名： 徐维石

学　　号	202130321646	座位编号	
学　　院	数学学院	专业班级	21 信息管理与信息系统
课程名称	统计学习与数据科学导论课程设计	任课教师	夏立
教师评语：			

华南理工大学

South China University of Technology

本论文成绩评定： _____分

A Multi-Scale Convolutional Neural Network with Ample Interaction Capability for Mechanical Fault Diagnosis

Weishi Xu

School of Mathematics, South China University of Technology, Guangzhou510630

xu_2906541850@163.com

Abstract

In this work, we focus on the fault diagnosis task and build a deep learning model with a new structure. This model takes advantage of the 1-d convolutional neural network model and makes improvements by adding the multi-scale feature extraction of the data, enabling the model to learn multi-scale features and improve the model's accuracy and robustness, as well as the generalization capability. The experiments are performed on the CWRU dataset with 3 different models compared. The proposed new structure in this paper achieved an accuracy of 99.42% which is higher than the other two methods, and the training cost is relatively small. The proposed method can be utilized to build an end-to-end fault diagnosis system in different fault diagnosis tasks and achieve reliable performance.

Keywords: Fault diagnosis, Convolutional Neural Network, Deep Learning, Multi-scale

1. Introduction

Fault diagnosis plays a crucial role in the realm of manufacturing systems, as it enables the timely detection of emerging issues, ultimately yielding significantly decreased time and cost overhead [1][2]. With the advent of smart manufacturing technologies, data-driven fault diagnosis has gained immense research attention [3][4]. By leveraging these data-driven approaches, manufacturers can identify and diagnose potential faults or abnormalities in the production process, machinery, or product quality. These developed approaches allow for the prompt initiation of corrective actions, minimizing downtime, optimizing productivity for practical equipment, and ensuring the delivery of high-quality products to customers.

For the data-driven approaches, traditional machine learning techniques are utilized to solve

this problem [5]. For example, the support vector machine (SVM) has been applied in many domains and achieved great success because of its superior performance and generalization ability [6][7]. It has been implemented to solve the fault diagnosis problems in previous research [8][9]. Besides, other typical machine learning methods, i.e., tree-based methods like random forest, have been broadly applied to solve the fault diagnosis problem [10]. They attracted much attention because they provide a powerful and flexible algorithm that shows great performance [11]. In recent years, it is more prominent to use deep learning methods to perform fault diagnosis. Generally, the deep learning methods show higher accuracy and robustness. Besides, it also enables us to train the model in an end-to-end learning manner, without a sophisticated feature extraction process [13][14]. Among them, the most representative method, i.e., autoencoder which is widely applied in signal processing [15], provides a powerful model for fault diagnosis[16][17]. Besides, the convolutional neural network (CNN), which has impressive ability in feature extraction and prediction in the computer vision domain [18], is also built and modified to perform fault diagnosis and achieve outstanding accuracy [19][20]. At the same time, methods that combined different neural structures are also been established [21][22]. These approaches are used to automatically diagnose faults in mechanical systems and therefore enhance operational efficiency, improve product reliability, and maintain a competitive edge in the dynamic landscape of modern manufacturing [12].

However, these methods ignore the capture of multi-scale characteristics of vibration signals. To solve this problem, we propose the multi-scale CNN model with ample interaction ability (MSCNN-AI) in this paper. In our method, the vibration signals are input through a shared CNN layer to obtain the convolution features. Then, the convolution features are decomposed into multi-scaled representations. The interaction between multi-scale features in different layers avoids insufficient interaction problems with stitching only in the last layer and finally improves the robustness and generalization ability of the diagnosis model. Our model is validated in the CWRU dataset, and the experimental results illustrate that the proposed method achieves increased accuracy and stability compared to the model without sufficient interactions.

The structure of the rest of our paper is as follows. In Section 2, we introduced the preliminary part, in which we briefly introduced CNN structure. In Section 3, we introduced the methodology section and explained the method proposed in our paper in detail. In Section 4, we conducted and introduced the validation process on the CWRU dataset. Finally, in Section 5, we made a conclusion to the paper.

2. Preliminary

Convolutional neural network is a type of deep learning algorithm, a special type of multilayer perceptron neural network, which is specifically designed for processing and analyzing visual data and information, such as images and videos. Its outstanding performance in extracting local features from visual data allows it to capture visual feature just like mammals do. The model was originally proposed by LeCun et al. [23] in a classification task for handwritten digits and is now widely utilized as a strong machine learning algorithm to solve many problems and have wide applications in different domains.

2.1 Convolutional layer

Convolutional layers contains a group of filters(convolution kernels) that slide over the input image, automatically learning and extracting features from input images. Convolutional layers capture local patterns and spatial hierarchies in the data, which is illustrated in Fig. 1. It has two attractive advantages: 1) Sparse connection. Each kernel is connected with a local patch of previous features which effectively reduces the amount of parameters and accelerates training. 2) Weight sharing. Feature maps share the same weight in the same kernels, which further reduces the amount of parameters. Mathematically, let $x \in R^{m*n}$ be the input image with a size of m*n, the *j*-th output feature map c^j can be described as:

$$c^j = x * w^j + b^j \quad (1)$$

in which w^j represents the *j*-th filter weights and b^j denotes the bias of the filter, $*$ denotes the convolutional operation.

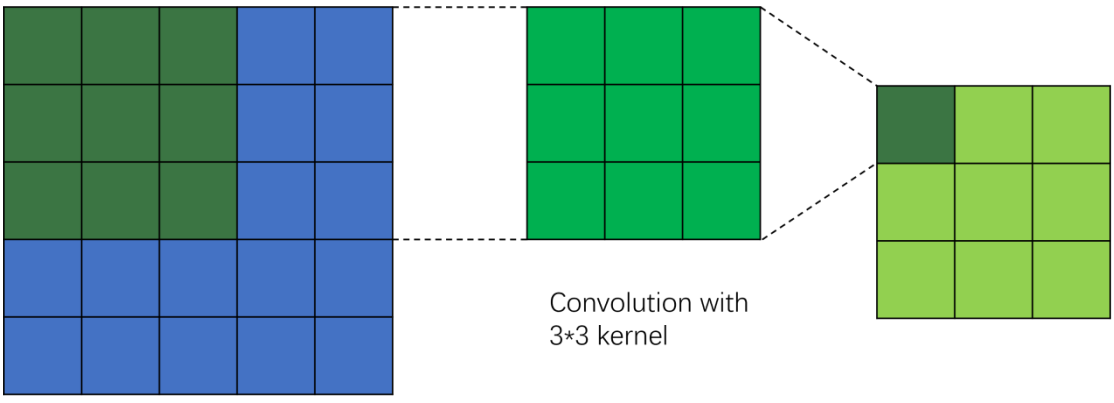

Fig 1. An illustration of the convolution operation.

2.2 Pooling layer

Pooling layers are built to reduce the dimensions and scale of the feature maps while retaining most of the important information of the data. It can effectively make the model

smaller and reduce the amount of parameters. Max pooling is the most commonly used pooling technique which works by extracting the maximum value from a small window of the feature map. Average pooling, which extracts the average value, is also widely used in applications. The mathematical description is as follows:

$$p_k^j = max\{c_{s:s+r-1;t:t+r-1}^j\} \quad (2)$$

$$p_k^j = avg\{c_{s:s+r-1;t:t+r-1}^j\} \quad (3)$$

where c^j represents the input, r is the pooling size, p_k^j represents the output. s and t denote the current position of the input.

2.3 Dropout [29]

Dropout is a technique widely used in convolutional neural networks to prevent overfitting and improve the generalization ability of the model. It works by randomly deactivating a certain proportion of neurons during each training iteration. The purpose of doing this is to prevent the model from relying too much on the data which probably leads to overfitting. It also provides a way to combine different models approximately, eventually improving the model's preformance.

2.4 Fully-connected layer

The fully-connected layer often comes after convolutional layers. It connects every neuron from the previous layer to every neuron in the next layer and processes features further. It can be described mathematically as:

$$fc^l = \sigma(w^l * fc^{l-1} + b^l) \quad (3)$$

where fc^l represents the output features of the l-th fully-connected layer, w^l and b^l denote the weights and bias, respectively.

2.5 Softmax

The softmax function is often used in multi-class classification problems and is typically applied to the output layer of a neural network. It transforms the raw scores (also known as logits) into values that represent a probability distribution, ensuring that the probability values for each class are between 0 and 1, and the sum of all class probabilities is equal to 1. For given raw scores $s = [s_1, s_2 \ldots, s_n]$, the mathematical description of the softmax function is:

$$p_j = \frac{e^{s_j}}{\sum_{j=1}^{n} e^{s_j}} \quad (4)$$

In this equation, p_j is the j-th output probability. " e " represents the base of the natural logarithm. "n" represents the number of classes, and "s_j" represents the raw score of the jth class.

2.6 Cross-Entropy

Cross-entropy loss is a commonly use loss function for classification tasks. It calculates the dissimilarity between the output of the softmax layer, which represents the predicted class possibility, and the true one-hot encoded class labels. The mathematical expression of this function is:

$$L(x, y) = -\sum_{i=1}^{C} x_i \log p_i \qquad (5)$$

In this equation, "x_i" presents the ith element of the true label, and "p_i" represents the probability predicted by the model that x belongs to the ith class.

3. Methodology

3.1. Problem Formulation

The task of fault diagnosis is essentially a pattern classification task. For given input $y \in R^n$, We need to build a neural network-formulated function $z = f(y)$, to present the estimation of the health condition z， where$z \in R^c$, and c denotes the number of classes.

3.2. Proposed method

In this section, we present the proposed Multi-Scale CNN with Ample Interaction (MSCNNAI) method for the fault diagnosis problem. The basic architecture of MSCNNAI is presented in Fig . 2:

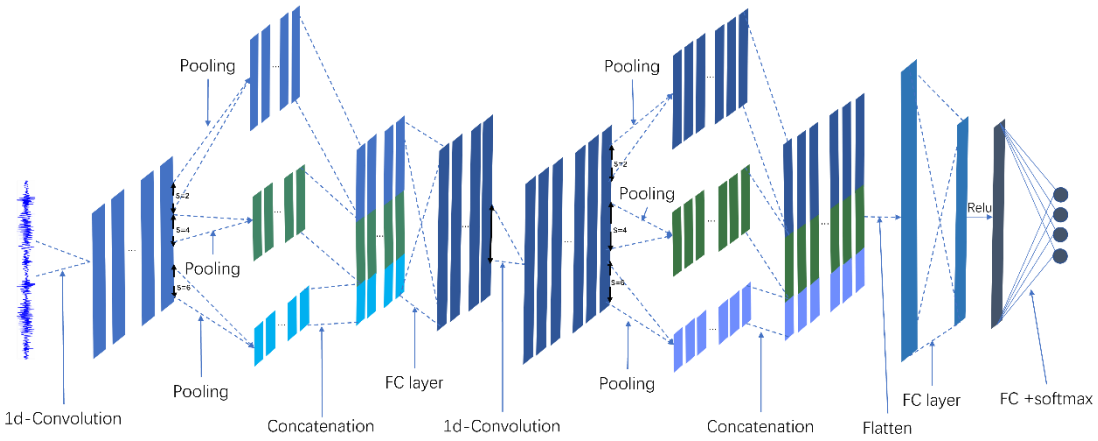

Fig. 2. The illustration of the proposed MSCNNAI. After each 1-d convolutional operation, different scales of pooling operations are performed to generate feature representations with different scales, thus enhancing the multi-scale capture ability of the model.

The key idea of the MSCNNAI model is to extract the multiscale feature from raw data, incorporate the multiscale feature learning capability, and improve the model's interaction capability in the hidden convolution layers. We extract the multiscale feature by using different pooling sizes after the 1-d convolution layer and concatenate them before the next layer to prevent them from training independently without interaction. With this approach, we not only increase the robustness and stability of this model by extracting the multiscale feature from the data but also enable the model with higher accuracy and smaller size by the interaction operation.

Now we introduce the different parts of this model in detail.

1) 1-d Convolution

We use a 1-D convolution layer to extract and learn high-level features from the raw data. We use 2 different convolution layers (C1 and C2), each of which has 4 components: **convolution, batch normalization, activation function (relu), and dropout**. For the input signal $\{y^{(s)}\}$ with a length of s and the kernel size of m, The output z_i after the convolution layer can be described as

$$z_i = \sigma(w^T * y_{i:i+m-1} + b) \quad (6)$$

where w represents the filter vector (kernel), b represents the bias, and σ is the activation function. In this study, we use rectified linear units (ReLU) [27] as the activation function to prevent the vanishing gradient problem and accelerate the convergence of the model:

$$\sigma(x) = \max(0, x) \quad (7)$$

.

After going through the activation function, the feature map can be described as $z = [z_1, z_2 \dots, z_{s-m+1}]$. The length of z can be calculated as $s - m + 1$, where s denotes the input length, and m represents the kernel size.

Afterward, the data will go through the **dropout layer** where the dropout rate equals 0.6. This means that 60% of the neurons will be randomly deactivated during each training iteration.

For the first convolution layer, the input channel number is 1 and the output channel number is 64. For the second convolution layer, the input channel number is 64 and the

output channel number is 128. We increase the channel number to increase the feature extraction capability. For the second convolution layer, we change the kernel size from 9 to 5.

2) Multiscale pooling

After the 1-d convolution layer, the feature map is transferred into 3 different-scale features by going through 3 different pooling sizes, in which the pooling sizes are 2, 4, and 6. We use Average pooling operation in this study. It works by calculating the mean value of the adjacent *r* feature and combining them into one value.

$$p_k^j = avg\{c_{k:k+r-1}^j\} \quad (8)$$

This operation extracts multiscale features after each pooling layer. Totally we use the multiscale pooling layer 2 times.

3) Multiscale interaction

After extracting the multiscale feature by the multiscale pooling operation, the model **concatenates** the features with different lengths into one feature map. The purpose of doing this is to improve the interaction capability of this model. By combining them into one feature map, we can perform comprehensive and interactive feature learning to extract high level features from different scales. This not only enhances the robustness of the model but also makes the model smaller, as the model only has one channel. If we train the different scale branches individually, there will be 3 branches with 6 different 1-d convolution layers, resulting in a much larger model with massive parameters.

Besides concatenation, the model incorporates an FC layer after the concatenation process to perform the interaction of the feature with different scales. The output of the FC layer is fed into the next convolution layer.

Mathematically, given q_k^j (k =1,2; j=1,2,3) as the j th output of the k th multiscale-pooling layer, the output of the kth interaction process can be described as

$$c^k = W^k * (q_k^1 \oplus q_k^2 \oplus q_k^3) + b^k \quad (9)$$

where $\oplus$ represents the concatenation operation, W^k denotes the weights of the kth FC layer, b^k denotes the bias of the kth layer.

4) Linear+softmax

The fault diagnosis problem is a multiclass classification problem. The output of the

2 convolution layer is fed into 2 additional fully connected layers. The first is a fully connected layer with Relu activation function, and the second is an FC layer with a Softmax activation function that outputs a conditional probability for each class.

For feature map $z = [s_1, s_2 \ldots, s_n]$, the mathematical description of the softmax function is:

$$p_j = \frac{e^{s_j}}{\sum_{j=1}^{n} e^{s_j}} \tag{10}$$

In this equation, p_j is the output probability of jth class. " e " represents the base of the natural logarithm. "n" represents the number of classes, "s_j" represents the raw score of the j-th class.

3.3 Model Training and Serving

3.3.1 Training

MSCNNAI is trained using the backpropagation (BP) algorithm [28] with gradient descent, similar to traditional CNNs. It inputs raw temporal vibration signals and predicts gearbox conditions, with different class labels representing various health conditions. During MSCNNAI training, the loss function is the cross-entropy between the predicted class labels and the true class labels. To efficiently compute and minimize the loss function, we utilize the Adam optimization algorithm[24], which requires minimal memory usage.

3.3.2 Serving:

For every online sample, we input the samples to the well-trained fault diagnosis system to automatically perform the calculation of multiscale features and directly diagnose the fault type of the machine gear.

4. Experiments

In this section, we evaluated the proposed MSCNNAI method by implementing experiments on the **CWRU-bearing dataset [26]**. The introduction of the dataset and the data pre-processing method will be first presented, and then we will show the result and discussion in detail.

4.1 Dataset introduction

The CWRU bearing dataset is widely used as a standard to evaluate the classification

algorithms for bearing fault diagnosis. It contains vibration signal data of different types of bearings. There are 4 classes of bearing signals in the dataset: 3 types of fault bearings and normal bearings:

- Normal bearings
- Inner raceway fault (IR)
- Outer raceway fault (OR)
- Ball fault(B).

The data is stored as Matlab files, it was recorded by accelerometers installed at the drive end (DE) and fan end (FE. The dataset uses a sampling rate of 12 kHz and each Matlab file contains sample points between 120k to 240k.

4.2. Data pre-processing

In this experiment, we utilize a sliding window with a fixed length of **500** to generate samples. Each sample is a vector with length of 500, and the step length equals to the sample length ensuring no overlapping between the samples. And we totally create **1.5k** samples to form the dataset we need.

In order to train and validate the model, we split the dataset into a training set and a validation set with a ratio of **0.5**. It means that half of the dataset is a validated set.

Because the deep learning approach enables us to do end-to-end learning, we skip other pre-processing operations and directly feed the data into the model and the model can do the feature extraction process automatically.

4.3. Methods

To evaluate the effectiveness of the proposed MSCNNAI method, we set 3 different CNN model with different structures and compared their performance. In order to make a comparison, we incorporate a 3-branch MSCNN model which is proposed in [25]. The 3 different models are:

- Simple 2-layer CNN
- 3-branch MSCNN
- MSCNNAI

All the models have the same depth of 2. We use Relu as the activation function of the hidden layers and Softmax as the activation function of the final FC layer. The 2-layer CNN model contains 2 convolution layers and 1 FC layer. After going through the convolution kernel, the data will go through the batch normalization process, the activation function, dropout, and the pooling layer before the next convolution layer. The second convolution layer contains a

similar process. For the first convolution layer, there is 1 input channel and 64 output channels. The kernel size equals 9 and the stride is 1, the padding equals 4. And for the second convolution layer, there are 64 input channels and 128 output channels, the kernel size, stride length, and padding are 5, 1, and 2, respectively. For the first layer, we use maxpooling layer where the pooling length and stride equal to 2. For the second layer, we use avgpooling layer where the pooling length and stride equal to 2. We set the dropout rate equal to 0.6 to ensure the generalization capability of the model.

The 3-branch MSCNN model totally contains 6 convolution layers and 2 FC layers. The model has 3 branches, each branch has 2 convolution layers, and the structure and settings of the hyper-parameters of the 2 convolution layers are the same as the 2-layer CNN model. The 3 branches have exactly the same settings on the 2 convolution layers. After the data is sent into the model, the data will go through 3 Avgpooling layers at the same time where the pooling length is 2, 4, 6 and the stride is 2, 4, 6 respectively. This process will generate 3 branches, each branch is a feature map with different scales. The 3 branches will go through the 2 convolution layers independently and concatenate after the 2 layers. The concatenated feature will go through 2 FC layers and present an output for which length is 4. The dropout rate is also 0.6 to avoid overfitting. The setting of the MSCNNAI is mentioned in the previous section. In this experiment, we train the model for 40 epochs.

4.4. Performance Comparison

In order to evaluate the performance of the 3 models, we record and present the **accuracy** and **loss** of the 3 models. In order to compare the training cost, we also report the training time. Besides, we use the F1 score as another important evaluation tool.

F1 score is a function used to evaluate the performance of binary or multiclass models. It is defined by calculating the harmonic mean of precision and recall. Precision represents the proportion of correctly predicted positive samples out of all samples predicted as positive by the model and can be calculated as

$$Precision = \frac{TP}{TP+FP} \tag{11}$$

where TP is the number of true positives and FP is the number of false positives.

Recall represents the proportion of correctly predicted positive samples out of all true positive samples and can be calculated as

$$Recall = \frac{TP}{TP+FN} \tag{12}$$

where FN is the number of false negatives.

Thus, the F1 score can be calculated as :

$$F1 = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{13}$$

In this study, we report the mean of the F1 score of every class as the representation of the performance. The performance is reported as follows:

Table1. Comparison of 3 models.

	2L-CNN	3-Branch MSCNN	MSCNNAI
Accuracy	0.9853	0.9888	0.9945
loss	0.03	0.0192	0.0298

Table 1 reported the accuracy and loss of the model's outcome, and we visualized the F1 scores and training time of the 3 models in Figs. 4~5.

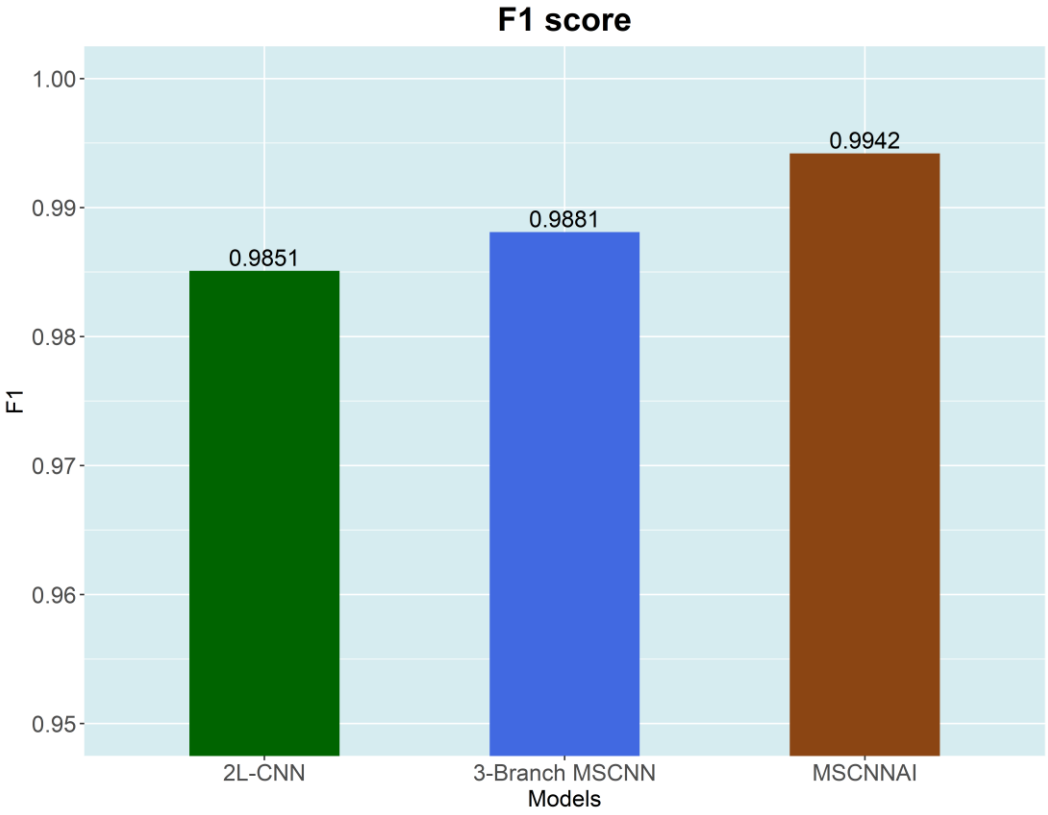

Fig. 4. The comparison of 3 model's F1 score.

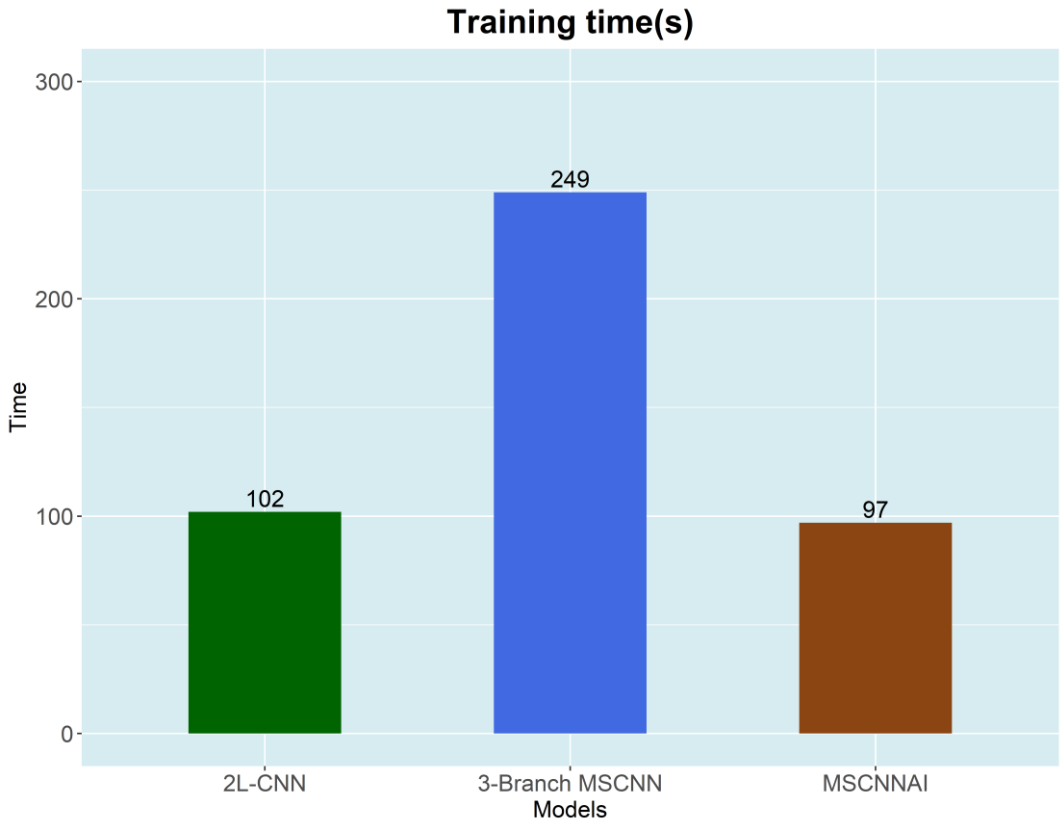

Fig. 5. The comparison of 3 model's training time (seconds).

All the reported accuracy and F1 scores are from the validation set. As we can see, both 3 models have achieved an accuracy higher than 98%. It means all the models fit the data well and have strong generalization ability. However, the improvements of incorporating the multiscale feature in the CNN model are quite obvious: both the accuracy and F1 score raised compared to the 2-L CNN model. The proposed MSCNNAI model has outstanding performance on the dataset which is significantly higher than the other 2 models and higher than 99%. The improvement of the training cost of the MSCNNAI model is also obvious as compared with the 3-branch MSCNN model. It took only 97s to finish all the training, which is less than half of the time the 3-branch MSCNN spent, and achieved higher accuracy than the 3-branch MSCNN. So we can conclude that the proposed MSCNNAI has significant improvement in both the accuracy and the efficiency in the fault diagnosis task.

5. Further analysis

In this section, we will report the loss and accuracy curve of the 3 models and make further analysis.

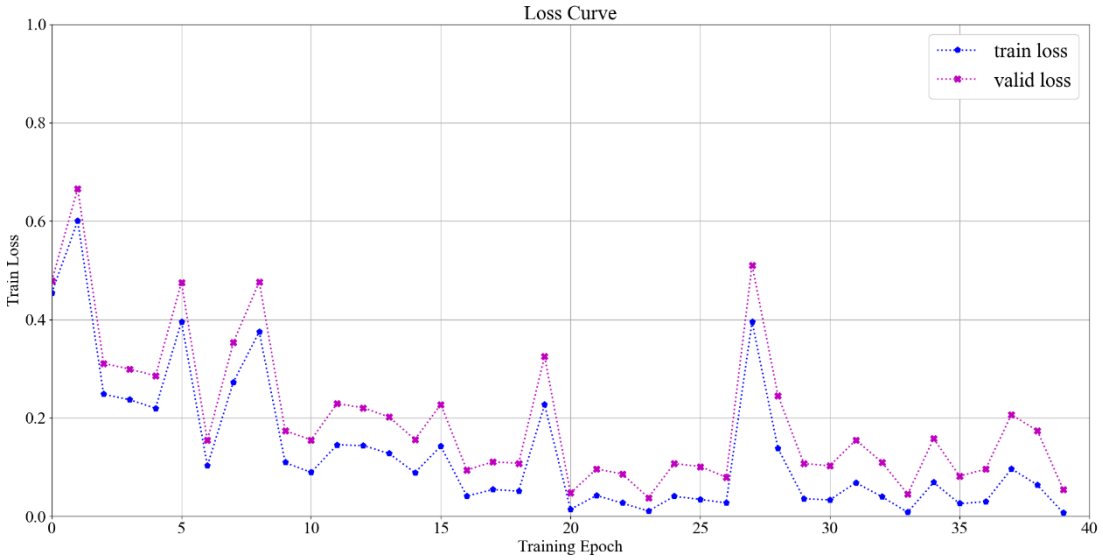

Fig6. The loss curve of the 2-layer CNN model.

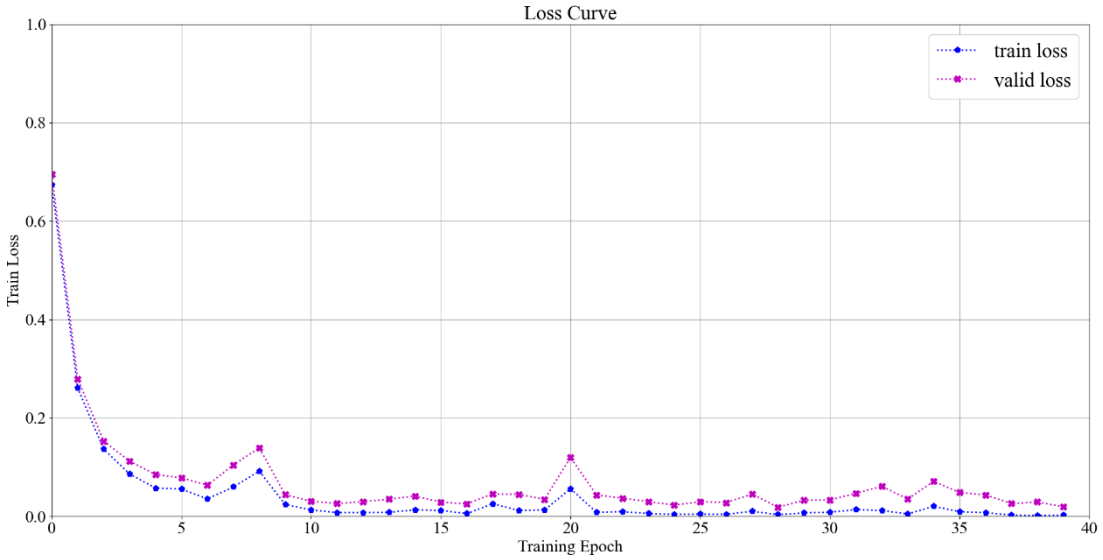

Fig7. The loss curve of the MSCNNAI model.

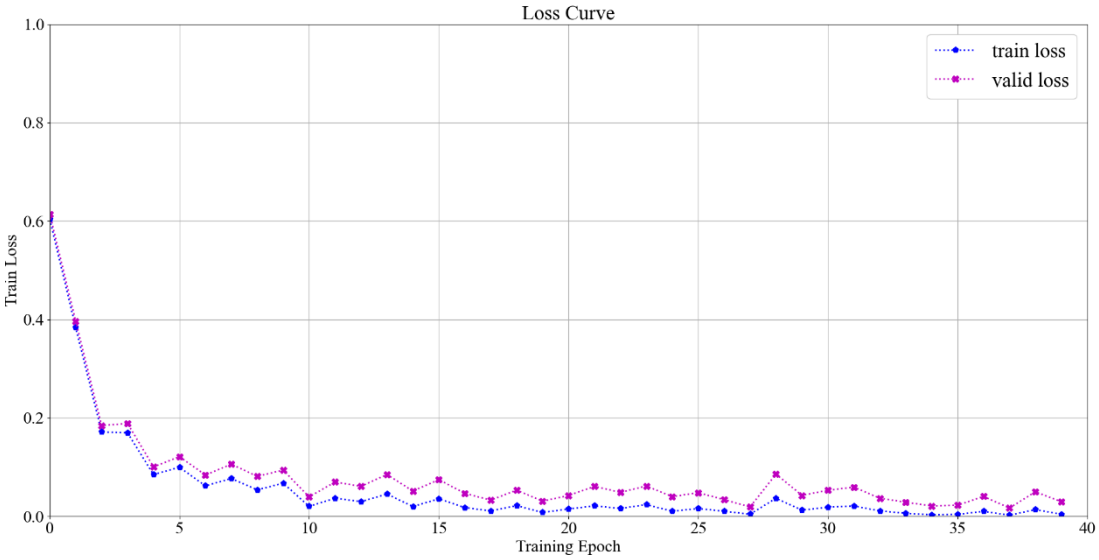

Fig8. The loss curve of the 3-branch MSCNN model.

The loss curve can imply the robustness and convergence rate of the 3 models. As the outcome shows, the 2-L CNN model is unstable compared to the other 2 models. It has significant volatility after 15 epochs while the other 2 models have achieved steadiness after 15 epochs. It means that both the 3-branch MSCNN and MSCNNAI models have converged after 15 epochs and have strong robustness. Both the 2 models have rather small fluctuations after 10 epochs, so the convergence rate of the 2 models are fast.

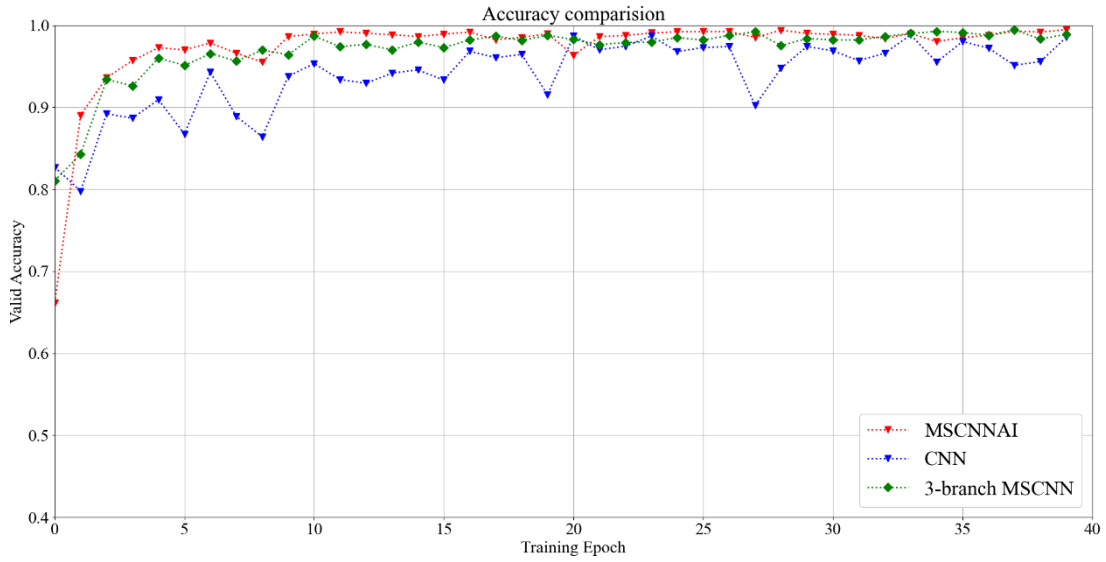

Fig9. The accuracy comparison of the 3 models.

The Fig 9 shows the accuracy curve of the 3 models. As we can see the convergence rate of MSCNNAI and 3-branch MSCNN is quite similar. We can conclude that the robustness of the MSCNNAI and 3-branch MSCNN is a lot better than the simple 2-layer CNN. The robustness of the MSCNNAI and 3-branch MSCNN have no significant difference, and the accuracy of the MSCNNAI is slightly better than the 3-branch MSCNN and reaches high accuracy faster.

5. **Conclusion**

This paper focuses on the fault diagnosis task and proposes a new method based on deep learning. In this work, we built and trained a new model with a new structure based on the convolutional neural network. Building this method enables us to do end-to-end learning without a complex feature extraction process, and has outstanding performance and accuracy as well. This model incorporates multi-scale feature learning capability compared to a simple CNN structure and enhances the multi-scale interaction capability by adding a concatenation process between the hidden layers. This operation not only increases the accuracy and generalization capability of the model significantly but also reduces the training cost compared to a Multi-Scale CNN method which has 3 individual branches. The proposed method achieved high accuracy with a relatively small scale, which means this model has

strong generalization capability, enabling it to fit different kinds of datasets in different industrial situations. In future work, we might combine the multi-branch training process and the interaction process and build a larger model to fit in more complicated datasets and further improve its performance.

Reference

[1] Li D, Wang Y, Wang J, et al. Recent advances in sensor fault diagnosis: A review[J]. Sensors and Actuators A: Physical, 2020, 309: 111990.

[2] Gao Z, Ding S X, Cecati C. Real-time fault diagnosis and fault-tolerant control[J]. IEEE Transactions on industrial Electronics, 2015, 62(6): 3752-3756.

[3]Chen H, Jiang B, Ding S X, et al. Data-driven fault diagnosis for traction systems in high-speed trains: A survey, challenges, and perspectives[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 23(3): 1700-1716.

[4] Gonzalez-Jimenez D, Del-Olmo J, Poza J, et al. Data-driven fault diagnosis for electric drives: A review[J]. Sensors, 2021, 21(12): 4024.

[5] Cen J, Yang Z, Liu X, et al. A review of data-driven machinery fault diagnosis using machine learning algorithms[J]. Journal of Vibration Engineering & Technologies, 2022, 10(7): 2481-2507.

[6] Chandra M A, Bedi S S. Survey on SVM and their application in image classification[J]. International Journal of Information Technology, 2021, 13(5): 1-11.

[7]Abdullah D M, Abdulazeez A M. Machine learning applications based on SVM classification a review[J]. Qubahan Academic Journal, 2021, 1(2): 81-90.

[8] Tuerxun W, Chang X, Hongyu G, et al. Fault diagnosis of wind turbines based on a support vector machine optimized by the sparrow search algorithm[J]. IEEE Access, 2021, 9: 69307-69315.

[9] Wang M, Chen Y, Zhang X, et al. Roller bearing fault diagnosis based on integrated fault feature and SVM[J]. Journal of Vibration Engineering & Technologies, 2021: 1-10.

[10] Cao Y, Ji Y, Sun Y, et al. The fault diagnosis of a switch machine based on deep random forest fusion[J]. IEEE Intelligent Transportation Systems Magazine, 2022, 15(1): 437-452.

[11] Wan L, Gong K, Zhang G, et al. An efficient rolling bearing fault diagnosis method based on spark and improved random forest algorithm[J]. Ieee Access, 2021, 9: 37866-37882.

[12] Xiong R, Sun W, Yu Q, et al. Research progress, challenges and prospects of fault diagnosis on battery system of electric vehicles[J]. Applied Energy, 2020, 279: 115855.

[13] Zhu Z, Lei Y, Qi G, et al. A review of the application of deep learning in intelligent fault diagnosis of rotating machinery[J]. Measurement, 2023, 206: 112346.

[14] Surendran R, Khalaf O I, Tavera Romero C A. Deep learning based intelligent industrial fault diagnosis model[J]. Computers, Materials & Continua, 2022, 70(3).

[15] Pinaya W H L, Vieira S, Garcia-Dias R, et al. Autoencoders[M]//Machine learning. Academic Press, 2020: 193-208.

[16] Yang Z, Xu B, Luo W, et al. Autoencoder-based representation learning and its application in intelligent fault diagnosis: A review[J]. Measurement, 2022, 189: 110460.

[17] Wu X, Zhang Y, Cheng C, et al. A hybrid classification autoencoder for semi-supervised fault diagnosis in rotating machinery[J]. Mechanical Systems and Signal Processing, 2021, 149: 107327.

[18] Bhatt D, Patel C, Talsania H, et al. CNN variants for computer vision: History, architecture, application, challenges and future scope[J]. Electronics, 2021, 10(20): 2470.

[19] Ruan D, Wang J, Yan J, et al. CNN parameter design based on fault signal analysis and its application in bearing fault diagnosis[J]. Advanced Engineering Informatics, 2023, 55: 101877.

[20] Wang X, Mao D, Li X. Bearing fault diagnosis based on vibro-acoustic data fusion and 1D-CNN network[J]. Measurement, 2021, 173: 108518.

[21] Huang T, Zhang Q, Tang X, et al. A novel fault diagnosis method based on CNN and LSTM and its application in fault diagnosis for complex systems[J]. Artificial Intelligence Review, 2022, 55(2): 1289-1315.

[22] Chen X, Zhang B, Gao D. Bearing fault diagnosis base on multi-scale CNN and LSTM model[J]. Journal of Intelligent Manufacturing, 2021, 32(4): 971-987.

[23]. LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[24]Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

[25] Jiang G, He H, Yan J, et al. Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox[J]. IEEE Transactions on Industrial Electronics, 2018, 66(4): 3196-3207.

[26] Neupane D, Seok J. Bearing fault detection and diagnosis using case western reserve university dataset with deep learning approaches: A review[J]. Ieee Access, 2020, 8: 93155-93178.

[27] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines[C]//Proceedings of the 27th international conference on machine learning (ICML-10). 2010: 807-814.

[28] Hecht-Nielsen R. Theory of the backpropagation neural network[M]//Neural networks for perception. Academic Press, 1992: 65-93.

[29] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. The journal of machine learning research, 2014, 15(1): 1929-1958.