A Recommendation Mechanism of Selecting Machine Learning Models for Fault Diagnosis

Wen-Lin Sun[a], Yu-Lun Huang[b] and Kai-Wei Yeh[c]
Department of Electronics and Electrical Engineering, National Yang Ming Chiao Tung University, Hsinchu City, Taiwan, Republic of China

Keywords: Smart Manufacturing, Industry Automation, Fault Diagnosis, Machine Learning.

Abstract: Faults of a machine tool generally lead to a suspension of a production line when the defeated parts need a long lead time. The prevention of such suspension depends on the health condition of machine tools in a factory. Hence, monitoring the health conditions of machine tools with modern Machine Learning (ML) technologies is one of the highlights of industry evolution 4.0. Though researchers presented several methods and mechanisms to solve the fault detection and prediction of machine tools, the current works usually focus on deploying one ML algorithm to one specific machine tool and generating a well-trained model for fault diagnosis and detection for that machine tool, which are impractical since a factory typically runs a variety of machine tools. This paper presents an Automatic Fault Diagnosis Mechanism (AFDM), taking historical data provided by an administrator and then recommending a machine-learning algorithm for fault diagnosis. AFDM can handle different types of data, diagnose faults for different machine tools, and provide a friendly interface for a factory administrator to select a proper analytical model for the specified type of machine tools. We design a series of experiments to prove the diversity, feasibility, and stability of AFDM.

1 INTRODUCTION

Industry Evolution 4.0 promises new potential by integrating modern technologies with machine tools, including the Internet of Things (IoT), cyber-physical systems, and cloud computing. Such potential brings the trend of smart manufacturing. The concept of smart manufacturing innovates the existing manufacturing processes and achieves more intelligent features and applications. One of the smart manufacturing applications is intelligent maintenance of machine tools (Kumar and Galar, 2018). After long-term operations, the fatigue of machine components is inevitable, which may reduce the production quality. Administrators need to stop the production lines while waiting for the defeated components, which decreases the throughput of production lines. Thus, it is crucial to monitor the health conditions of machine tools and diagnose faults in advance.

Recently, a standard solution has been to use an ML model to diagnose the faults of the target machine tools. For example, FANUC, one of the largest manufacturers of factory automation systems, has presented a novel service for monitoring the health conditions of spindles in Computer Numerical Control (CNC). This service first collects the historical data (e.g., torque values) from spindles and then trains the ML model for predicting the anomaly score of spindles. This service calculates the anomaly score based on measured data during online monitoring and warns the administrator if the score exceeds a threshold specified by the administrator. Besides, researchers have leveraged different ML models for various machine components, such as gearbox (Jia et al., 2016), centrifugal pump (Wen et al., 2017), and drill bit (Thirukovalluru et al., 2016). These solutions work well but are only dedicated to specific machines and thus may not be practical enough since most factories run more than one type of machine tool for production. Thus a factory administrator may need a generic fault diagnosis mechanism with a proper ML model to develop smart factories with various machines. Moreover, most of the above solutions consider accuracy the only criterion for selecting the ML model for their machines. These solutions may skip some essential criteria like the computation time required for generating the prediction results. The so-

[a] https://orcid.org/0000-0003-4298-037X
[b] https://orcid.org/0000-0001-7618-0114
[c] https://orcid.org/0000-0001-6262-2137

Sun, W., Huang, Y. and Yeh, K.
A Recommendation Mechanism of Selecting Machine Learning Models for Fault Diagnosis.
DOI: 10.5220/0011287000003271
In *Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2022)*, pages 49-57
ISBN: 978-989-758-585-2; ISSN: 2184-2809

lutions may not be suitable for the environment or a specific service without careful consideration. For example, a model generating an accurate prediction may require a long computation time. The model may not be suitable for a production line requiring real-time analysis.

Factory administrators need a mechanism to recommend the best-fit model according to their preferences and the production requirements. The mechanism should address the above issues: (1) selecting the best-fit model for a machine tool to monitor its health conditions; and (2) considering multiple criteria when selecting the best-fit model.

Hence, we propose a mechanism, AFDM, to automatically recommend the best-fit ML model according to the historical data of the specified machine and the preference given by the factory manufacturer.

2 RELATED WORK

ML algorithms can solve critical problems like fault diagnosis of machine tools. During the fault diagnosis process, the ML algorithms train and generate the corresponding classification models to find or predict potential faults from different components of machine tools (Leukel et al., 2021). Since the characteristics of the data or signals collected from different machine tools vary a lot, the most challenging part of applying ML algorithms to the fault diagnosis is how to get a proper algorithm for a specific machine tool. This section reviews the research about fault diagnosis mechanisms of machine tools that adopt ML-based classification algorithms. The detailed comparison between the related research and AFDM is discussed in Section 5.

Sun et al. (Sun et al., 2017) presented a bearing fault diagnosis method based on compressed sensing (Donoho, 2006) and deep learning. They presented an intelligent diagnosis system with two steps, data preprocessing and fault classification. For data preprocessing, they used compressed sensing data to perform dimension reduction. They used a Stacked Sparse Autoencoder (SSAE) with Softmax function as the classification model for fault classification. They tried different model parameters, including compression ratio, number of neurons, sparsity parameter, and decay parameter, to verify the impacts on performance by these parameters. They compared their work with Support Vector Machine (SVM) and Multi-layer Perceptron (MLP) by classification accuracy. Sun's work is typical research using one algorithm for one specific type of machine tool.

Selecting a proper algorithm (model) for a specific machine tool is challenging. Brecher et al. (Brecher et al., 2017) presented a strategy for training several ML models (e.g., SVMs, k-Nearest Neighbors, k-Means) with different data features to determine which combination had the best classification accuracy for a specific machine tool. In this work, the authors estimated the state of a packing machine and monitored the health condition of the belts of the packing machine to predict faults in advance. These actions could reduce unplanned downtime. Brecher's work showed that using different classification models could obtain different accuracies. They selected a model for deployment based on accuracy. However, the authors did not explain how to select a model when encountering multiple criteria during the selection.

In 2016, Thirukovalluru et al. (Thirukovalluru et al., 2016) presented a fault diagnosis approach enabled by Deep Neural Network (DNN). The work aimed to analyze the difference in the performance of a classification model when using the standard features and the features generated by DNN. The authors ran DNN with SVM and Random Forest. Thirukovalluru's approach assessed the performance of classification models by their accuracies. The results proved that a model could improve the classification accuracy with the features generated by DNN, especially for the drilling bits. The results also showed that one single model could not work well for all types of machine tools.

In summary, all the mentioned works performed data preprocessing when dealing with signals from machine tools, deployed ML algorithms to classify the processed data, and selected a suitable model based on the classification accuracy. However, none of them has considered the multiple types of machine tools and the multiple criteria for model selection. To address these two issues, we generalize these methods and propose AFDM in Section 3. AFDM adopts multiple ML algorithms for multiple types of machine tools. AFDM trains, evaluates and ranks the models for a specified machine tool when considering multiple criteria.

3 AFDM

By leveraging modern ML algorithms, smart factories can predict machine faults in advance, increase the production line's throughput, and reduce manufacturing costs. We propose AFDM, an Automatic Fault Diagnosis Mechanism, for selecting the best-fit classification model to diagnose and predict faults for different machine tools and help the factories build

up an intelligent manufacturing system. By adopting multiple-criteria decision-making (MCDM) methods to AFDM, we can recommend better-fit classification models according to the characteristics of the collected data and the customized requirements (e.g., limited training time) to diagnose faults of machine tools.

3.1 Overview

AFDM provides an objective way to help factory administrators get more insights into their machine tools. Figure 1 illustrates the fault diagnosis procedure of AFDM and its implementation. The procedure contains four phases, including Training, Evaluation, Selection, and Diagnosis (Phase I to IV in Figure 1, respectively).

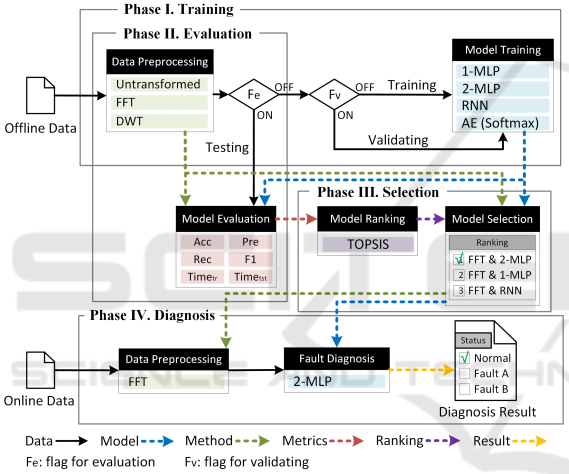

Figure 1: The Fault Diagnosis Procedure of AFDM.

By reviewing the existing literature, we can find a general fault diagnosis procedure that includes model training and fault diagnosis. Generally, the existing methods train a classification model with offline data. Then, the trained model is deployed to a field and analyzes the online data in the fault diagnosis phase. AFDM revises the traditional procedure and adds two more phases (Evaluation and Selection) for an automatic recommendation.

Different from other methods, AFDM trains multiple classification models with different ML algorithms at a time. When an administrator launches AFDM in a factory, the administrator collects data from the machines in the factory. In the first phase, the administrator enters the collected data to AFDM for training multiple classification models (e.g., 1-MLP, 2-MLP, RNN, and AE (Softmax) in Figure 1). Phase II estimates these models' performance (like accuracy.) Then, according to the evaluation metrics, AFDM ranks these models and recommends the best-fit model to the administrator in Phase III. If the administrator accepts the recommended model, AFDM diagnoses the data acquired from machine tools in the factory using the model selected in the Diagnosis phase. The following subsections detail the four phases.

3.2 Phase I: Training

We design the Training phase to tune classification models that may be candidates for the specified machine tool. In this phase, AFDM tunes the candidate classification models with the historical data collected from the target machine tool. In the Training phase, AFDM designs two primary operations for training multiple classification models: Data Preprocessing and Model Training (see Figure 1). Depending on the data type, some data cannot be analyzed in its raw format. For example, the features of raw signals are sometimes hard to be discovered in the time domain. These types of data should be filtered or converted before further processing. The primary purpose of Data Preprocessing is to prepare raw data for subsequent training. AFDM transforms the raw data into another domain depending on the data type. For instance, FFT is a popular preprocessing method that transforms signals (raw data) into the frequency domain and quickly extracts and analyzes the signals' features.

When realizing AFDM, we can install plenty of data preprocessing methods as a plug-in, like slicing the raw signals into pieces (Untransformed), FFT, and DWT. After preprocessing, AFDM splits the processed data into three sets: training, validation, and testing sets. Namely, data in the training set trains the classification models, data in the validation set tunes the parameters of the classification models, and the testing data evaluates the performance of the classification models installed in AFDM. We design two flags (F_e and F_v) to control the processing of training, validating, and testing. Once F_e is ON, AFDM forwards data to Model Evaluation in Phase II; otherwise, AFDM forwards the data to Model Training. Once F_v is ON, data is used to validate the trained models in Model Training; otherwise, data is used to train the classification models listed in Model Training.

Different classification models function differently. Some classification models are suitable for nonlinear data, while others are more effective when dealing with time-series data. To make AFDM analyze different data types, we install three variants of artificial neural networks in AFDM as the default classification models, including MLP, Recurrent Neural

Network (RNN), and Autoencoder (AE) with Softmax function. For simplicity, we define 1-MLP and 2-MLP for MLP with one and two hidden layers, respectively. MLP is a class of feedforward neural networks. In addition to the input and output layers, MLP contains some hidden layers, and neurons in two adjacent layers are interconnected.

RNN leverages sequential information of the input data from the previous step and feeds it as input to the next step, which is beneficial to recognizing patterns of time series data like text and speech recognition. AE learns a good representation of input data and is suitable for dimension reduction. AE extracts features from the input data and generates the reduced representations that can reconstruct the original data. An AE model contains an encoder to explore features and a decoder to reconstruct input data. By running with a Softmax function at the output of the encoder, an AE model can perform data classification.

The upper rectangle in Figure 1 shows the implementation of the Training phase. As illustrated in the figure, the default methods for Data Preprocessing are 'untransformed,' 'FFT,' and 'DWT,' where the 'Untransformed' means no preprocessing is required. Data will be forwarded to the next phase as it is. The default classification models for Model Training in Phase I include 1-MLP, 2-MLP, RNN, and AE (Softmax). An administrator can extend the preprocessing methods and classification models listed in Phase I as needed.

3.3 Phase II: Evaluation

AFDM mainly targets ranking and recommending the best-fit classification model to an administrator to analyze and predict faults of machine tools. Based on the results of analyzing the raw signals collected from the machine tools in the factory, AFDM makes recommendations to the administrator. Thus, AFDM needs to be able to handle different types of signals provided by different types of machine tools. For this purpose, AFDM has to evaluate different classification models' performance (e.g., prediction accuracy) and find the best-fit model for the specified machine tool(s).

Then, AFDM evaluates the classification models trained in the previous phase. The Evaluation phase contains two significant operations: Data Preprocessing and Model Evaluation, as illustrated in Figure 1. The Data Preprocessing operation in the Training and Evaluation phases are the same. Signals are forwarded to Model Evaluation as it is when 'Untransformed' is selected. Signals are processed and forwarded to the next phase when 'FFT,' 'DWT,' or other data preprocessing methods are selected. Compared with other research, the Model Evaluation operation works similarly to the model testing operation in other research. After testing the classification models trained in Phase I with the preprocessed data, AFDM calculates the performance for those trained models in terms of different metrics, including accuracy (Acc), precision (Pre), recall (Rec), f1-score (F1), training time ($Time_{tr}$), and testing time ($Time_{tst}$) (Ali et al., 2017; Mehdiyev et al., 2016).

The first four metrics, Acc, Pre, Rec, and F1, are defined by the confusion matrix for a two-class classification problem. The training time $Time_{tr}$ is the computation time required for training and tuning a classification model. The testing time $Time_{tst}$ is the computation time required for making a single prediction. AFDM uses these metrics to evaluate and rank the candidates of classification models trained in Phase I.

3.4 Phase III: Selection

According to the evaluation results obtained in Phase II, AFDM can rank the classification models trained in Phase I. The Selection phase defines two operations: Model Ranking and Model Selection. Model Ranking ranks the classification models by the metrics defined in the Evaluation phase and the preferences specified by a factory administrator. Since AFDM ranks models with multiple metrics, Phase III deals with an MCDM problem, so we cannot simply apply a sorting algorithm to rank these models. Some algorithms, like Analytic Hierarchy Process (AHP), Adjusted Ratio of Ratios (ARR), and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS). This research adopts TOPSIS in AFDM to solve such an MCDM problem. Conceptually, TOPSIS selects a positive ideal (best) solution and a negative ideal (worst) solution for each criterion (metric) and then ranks each candidate solution with its Relative Closeness (RC). The definition of RC is:

$$RC = \frac{S^*}{S^* + S^-}, 0 \leq RC \leq 1. \quad (1)$$

The equation defines a ratio of the distance of the candidate to the positive ideal solution (S^*) and the distance to the negative ideal solution (S^-). A higher RC represents a better solution, which should have a higher ranking. With TOPSIS, AFDM can rank the classification models and generate an ordered list of models (ranking). After obtaining the list, the administrators can select the best-fit classification model for their factory according to the ranking, experience, or other considerations. The rectangle of Phase III in

Figure 1 shows the processes of the Selection phase and its implementation.

3.5 Phase IV: Fault Diagnosis

The primary purpose of the previous three phases is to train and determine the best-fit classification model according to the historical data collected from the machine tools. The operations are time-consuming, so that we can process the operations in an offline manner. The fourth phase, the Diagnosis phase, is a process for diagnosing data collected from machine tools in real-time. The diagnosis should proceed immediately.

The Diagnosis phase contains two operations: Data Preprocessing and Fault Diagnosis. Similar to the Data Preprocessing in Phase I and II, the Data Preprocessing in Phase IV transforms the raw signals into a different type of data so that AFDM can extract features for analysis more efficiently. The only difference is that Data Preprocessing in the Diagnosis phase contains only one preprocessing method according to the classification model selected in Phase III. After preprocessing, we analyze the raw signals and the extracted features by the selected model. The analyzed results (diagnosis results) present the current status of the target machine tool. The administrator can monitor the target machine tools through diagnosis results. The Phase IV rectangle in Figure 1 shows the processes and implementation of the Diagnosis phase.

4 EXPERIMENTS

This section conducts four experiments to investigate the functionality of the main building blocks designed in AFDM. The four experiments include one diversity test, two feasibility tests, and one stability test. The diversity test shows AFDM's ability to handle raw signals collected from different types of machine tools (e.g., bearing, hydraulic pump, and drill bit). Then, we use the feasibility tests to show the feasibility of each phase in AFDM. In the first feasibility test, we investigate the impact of different data preprocessing methods with the same classification model. We evaluate and rank multiple classification models using different configurations in the second feasibility test. Finally, in the stability test, we investigate the stability of the ranking method (TOPSIS) adopted in AFDM. We evaluate the ranking results of AFDM by deploying various weights of the selected performance metrics.

4.1 Diversity Test

As mentioned in section 3.3, one of the significant objectives of AFDM is to recommend the best-fit classification model according to the characteristics of the input data. With so, AFDM can provide a flexible fault diagnosis mechanism for various machine tools. To show AFDM's diversity, we deploy different types of datasets to AFDM. In this experiment, we use datasets from different institutions with different kinds of machine tools, including one from Case Western Reserve University (CWRU) with bearing (Bearing 1), one from the University of Cincinnati with bearing (Bearing 2), one from Beihang University with hydraulic pump (Pump), and two from Indian Institute of Technology with drill bit (Drill 1 and Drill 2).

We train these datasets with 2-MLP models and list their performance metrics in Table 1. The accuracy of Bearing 1, Bearing 2, and Pump exceeds 0.8. The accuracy for Drill 1 and Drill 2 falls below 0.5. The results imply that we cannot apply a single algorithm to analyze raw signals collected from different types of machine tools. The results conclude that we need multiple classification models for analyzing data of different types. A fault diagnosis mechanism needs to select a dedicated model according to the characteristics of data (raw signal) collected from a machine under-diagnosis. If a factory administrator has no idea which model should be selected, he or she may need a recommendation mechanism.

Moreover, the results show the diversity of AFDM to handle various types of data from different machine tools. For simplicity, we use the CWRU dataset in the subsequent experiments.

4.2 Feasibility Test I: Different Data Preprocessing Methods

In this experiment, we evaluate the influences of different data preprocessing methods based on the performance of a classification model. In Phase I, we choose 2-MLP as the classification model, and each hidden layer contains 100 neurons. Then, we train the classification models with the same dataset, split ratio, and hyper-parameter but using different data preprocessing methods. We split the dataset into two subsets, 70% for training and 30% for testing. The training data contains the validation data.

We design six cases with different preprocessing methods in the experiment, including one with FFT, four with DWT, and one with untransformed. For DWT, we adopt four different configurations, including level 1 to 3 detail coefficients and approxima-

Table 1: Results of Diversity Test.

	Acc	**Pre**	**Rec**	**F1**	**Time**$_{tr}$	**Time**$_{tst}$
Bearing 1	0.8389	0.8444	0.8377	0.8410	1.2319	0.00058
Bearing 2	0.8581	0.8624	0.8566	0.8595	0.8409	0.00045
Pump	0.8226	0.8569	0.8284	0.8424	0.4707	0.00049
Drill 1	0.3885	0.3751	0.3861	0.3805	0.7361	0.00054
Drill 2	0.4889	0.4883	0.4887	0.4885	5.0848	0.00058

tion coefficients (abbreviated as DWT-L1, DWT-L2, DWT-L3, and DWT-approx). We use the untransformed data as the baseline for these cases. Each case repeats ten times with different random seeds, and then we calculate the average of each performance metric.

Table 2 shows the prediction results obtained when applying different data preprocessing methods to the classification model. The accuracy falls between 0.8394 and 1, precision falls between 0.8571 and 1, recall falls between 0.8394 and 1, and F1-score falls between 0.8481 and 1. When applying FFT to the raw signals, the accuracy is higher than in any other cases using DWT. Compared with the raw signals (untransformed), most cases activating data preprocessing have better prediction accuracy, except the one using DWT-L3. The results show that activating data preprocessing methods may extract essential features from the raw signals and help train the classification model. As for the computation time, four DWT cases require less time than the untransformed case. When applying DWT, both training time and testing time tend to increase as the dimension of data increases. This experiment requires more time to train the model when using FFT.

Since data preprocessing may affect the prediction accuracy of a classification model, AFDM provides the flexibility to bundle a preprocessing method and a classification model as a pair for ranking. Additional metrics, such as the preprocessing time, may be required when ranking such pairs. Hence, AFDM also provides the flexibility for adding these additional metrics in Phase II.

4.3 Feasibility Test II: Different Configurations and Parameters

While model training, we can apply different configurations and parameters to an ML algorithm and generate different classification models for better performance. This experiment considers four ML algorithms: 2-MLP, 1-MLP, RNN, and AE with Softmax function. Each algorithm has a different configuration (different numbers of hidden layers and neurons). 1-MLP and RNN have only one hidden layer, 2-MLP has two hidden layers, and AE has four hidden layers. We mark the configuration on the superscript of each algorithm. For example, we mark a 2-MLP algorithm running with two hidden layers, in which each layer has 100 neurons, as 2-MLP$^{(100,100)}$, as illustrated in Table 3.

Same as the previous experiment, the experiment analyzes the CWRU dataset (bearing). The percentage of training data to all data is 70%, and validation data is included in the training data. In this experiment, we do not activate any data preprocessing method. Data is untransformed. The experiment repeats ten times with different random seeds. Then, we calculate the average of each performance metric. AFDM ranks the twelve classification models and recommends one of them as the best-fit model. The experiment investigates the impact of different configurations.

Table 3 shows the performance metrics of the twelve models. The results show that more neurons lead to better accuracy, precision, recall, and F1-score. Generally, more time is required to train and test a model when using more neurons. Nevertheless, there might be exceptions in some cases. Even if fewer neurons are used, a model can still get better prediction results and a shorter training time. For example, in 2-MLP$^{(100,100)}$, the model has better performance than 2-MLP$^{(10,10)}$ and 2-MLP$^{(1000,1000)}$. 2-MLP$^{(100,100)}$ has the best performance both in the prediction accuracy and computational time.

Compared to the models using 1-MLP, 2-MLP, and RNN, RNN has better prediction accuracy than 1-MLP and 2-MLP when these models use the same number of neurons of the hidden layers. Undoubtedly, more time is required to train the RNN model and make a prediction. As for AE with the Softmax function, the results show that AE$^{(1000,200,40,5)}$, AE$^{(1500,300,60,8)}$, and AE$^{(2000,400,80,10)}$ obtain similar prediction accuracy, but the training and testing times increase as the network grows.

Since six performance metrics are used in AFDM to rank the twelve models, factory administrators need to decide the relative weights of the metrics to get the best-fit model for the factory. In this experiment, we give equal weights to the six metrics. AFDM uses

Table 2: Results of Feasibility Test I.

	Acc	**Pre**	**Rec**	**F1**	**Time**$_{tr}$	**Time**$_{tst}$
FFT	1	1	1	1	3.5418	0.00039
DWT-L1	0.8880	0.9054	0.8902	0.8977	0.9355	0.00050
DWT-L2	0.8704	0.8965	0.8689	0.8824	0.7040	0.00046
DWT-L3	0.8394	0.8571	0.8394	0.8481	0.6830	0.00045
DWT-approx	0.8610	0.8748	0.8577	0.8662	0.6948	0.00044
Untransformed	0.8477	0.8568	0.8490	0.8529	1.2729	0.00055

Table 3: Results of Feasibility Test II.

	Acc	**Pre**	**Rec**	**F1**	**Time**$_{tr}$	**Time**$_{tst}$	**RC**	**Rank**
2-MLP$^{(10,10)}$	0.7343	0.7436	0.7388	0.7412	1.3121	0.00057	0.88080	4
2-MLP$^{(100,100)}$	0.8408	0.8480	0.8440	0.8460	1.3057	0.00057	0.95922	1
2-MLP$^{(1000,1000)}$	0.7939	0.7984	0.7937	0.7961	2.5651	0.00071	0.87845	5
1-MLP$^{(10)}$	0.7040	0.7107	0.7079	0.7093	1.3615	0.00051	0.86176	6
1-MLP$^{(100)}$	0.8197	0.8230	0.8214	0.8222	1.3756	0.00054	0.94452	2
1-MLP$^{(1000)}$	0.8328	0.8275	0.8332	0.8303	2.5778	0.00062	0.91150	3
RNN$^{(10)}$	0.7357	0.7373	0.7338	0.7356	7.3714	0.00112	0.61100	8
RNN$^{(100)}$	0.8481	0.8650	0.8470	0.8559	3.1549	0.00115	0.76070	7
RNN$^{(1000)}$	0.8829	0.8890	0.8824	0.8857	8.6092	0.00255	0.30280	12
AE$^{(1000,200,40,5)}$	0.8767	0.8776	0.8715	0.8745	10.0747	0.00075	0.60286	9
AE$^{(1500,300,60,8)}$	0.8702	0.8708	0.8706	0.8707	12.1400	0.00084	0.52780	10
AE$^{(2000,400,80,10)}$	0.8757	0.8772	0.8764	0.8768	14.7346	0.00088	0.46457	11

TOPSIS for ranking the models by the six metrics. The RC value of each model is calculated and listed in the second last column of Table 3.

In the table, we can see that 2-MLP$^{(100,100)}$ has the highest RC (0.95922), and 1-MLP$^{(100)}$ owns the second-high RC value (0.94452). The RC values of the two models are very close. Whenever there is any vibration of evaluation results, the rank of the models may change. The administrator can substitute the working model with the recommended one. The substitution between the models may cost some overheads. The overheads could be considerable since the structures (e.g., the number of hidden layers and the number of neurons in each hidden layer) vary significantly between ML algorithms. Thus, we design the fourth experiment, the Stability Test, for further discussions about the stability of model ranking.

4.4 Stability Test: Stable Model Ranking

AFDM recommends the best-fit classification model for fault diagnosis by ranking the candidates with the performance metrics. The variation of ranking may cause model substitution. The substitution can be as small as modifying hyper-parameters only or as big as changing the structure of the classification model. If the ranking of models varies from time to time, the substitution overhead could be considerable and influence the overall performance. In this experiment, we evaluate the stability of TOPSIS's rankings when the weights of performance metrics change.

To observe the weights and rankings, we only consider the relative changes in weights between two performance metrics: accuracy and training time. We define two weights for the two metrics, the weight of accuracy (w_a) and the weight of training time (w_t). The summation of the two weights is 1. We choose the top-six cases in Table 3 and evaluate the changes in their rankings for the varied weights. w_a varies from 1 to 0 and w_t from 0 to 1. The combination of the weights are recorded as w_a/w_t (e.g., 0.6/0.4.) Figure 2 shows the change in rankings for the selected models.

In TOPSIS, the ranking is generated based on the RC value of each candidate in descending order. Thus, a larger RC stands for a higher ranking. In Figure 2, we can see that 2-MLP$^{(100,100)}$ has the highest RC value ($RC = 1$) among all cases. AFDM ranks the 2-layer MLP with 100 neurons in the hidden layer as the best solution within all the combinations of weights. 1-MLP$^{(100)}$ (the line marked with $\star$) is the second candidate recommended for the cases using the weight combinations from 0.9/0.1

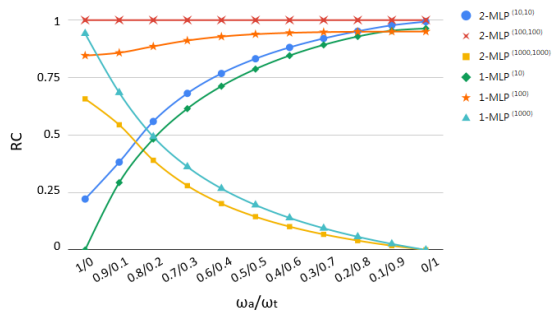

Figure 2: RC values under different combination of weights.

to 0.3/0.7. In short, the rankings of 2-MLP$^{(100,100)}$ and 1-MLP$^{(100)}$ remain unchanged under the weight combinations within the range of 0.9/0.1 to 0.3/0.7.

As for other models, the rankings vary as the relative weights change. For example, when w_t is considered much more important than w_a, 2-MLP$^{(10,10)}$ and 1-MLP$^{(10)}$ are recommended; otherwise, AFDM recommends 2-MLP$^{(1000,1000)}$ and 1-MLP$^{(1000)}$. The results show that AFDM can consider administrators' preferences while keeping the generated rankings stable to a certain extent. This experiment proves that adopting TOPSIS as the selection method in AFDM can obtain feasible, adaptable, and stable results.

5 DISCUSSION

We compare AFDM with Sun's, Brecher's, and Thirukovalluru's work and summarize their differences in Table 4. As shown in Table 4, the comparison contains three different aspects: target, training, and evaluation. The target aspect indicates whether a candidate supports fault diagnosis targeting multiple types of machine tools. The training aspect shows the ability to support different data preprocessing methods and ML models for fault diagnosis. The evaluation aspect discloses the metrics emphasized during model evaluation.

Sun's work used a bearing dataset from CWRU as the input data and identified the fault conditions of bearings. In Brecher's work, a packing machine was considered, which monitored the health conditions of the belt. Although Brecher et al. mentioned the possibility of supporting multiple machines with cloud computing technology, the detail about related design was lacking. Thus we marked this feature as △. Among the related work, only Thirukovalluru's work investigated the diagnosis for different machine tools, including an air compressor, drill bit, bearing, and steel plate. In this paper, we design and implement AFDM to diagnose faults of bearing, drill bit, and pump, but the framework of AFDM is also flexible in analyzing faults of different machine tools.

In Sun's and Thirukovalluru's works, the researchers adopted one primary ML algorithm to improve the model training process for the target machine tool. Differently, Sun et al. directly deployed SSAE as the classification model, and Thirukovalluru et al. applied DNN to improve the classification models through feature extraction. None of them mentioned how to customize the data preprocessing methods and classification models such that an administrator can analyze the data of their machine tools more precisely. In Brecher's work, the authors applied many ML algorithms to diagnose the belt faults with different data features. Then, the authors selected the model with the best accuracy for their packing machine. Comparatively, AFDM provides a flexible framework and allows an administrator to install user-defined data preprocessing methods and classification models. An administrator can specify the weights of performance metrics to find the best-fit model(s) for their machine tools. Such a design makes AFDM adaptable to different scenarios and users' preferences.

Most related works selected a classification model based on accuracy. Sun's work investigated each classification model's classification accuracy and computation time. Although they considered the trade-off between accuracy and computation time, they did not explain how to solve it when making the final selection. Also, they did not specify what kind of computation time they used for model evaluation, so we marked both the training and testing time as △. In Brecher's work, the authors investigated the advantages and disadvantages of each classification model but did not specify how they selected the model based on these advantages and disadvantages. Thus we mark their support of other criteria as △.

With AFDM, an administrator can select the best-fit classification model based on the recommended rankings generated by TOPSIS. The administrator only needs to determine the relative weight of each performance metric according to their preferences and experiences, and then AFDM can automatically generate rankings of the classification models. In addition to the performance metrics used in the experiments, the administrator can add more quantitative criteria for evaluating classification models, showing the flexibility of AFDM in model selection.

Table 4: Comparison among fault diagnosis mechanisms.

Supporting Features		Sun's	Brecher's	Thirukovalluru's	AFDM
Target	Multiple Machine Tools	✗	△	✓	✓*
Training	Multiple Preprocessing Methods	✗	✗	✗	✓*
	Multiple ML Models	✗	✓	✓	✓*
Evaluation	Accuracy	✓	✓	✓	✓
	Training Time	△	✗	✗	✓
	Testing Time	△	✗	✗	✓
	Others	✗	△	✗	✓*

* support customized options

6 CONCLUSION

Nowadays, rapidly developing ML technology and related applications are introduced to manufacturing to make it "smarter." Fault diagnosis of machine tools, for example, traditionally depended on the experience owned by the administrators. However, by deploying ML technology, the faults of running machine tools can be detected or even predicted immediately. This paper proposes AFDM, a generic fault diagnosis mechanism for different machine tools. AFDM, operating in four phases, can automatically recommend the best-fit model according to multiple metrics, including the nature of input data and user preferences. We conduct four experiments to show AFDM's diversity in handling various data from different machine tools, the feasibility of configuring different methods and parameters in each phase, and the stability in ranking and recommending the best-fit classification model. In comparison to existing works, AFDM is the only approach that can:

1. adapt to various data from different kinds of machine tools,
2. support multiple data preprocessing methods and ML models, and
3. stably evaluate and rank the candidate models with multiple criteria, where the weight of each criterion is configurable.

AFDM leaves flexibility for administrators to add or select data preprocessing methods, ML algorithms, and metrics to train and evaluate the models according to the user's experience. We conclude that AFDM can stably and automatically recommend the best-fit ML model for the fault diagnosis of machine tools based on user's preferences.

ACKNOWLEDGEMENTS

This work was financially supported in part by Ministry of Science and Technology, Taiwan, under grant numbers MOST106-2218-E009-008 and MOST107-2218-E009-059.

REFERENCES

Ali, R., Lee, S., and Chung, T. C. (2017). Accurate multi-criteria decision making methodology for recommending machine learning algorithm. *Expert Systems with Applications*, 71:257–278.

Brecher, C., Obdenbusch, M., and Buchsbaum, M. (2017). Optimized state estimation by application of machine learning. *Production Engineering*, 11(2):133–143.

Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306.

Jia, F., Lei, Y., Lin, J., Zhou, X., and Lu, N. (2016). Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, 72:303–315.

Kumar, U. and Galar, D. (2018). Maintenance in the era of industry 4.0: issues and challenges. *Quality, IT and business operations*, pages 231–250.

Leukel, J., González, J., and Riekert, M. (2021). Adoption of machine learning technology for failure prediction in industrial maintenance: A systematic review. *Journal of Manufacturing Systems*, 61:87–96.

Mehdiyev, N., Enke, D., Fettke, P., and Loos, P. (2016). Evaluating forecasting methods by considering different accuracy measures. *Procedia Computer Science*, 95:264–271.

Sun, J., Yan, C., and Wen, J. (2017). Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning. *IEEE Transactions on Instrumentation and Measurement*, 67(1):185–195.

Thirukovalluru, R., Dixit, S., Sevakula, R. K., Verma, N. K., and Salour, A. (2016). Generating feature sets for fault diagnosis using denoising stacked auto-encoder. *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–7.

Wen, L., Li, X., Gao, L., and Zhang, Y. (2017). A new convolutional neural network-based data-driven fault diagnosis method. *IEEE Transactions on Industrial Electronics*, 65(7):5990–5998.