Fault diagnosis method for worm gearbox using convolutional network and ensemble learning

J C Hsiao*1, 2, Kumar Shivam2, T Y Kam1

1Department of Mechanical Engineering, National Chiao Tung University, 1001 University Road, Hsinchu 300, Taiwan, ROC
2Precision Machinery Research & Development Center, Taichung, Taiwan
No.27, 37th Road, Taichung Industrial Park, Taichung 40768, Taiwan, ROC

*E-mail: jchsiao.me01g@nctu.edu.tw

Abstract. Worm gearboxes are popular across various industrial applications since they offer significant gear ratios in small installation spaces. Despite having multiple advantages, worm gearboxes are subjected to higher friction due to sliding design and are prone to damage and increased transmission error over the period of operation. Delayed diagnosis of worm gearbox degradation can lead to low-quality products and/or unnecessary production line downtime. Using vibration characteristics of worm gearbox, it is possible to determine the fault and transmission error at a given period in time. In this paper, an ensemble machine learning model is trained and deployed to monitor the transmission error of worm gearbox and classify between new, operational and old conditions. 1D CNN (one dimensional convolutional neural network) model is used to automatically extract features in vibration signal of X, Y, and Z axes and predict the relevant state of worm gear. The proposed technique uses ensemble machine learning technique fusion of features extracted by multi-layer 1D CNN for three axes vibration data. The proposed method could achieve 96% accuracy and performs significantly better than traditional sequential and ensemble machine learning models on a dataset of 7,870 samples with 800 samples labeled as new condition, 4,740 samples as operational and 2,330 as old.

1. Introduction

Worm gearboxes are used widely in machinery where space-saving is critical for design. Worm gearboxes can provide significantly higher gear ratio than traditional transmission in a very small size factor. Worm gearboxes are subjected to higher friction due to their sliding design. However, fault detection using vibration analysis technique is rather difficult [1]. As the production rate and quality become significant for an organization, the unexpected downtime caused due to machine failure can cause great loss. Similarly, a degraded piece of machinery can become a liability when it comes to quality control. The same concept is true for worm gearboxes since they are an integral part of power transmission in modern machinery. In a previous study [1], kurtograph and frequency spectrum of vibration data are used to identify the pitting fault of worm gears. Since the popularity of machine learning and ANN (artificial neural networks), many studies have tried to apply ANN and machine learning concepts to develop a method for fault detection in rotating machinery and worm gearbox. In a study [2], authors have used a support vector machine to learn the statistical features (e.g. RMS (root mean square), mean, etc.) of the vibration of an automobile gearbox. Another study used statistical features of sound signals for fault diagnosis of rotating machinery using a decision tree algorithm [3]. Concept of machine learning has also been applied to fault monitoring of dc motors using 1D CNN (one...
dimensional convolutional neural networks) to find features in motor current signals [4]. A review paper has analyzed the fault detection methods for worm gearboxes and has mentioned that machine learning techniques can provide better classification of fault and an automatic approach to fault detection [5].

Since the rise of deep learning algorithms, many studies have explored the application of deep learning algorithms for fault diagnosis of rotating machinery. Deep Boltzmann Machines, Deep Belief Networks, and Stacked Auto-Encoders can be applied to time-series and frequency series data to achieve reliable accuracy for fault diagnosis of rolling bearing [6]. The CNN (convolutional neural network) has proven to be a reliable method for feature extraction. CNN with Dense Neural networks has shown better feature learning and classification results than traditional support vector machines, Feed-forward neural networks, and decision tree algorithms [7]. CNN can be used for automatic feature extraction from the raw signal of multiple sensors and has shown better accuracy than methods like support vector machine and KNN (K-nearest neighbor) algorithm where manual selection of features are required [8]. While using CNN for feature extraction, no pre-processing of the signal is required, as the CNN networks can be pre-trained and applied to a different system using transfer learning to require a very small dataset for retraining [9]. Adaptive CNN techniques can be developed to analyze multiple sensor data (sensor fusion) in order to diagnosis the health status of rotating machinery. An adaptive CNN based fault diagnosis method was proposed in [10], which uses DNN (dense neural networks) for feature fusion of CNN extracted features of multiple sensors. In a recent year many other studies have also used CNN to extract features from multiple sensors and then train DNN for fault diagnosis of gearboxes or rotating machine parts [11]–[13].

In this study, we propose a fault diagnosis method for worm gearbox using a convolutional network and ensemble learning. By collecting the vibration data from a test bench designed to emulate the operation of a worm gear working in industrial conditions, and building an ensemble model of convolutional networks to systematically learn and detect anomalies in the data. By using supervised learning, we train the model to learn the low transmission error and high transmission error vibration characteristics of a worm gear, which are often hard to visualize using traditional vibration analysis methods. The paper discusses the underlying idea behind convolutional neural networks in section 2 and the ensemble learning in section 3. Section 4 provides the details of the test platform, data collection methods and the ensemble learning model used for this research. Section 5 discusses the results obtained during this research and the conclusion is provided in section 6.

2. Machine Learning For Time-series Data Classification

Vibration is a classical example of time-series data[14]. For our comparison, we consider using several machine learning models including MLP (multi-layer perceptron), CNN, and ensemble models. Models that utilize DNN are considered as complex machine learning models [15], and CNN is a bio-inspired neural network [16]. A brief introduction of these machine learning techniques is given in below sections.

2.1 Multi-Layer Perceptron

Multi-layer perceptron is one of the most traditional architectures of deep learning models. In this architecture each neuron in a layer, \( l_i \), are connected to each neuron in the layer \( l_{i-1} \) with \( i \in [1, L] \), where L is the number of last layer in MLP architecture. The connections between the neurons are modeled using weights in a neural network. A general form of applying a non-linearity to an input time-series X can be represented as following equation:
\[ A_{l_i} = f(\omega_{l_i} \ast X + b) \]  

(1)

Where \( \omega_{l_i} \) is the set of weights with length and number of dimensions identical to X's, b, the bias term and \( A_{l_i} \), the activation of the neurons in layer \( l_i \). The number of neurons in a layer is considered a hyperparameter.

MLP is used for time-series data because they do not exhibit any spatial invariance, meaning each timestamp/data element has its own weight and the temporal information is lost. In this architecture, each time-series element is treated independently from each other. MLP architecture for classification the final layer is a discriminative layer that takes as input the activation of previous layer and gives a probability distribution over the classes in the dataset. In machine learning approach, commonly a softmax activation function is employed due to the sum of the probabilities for the output classes is guaranteed to be equal to 1. The result of a softmax function is defined as follow:

\[ \hat{Y}_j(X) = \frac{e^{A_{L-1} \ast \omega_j + b_j}}{\sum_{k=1}^{K} e^{A_{L-1} \ast \omega_k + b_k}} \]  

(2)

\( \hat{Y}_j \) represents the probability of X having the class Y equal to class \( j \) out of \( K \) classes in the dataset. The set of weights \( \omega_j \) (and the corresponding bias \( b_j \)) for each class \( j \) are linked to each previous activation in layer \( l_{L-1} \).

### 2.2 Convolutional Neural Networks

Convolutional neural networks are also known as ConvNet or CNN are bio-inspired neural networks that mimic the structure of visual system [16]. It has successfully been applied in detection, segmentation, and recognition of regions or patterns in images, speech, and time-series data [17]–[19]. CNN is now the most dominant approach for most image detection and speech recognition task. Motivated by the success of these CNN architectures in these various domains, researchers have started adopting them for the time-series analysis [20]. Figure 1 is a generic representation of CNN architecture, which is structured by a series of stages. The first few stages are a combination of two types of layer: convolutional layer and pooling layer, while the last stage to the architecture consists of a fully connected layer and a traditional classification model[21].

![Figure 1. General Architecture of a CNN](image)

The convolutional layer contains a number of filters, which convolute the input from the previous layer through a set of weights and compose a feature output, also known as a feature map. Within each filter, neurons are directly connected to the input data and multiply the data points by the assigned
weights. All neurons in the same filter share their weights, which reduces the optimization time and complication of CNN. Let us assume, the convolutional layer is \( X \in \mathbb{R}^{A \times B} \), where, A and B are the dimensions of the input data, then the output of the convolutional layer can be calculated as following [22]

\[
C_{cn} = f(X \ast W_{cn})
\]

In equation (3), \( \ast \) is an operator of a convolution; \( C_{cn} \) is the \( cn \)-th feature map of the convolutional layer, and the number of filters is \( cn \); \( X \) is the input data matrix; \( W_{cn} \) is the weight matrix of \( cn \)-th filter of the current layer; \( b_{cn} \) is the denotation of \( cn \)-th bias; and \( f \) (an activation function) is applied to the result, whereas typically rectified linear units (ReLU), hyperbolic tangent or sigmoid function.

However, in time-series data application, a modified version of the 2D convolution is applied as a 1D convolution layer. 1D convolution can be visualized as a sliding filter over the time-series in our case vibration data. Different from two-dimensional data such as images (height and width), for one-dimensional data like a time-series, these filters only exhibit one dimension (time). These filters can also be summarized as a generic non-linear transformation of a time-series. For example, convoluting (multiplying) a filter of length 3 with a univariate time-series, by setting filter values to \( [1, \frac{1}{3}, \frac{1}{3}] \), the convolution will result as a moving average with a sliding window of length 3. A general equation of applying the convolution for a centered time stamp \( t \) can be written as following:

\[
C_t = f(\omega \ast X_{t-l/2:t+l/2} + b) \quad \forall \ t \in [1, T]
\]

where \( C \) denotes the result of a convolution (dot product \( \ast \)) applied on a univariate time-series \( X \) of length \( T \) with a filter \( \omega \) of length \( l \), a bias parameter \( b \) and a final non-linear function \( f \) such as the Rectified Linear Unit (ReLU). The result of a convolution (one filter) on an input time-series \( X \) can be considered as another univariate time-series \( C \) that underwent filtering. Thus, applying several filters on a time-series will result in a multivariate time-series whose dimensions are equal to the number of filters used. An intuition behind applying several filters on an input time-series would be to learn multiple discriminative features useful for the classification task.

A pooling layer follows a convolution layer to obtain a lower resolution form of the convolution layer activations through sub-sampling. Neighbor pooling units take input from the patches that are shifted by more than one column or row, hence performing dimension reduction of the representation and making the model less sensitive to small shifts and distortions. The pooling function computes the statistics of the activation and can be applied by max-pooling, mean pooling or weighted pooling. In a max-pooling function, activations can be computed as following [22] :

\[
P_{cn} = \max_{C_t \in S}(C_t)
\]

In equation (4), \( P_{cn} \) is the pooling layer’s output; S is the block size. All the pooling layers of the pooling blocks will be arranged together as the output of the pooling layer in a way that the output will be \( S \) times smaller along both spatial dimensions. For example, for a convolution layer with dimension \( (200, 200) \) if the max-pooling of block size, \( S = 2 \) is applied, will result in a max-pooling layer of size \( (100, 100) \).

After several combinations of convolutional layers and pooling layers, fully connected layers are applied. Fully connected or dense networks are similar to the traditional multilayer neural networks and can be applied through different classification models. In this paper, we chose hidden layers followed by categorical cross-entropy classifier as our loss function to achieve maximum accuracy for the classifying between multiple classes of new, operational and old worm gear. Categorical cross-entropy
is a sigmoid activation followed by a cross-entropy loss[23]. The advantage of this method is that the loss computed for even CNN output vector component is not affected by the other components value.

### 2.3 Ensemble Learning

Ensemble learning consists of a set of individually trained classifiers e.g. neural networks or decision trees whose predictions are combined when classifying instances. Ensemble learning has shown more accurate classification results than any of the single classifiers in the ensemble [24], [25]. Combining the output of several classifiers is useful only if there is disagreement among the classifiers’ results. Methods of creating ensembles center around creating classifiers that disagree on their predictions. Generally, the model focuses on changing the training process, so that the resulting classifiers will produce different predictions. For example, neural networks that are trained only on a portion of the training set and their outputs are combined together for final classification.

There are various architectures of ensemble learning, e.g. boosting, bagging, etc.[25]. The classifier in ensemble learning has to be combined to get increased classification accuracy. Classifiers used in the model could be the same or different. In terms of features, the ensemble models may vary [26]. Figure 2(a) is a representation of ensemble model with feature ensemble method. In this method, different features of input data are learned first then ensemble to reduce the bias of individual features and then feed to a classifier for final classification, this method is also known as feature fusion. Figure 2(b) represents an ensemble method in which classifiers classify the input data based on different features and then the results of the classifier are ensemble as a final result. This is similar to a decision fusion model. In this paper, we use ensemble model for achieving higher accuracy, as feature fusion method removes the classifiers bias.

![Figure 2(a) Feature ensemble model for Artificial Neural Networks (b) Classifier Ensemble model for Artificial Neural Networks](image)

3. Experimental Investigation

The worm gearbox test platform (figure 3(a)) is designed for collecting the operational vibration data. Test platform consists of a servo motor as the power input, a worm gearbox (high precision, Detron GXA-125S) as transmission, a 25-bit rotary encoder with positional accuracy of 0.039 arcseconds for
measuring output speed and position of gearbox while a powder brake is used as load. The servo motor is controlled using a real-time PC-based (personal computer based) system, which also collects the real-time motor current (torque) data. The worm gearbox is equipped with three accelerometers (figure 3(b)) for collecting vibration data of X, Y and Z-axis simultaneously. Accelerometers are connected to NI-Compaq DAQ (NI-9181), a high-speed real-time data acquisition system which logs 2,000 data points per second at a sampling rate of 20KHz. An additional oil-debris sensor is added to monitor the oil condition during test. The transmission error of the worm gear is calculated using input position and output position. The worm gear is rotated clockwise 360° and then counter-clockwise, then the return position of output encoder attached to worm gearbox’s output shaft is compared with initial value with operational load. This operation is continued for 10 minutes (equal to 100 cycles).

\[
b_{wg} = \frac{1}{4} \sum_{n=1}^{4} (P_{fin} - P_{fin})
\]  

In equation (6) \(b_{wg}\) represents the average transmission error of the worm gearbox, where \(P_{fin}\) is the n-th test’s initial position recorded by the encoder and \(P_{fin}\) the n-th test’s final position recorded by the encoder upon return to the initial position. The number of tests used for calculating the averaging transmission error was 4 for this test.

In this study, experiment included collecting vibration data of worm gearbox without any load applied and calculating the transmission error of worm gear at the end of 10 minutes operation cycle and the test was continued until the transmission error was significantly high (>120 arcseconds) to determine the gearbox is no longer acceptable for the high precision applications. The operating load on the worm gearbox was set to be in a varying range from -20 Nm to 20 Nm and servo motor would repeat three clockwise and three counter-clockwise cycles to mimic the conditions of a CNC (computer numerical control) rotary table operation. For our investigation and machine learning model training, we use raw vibration data as shown in figure 4. In figure 4, we can see that collective total vibration samples acquired during test is equal to 1.574 x 10^7. In figure 5, the three classes for transmission error condition is shown followed by table. 1 with description of each classes meaning. Worm gear’s transmission error was measured 7,870 times during the whole experiment at equal interval of time (i.e. end of one operational cycle = 10 minutes).
Figure 4. Worm gear vibration data acquired during the accelerated wearing platform test platform for
(a) X-axis, (b) Y axis, (c) Z-axis
Figure 5. Class distribution over the vibration data set

Table 1. Classes description according to the measured transmission error of worm gear

| S. No. | Class Numeric Label | Class String Label | Number of data set | Class description |
|-------|---------------------|--------------------|--------------------|-------------------|
| 1.    | 0                   | ‘New’              | 800                | 30 arcsecond < Transmission error < 60 arcsecond |
| 2.    | 1                   | ‘Mid’              | 4,840              | 60 arcsecond < Transmission error < 100 arcseconds |
| 3.    | 2                   | ‘Old’              | 2,330              | 100 arcsecond < Transmission error < 120 arcseconds |

Table 1 describes the labeling method for classes used to separate the different vibration dataset. Class labeled as “New” represents the new or factory setup of the worm gear with transmission error in range of 30 to 60 arcsecond. ‘Mid’ label shows the operationally acceptable transmission error range of worm gear within 60 to 100 arcsecond. Operational range was observed to be stable, and the transmission error increased from 60 to 100 over a longer period (i.e. 48,400 minutes of operation). ‘Old’ is used to label the dataset that has a transmission error of over 100 arcseconds and is considered as repairable condition in some high precision applications such as aerospace parts manufacturing, etc. In figure 6(a), box plot representation of vibration data for different axes and for three classes are represented. In addition to that in figure 6(b), a violin plot representing the probability density distribution of different axes vibration for different classes are also shown. From these two figures we can notice that there are many outlier present in our dataset, this could be caused by presence of noise during data acquisition or various environmental factors. From figure 6(b), it is clear that the probability distribution of vibration dataset is uni-modal for each axes for different classes. Due to the nature of vibration data being bi-direction, the mean and median values are approximately zero.
In this paper, four machine learning models are used to analyze and classify the vibration data of the worm gearbox. In Figure 7 (a), a generic MLP model is shown that is used to analyze three-axis vibration data. Using this model X, Y and Z axis vibrations are classified into ‘New’, ‘Mid’ and ‘Old’ categories using softmax function (equation 2). In Figure 7(b), a simple CNN architecture is used with two 1D CNN layers to convolute and find automatically find features in the vibration dataset. The classifier activation function is softmax and “categorical cross-entropy” is used as the loss function for optimization during the training. In Figure 7(c), a deep CNN model is used and 3 axis vibration data is fed into the model simultaneously. A similar model has been used previously by [4], [8] for analyzing vibration data. Classifier activation function and loss function are same as a simple CNN model. In
Figure 7 (d), proposed method in this paper for analyzing three-axis vibration data of worm gearbox is shown. In this method, X, Y and Z axis data features are extracted and learned bi-axially and then their average weights are used to train final layer of dense neural network, also known as fully connected layer. This model is inspired by the ensemble machine learning and assumes that by separate evaluating features of a pair of two-axis vibration data, X and Y, Y and Z, and X and Z, we can find more robust features that correlate the axial vibration to the worm gear's transmission error, so that model can perform better even in absence of features or any strong activity (amplitude changes) in either one of the axis vibration data. From figure 4 we can notice that change in vibration pattern over time for either axis is not evident enough to visually establish a classification. In figure 4(c), Z-axis vibration seems to increase in amplitude over the time as the transmission error increases but on the vibration scale the difference is very minute. Thus, by training a model using bi-axial vibration data and by combining the features of three bi-axial pair we expect to find the spatial relationship between these pairs that would be better suited to discriminative for classification of dataset. Training of each model is performed using binary cross-entropy loss function and Adam, a first-order gradient-based stochastic optimizer for back-propagation. In figure 7, ‘Dense’ refers to a fully connected / MLP layer, ‘CNN’ refers to convolutional layer, ‘Avg. Pooling’ refers to a global average pooling layer. ‘Dense’ in Dense layer block in figure 7, followed by a numeric N e.g. ‘500N’ represents that there are 500 neurons in the layer, similarly ‘1D CNN’ in 1D CNN layer blocks followed by a numeric and F represents a number of filters present in layer (e.g. 64F represents 64 convolutional filters). For ensemble CNN + FCN model, a lasso regression regularization is applied to the classifier layer to avoid over-fitting and maximizing the accuracy of the model with a better generalization ability. Batch normalization is applied to each convolution layer to regularize the CNN outputs and before applying activation. This method allows the CNN layers to be trained faster by scaling the results of convolution operators. Architecture hyper-parameters of above mentioned models are listed in table 2, followed by optimization hyper-parameters of architectures in table 3. A comparison was made between different machine learning models to evaluate their accuracy and training time. The dataset includes 7870 vibration data set with 75% used as the training set and 25% as test sets. Test set data is used to evaluate the performance of each model precision, recall, and f1-score. The models are created in python programming language with Keras using TensorFlow backend. Scikit-learn package is used for measuring performance.
Ensemble CNN + FCN Model for bi-axial feature fusion analysis

Table 2. Architecture's hyper-parameters for the machine learning approaches

| Method               | #Layers | #Conv | Normalize | Pooling | Features     | Activation | Regularize |
|----------------------|---------|-------|-----------|---------|--------------|------------|------------|
| MLP                  | 3       | 0     | None      | None    | Fully Connected | ReLU       | Dropout    |
| Simple CNN           | 3       | 2     | None      | None    | convolution   | ReLU       | None       |
| Deep CNN             | 12      | 7     | None      | Max Pooling | Global Average Pooling | ReLU       | Dropout    |
| Ensemble CNN + FCN   | 41      | 18    | Batch     | Max Pooling | Fully Connected | Leaky ReLU | None       |

Table 3. Optimization’s hyper-parameters for the machine learning approaches

| Method               | Algorithm | Valid     | Loss   | Epochs | Batch | Learning rate |
|----------------------|-----------|-----------|--------|--------|-------|---------------|
| MLP                  | ADAM      | Split 25% | Entropy| 1000   | 128   | 0.01          |
| Simple CNN           | ADAM      | Split 25% | Entropy| 500    | 64    | 0.01          |
| Deep CNN             | ADAM      | Split 25% | Entropy| 500    | 64    | 0.001         |
| Ensemble CNN + FCN   | ADAM      | Split 25% | Entropy| 500    | 64    | 0.001         |

4. Results and Discussions

Machine learning models mentioned in section 3 are trained using 7,870 vibration datasets with 5,902 as the training set and 1,968 as the test set. From table 4, we can see that MLP architecture ended up with 100% accuracy with a loss of 0.1%; however it could only achieve a validation accuracy of 86.5% with 81.6% validation loss. Similarly, simple CNN architecture could achieve a training accuracy of 91.2% but could only achieve a validation accuracy of 88.1%. Deep CNN model could train much better than the previous two models with training accuracy of 100% and validation accuracy of 94.5%. This confirms that the Deep CNN model could learn features in the X, Y, and Z-axis vibration dataset much accurately. Our Ensemble CNN + FCN model also achieves same training accuracy as Deep CNN but provides a better validation accuracy at 95.2%. Also, it can be seen that our model was able to converge faster at 157th epoch compared to Deep CNN’s 329th epoch.
Table 4. Training results of machine learning approaches

| Method          | Training Accuracy | Training Loss | Validation Accuracy | Validation Loss | Best Model Epoch |
|-----------------|-------------------|---------------|---------------------|-----------------|------------------|
| MLP             | 1.000             | 0.001         | 0.865               | 0.816           | 376              |
| Simple CNN      | 0.912             | 0.240         | 0.881               | 0.257           | 264              |
| Deep CNN        | 1.0               | 0.012         | 0.945               | 0.301           | 329              |
| Ensemble CNN + FCN | 1.0                 | 0.016       | 0.952               | 0.621           | 157              |

After, training of models, the best models of each architecture were saved and loaded to predict the classes on 1,968 test datasets. Precision, recall, accuracy, and loss was tested for architectures and listed in table 5. Above metrics were calculated using the following equations:

\[
\text{Precision} = \frac{\sum (\text{True positive}/\text{True positive} + \text{False Positive})}{\text{number of classes}} \tag{7}
\]

\[
\text{Recall} = \frac{\sum (\text{True positive}/\text{True positive} + \text{False Negative})}{\text{number of classes}} \tag{8}
\]

\[
\text{Accuracy} = \frac{\sum (\text{True positive} + \text{True Negative})}{\text{Total number of datasets}} \tag{9}
\]

Using equation 7, 8, 9 and while loss being machine learning models categorical cross-entropy loss, we can measure the performance of predictions of machine learning architectures. From table 5, we can see that MLP and simple CNN are only able to achieve an accuracy of 86% with simple CNN having better precision and recall than MLP. It suggests CNN model performs better on time-series data than MLP. Deep CNN model was achieving a remarkable accuracy of 95.2% with a precision of 95% and recall of 96%. Our model, Ensemble CNN + FCN architecture was able to achieve a better result than Deep CNN with an accuracy of 97.3% and precision and recall both being above 97%.

Table 5. Test results of machine learning approaches

| Method              | Test results |
|---------------------|--------------|
|                     | Precision    | Recall | Accuracy | Loss  |
| MLP                 | 0.874        | 0.840  | 0.865    | 2.162 |
| Simple CNN          | 0.891        | 0.864  | 0.869    | 1.212 |
| Deep CNN            | 0.960        | 0.952  | 0.952    | 0.410 |
| Ensemble CNN + FCN  | 0.971        | 0.973  | 0.973    | 0.527 |

In figure 8, confusion matrix of predictions of each architecture is shown with the x-axis being
predicted classes and the y-axis being true classes. The results are normalized using \( r = \frac{r_p}{r_t} \), where \( r \) is the normalized value, \( r_p \) is the number of predictions belong to the given class and \( r_t \) is the total number of datasets belonging to the given class. From the confusion matrix of Deep CNN and our Ensemble CNN + FCN, we can observe that both models were able predict ‘Mid’ class with 96% accuracy, but our model was able to predict the ‘New’ and ‘Old’ classes better at accuracy of 98% and 97% compared to that of Deep CNN’s 96% and 94%.

![Confusion Matrix](image)

Figure 8. Confusion matrix of predicted results of machine learning approaches

5. Conclusion

Using machine learning for vibration analysis and fault detection is a rising field of study. However, there is no universal solution or machine learning model that can be applied to a different kind of fault analysis tasks. In this paper, we collected vibration signal of the worm gearbox at different transmission error levels at no loading condition. Without any loading condition, worm gearbox vibration signals for X, Y and Z-axis showed a very minute and slow increase in amplitude over the accelerated testing period. Therefore, we compared existing machine learning models used for the time-series classification and tried to develop a machine learning model that can outperform traditional approaches in the task of classifying worm gearbox vibration dataset. In our experiment, the vibration signal had very minute difference in amplitude from beginning to the end of dataset due to vibration signals were acquired without any loading condition on worm gear. Therefore, by employing various traditional machine learning architectures, we explored the results. We discovered that Deep CNN architecture was able to
perform significantly better than MLP and simple CNN. Intuitively, by analyzing pair of axes' vibration data, we developed an ensemble CNN + FCN model for our application, which was able to outperform Deep CNN, one of most utilized architecture for automating feature extraction and deep learning. Our proposed method was able to better generalize the test vibration datasets using bi-axial vibration features extracted using multiple 1D CNN layers and provided higher accuracy on predicting classes on which Deep CNN underperformed (‘New’ and ‘Old’). As stated in [24], [25], [27], ensemble learning classification method performs better than single classifiers.

It is important to note that during this study the ambient and worm gearbox temperature was not taken in account, the worm gearbox was warmed up for 100 rotations at 10 Nm load. At lower temperature the transmission error of worm gear was recorded to be higher than after warm up cycle. The operating conditions were also artificially created using a powder brake and there was minimum vibration interference from surroundings since the test was conducted in a laboratory. For the future study, the authors will work on collecting data from industrial environment and also recording the transmission error trend with respect to temperature change. Future study will focus on building a robust machine learning model that can provide good performance even in industrial environment with varying operating conditions.

Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] F. Elasha, D. Mba, C. R.-C. G. Kiat, I. Nze, and G. Yebra, “Diagnostics of worm gears with vibration analysis,” 11th Int. Conf. Cond. Monit. Mach. Fail. Prev. Technol. C. 2014 / MFPT 2014, no. July, 2014.

[2] T. Praveenkumar, M. Saimurugan, P. Krishnakumar, and K. I. Ramachandran, “Fault diagnosis of automobile gearbox based on machine learning techniques,” Procedia Eng., vol. 97, p 2092–2098, 2014.

[3] M. Saimurugan and R. Nithesh, “Intelligent Fault Diagnosis Model for Rotating Machinery Based on Fusion of Sound Signals,” Int. J. Progn. Heal. Manag., p 1–10, 2016.

[4] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, “Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks,” IEEE Trans. Ind. Electron., vol. 63, no. 11, p 7067–7075, 2016.

[5] S. B. Ghodake, P. A. K. Mishra, and P. A. V Deokar, “A Review Paper on Fault Detection of Worm Gearbox,” vol. 3, no. 1, p 161–164, 2016.

[6] Z. Chen, S. Deng, X. Chen, C. Li, R. V. Sanchez, and H. Qin, “Deep neural networks-based rolling bearing fault diagnosis,” Microelectron. Reliab., vol. 75, no. June, p 327–333, 2017.

[7] L. Jing, M. Zhao, P. Li, and X. Xu, “A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox,” Meas. J. Int. Meas. Confed., vol. 111, no. July, p 1–10, 2017.

[8] M. Xia, T. Li, L. Xu, L. Liu, and C. W. De Silva, “Fault Diagnosis for Rotating Machinery Using Multiple Sensors and Convolutional Neural Networks,” IEEE/ASME Trans. Mechatronics, vol. 23, no. 1, p101–110, 2018.

[9] P. Cao, S. Zhang, and J. Tang, “Preprocessing-Free Gear Fault Diagnosis Using Small Datasets with Deep Convolutional Neural Network-Based Transfer Learning,” IEEE Access, vol. 6, no. 860, p
26241–26253, 2018.

[10] L. Jing, T. Wang, M. Zhao, and P. Wang, “An adaptive multi-sensor data fusion method based on deep convolutional neural networks for fault diagnosis of planetary gearbox,” Sensors (Switzerland), vol. 17, no. 2, 2017.

[11] Y. Elyassami, K. Benjelloun, and M. Er-rajouany, “GEAR FAULT DIAGNOSIS AND CLASSIFICATION USING DATA VIBRATION,” vol. 12, no. 19, p 5617–5625, 2017.

[12] L. Zhang, H. Gao, J. Wen, S. Li, and Q. Liu, “A deep learning-based recognition method for degradation monitoring of ball screw with multi-sensor data fusion,” Microelectron. Reliab., vol. 75, p215–222, 2017.

[13] R. C. Ümütlü, B. Hizarci, H. Ozturk, and Z. Kiral, “Pitting detection in a worm gearbox using artificial neural networks,” Proc. INTER-NOISE 2016 - 45th Int. Congr. Expo. Noise Control Eng. Towar. a Quieter Futur., p 6526–6534, 2016.

[14] S. Hussain and H. A. Gabbar, “Vibration Analysis and Time-series Prediction for Wind Turbine Gearbox Prognostics,” p 1–11, 2013.

[15] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, “Efficient BackProp BT - Neural Networks: Tricks of the Trade,” G. B. Orr and K.-R. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, p 9–50.

[16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, p 436, May 2015.

[17] S. C. Turaga et al., “Convolutional Networks Can Learn to Generate Affinity Graphs for Image Segmentation,” Neural Comput., vol. 22, no. 2, p511–538, Nov. 2009.

[18] F. Noman, C.-M. Ting, S.-H. Salleh, and H. Ombao, “Short-segment heart sound classification using an ensemble of deep convolutional neural networks,” p 1–7, 2018.

[19] D. Cireșan, U. Meier, J. Masci, and J. Schmidhuber, “Multi-column deep neural network for traffic sign classification,” Neural Networks, vol. 32, p 333–338, 2012.

[20] J. C. B. Gamboa, “Deep Learning for Time-Series Analysis,” 2017.

[21] A. LeNail, “NN-SVG: Publication-Ready Neural Network Architecture Schematics,” J. Open Source Softw., vol. 4, no. 33, p747, 2019.

[22] J. Bouvrie, “Notes on Convolutional Neural Networks,” 2006.

[23] S. Mannor, D. Peleg, and R. Rubinstein, “The cross entropy method for classification,” 2006, pp. 561–568.

[24] D. Opitz and R. Maclin, “Popular ensemble learning: an empirical study,” J. Artif. Intell. Res., vol. 11, p 169–198, 1999.

[25] H. Li, X. Wang, and S. Ding, “Research and development of neural network ensembles: a survey,” Artif. Intell. Rev., vol. 49, no. 4, p455–479, 2018.

[26] S. Asmita and K. K. Shukla, “Review on the Architecture, Algorithm and Fusion Strategies in Ensemble Learning,” Int. J. Comput. Appl., vol. 108, no. 8, p 21–28, 2014.

[27] M. Shao, X. J. Zhu, H. F. Cao, and H. F. Shen, “An artificial neural network ensemble method for fault diagnosis of proton exchange membrane fuel cell system,” Energy, vol. 67, p 268–275, 2014.