IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 August 2024, accepted 26 August 2024, date of publication 30 August 2024, date of current version 10 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3452188

RESEARCH ARTICLE

Two-Stage Fault Classification Algorithm for Real Fault Data in Transmission Lines

SE-HEON LIM, (Graduate Student Member, IEEE), TAEGEUN KIM, (Graduate Student Member, IEEE), KYEONG-YEONG LEE, (Graduate Student Member, IEEE), KYUNG-MIN SONG, AND SUNG-GUK YOON, (Senior Member, IEEE)
Department of Electrical Engineering, Soongsil University, Seoul 06978, South Korea

Corresponding author: Sung-Guk Yoon (sgyoon@ssu.ac.kr)

This work was supported in part by Korea Electric Power Corporation under Grant R22XO02-19; and in part by Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry and Energy (MOTIE) of the Republic of Korea, under Grant RS-2024-00398166.

ABSTRACT Fault classification in power transmission lines is important in distance relaying for identifying the accurate phases implicated in the fault occurrence. Generally, the accuracy of fault classification algorithms is evaluated by simulation data, which shows quite different characteristics from real fault data. Also, most of the previous works on fault classification used a single-stage method such as a rule-based algorithm or machine learning-based algorithm. Because of the diverse characteristics of real fault data, the performance of the single-stage method is limited. To address these issues, this paper proposes a novel two-stage algorithm that combines the strengths of rule-based and machine-learning algorithms to improve the accuracy of real fault data. A case study using real fault data shows that the proposed two-stage algorithm outperforms other conventional single-stage algorithms.

INDEX TERMS Two-stage algorithm, rule-based, artificial neural network, root mean square.

ABBREVIATIONS

- **IV**: Instantaneous Value
- **WT**: Wavelet Transform
- **DWT**: Discrete Wavelet Transform
- **CWT**: Continuous Wavelet Transform
- **MODWT-E**: Maximal Overlap Discrete Wavelet Transform Energy
- **MODWT**: Maximal Overlap Discrete Wavelet Transform
- **ITD**: Intrinsic Time Decomposition
- **DT**: Decision Tree
- **KNN**: K-Nearest Neighbor
- **RMS**: Root Mean Square
- **FRC**: Frequency Response Curve

I. INTRODUCTION

Accurate classification of fault types is necessary to restore power systems after a fault because different recovery methods and actions are required depending on the fault type. In addition, identifying fault types is essential for the relay decision-making process in transmission networks, which is pivotal for applications such as single pole tripping, distance relaying, etc [1]. The accurate classification enables fast tripping, which reduces further damage to the system, enhances system reliability, and improves transient system stability and power quality [2]. In the three-phase system of transmission lines, 11 different types of line faults can occur: one type of three-phase fault (LLL), one type of three-phase to ground fault (LLLG), three types of single line-to-ground fault (LG), three types of double line to ground fault (LLG), and three types of line to line fault (LL) [3], [4].

A. LITERATURE REVIEW

Previous research on fault type classification is generally divided through three perspectives: which ‘‘algorithm’’ is applied, which ‘‘data’’ is used for validation, and which ‘‘features’’ are used. From these points of view, a literature review is listed in Table 1.

The associate editor coordinating the review of this manuscript and approving it for publication was Guillermo Valencia-Palomo.

From the ''algorithm'' point of view, there are two main algorithms for fault classification: rule-based algorithms, i.e., traditional approach, and machine learning (ML)-based algorithms. ML-based algorithms can also be categorized into tree-based algorithms and neural network (NN)-based algorithms. Recently, many works on fault classification using NN-based algorithms including artificial neural networks (ANN), convolutional neural networks (CNN), and recurrent neural networks (RNN) have been actively investigated. The two types of algorithms, i.e., rule-based and ML-based algorithms, have different characteristics such as strengths and limitations. ML-based algorithms classify a fault type based on features that experts may not have considered because they are trained from datasets without relying on expert knowledge. In other words, they are more flexible compared to rule-based methods. However, one of the major drawbacks of ML-based algorithms is overfitting which comes from the disparity between the training and test data. Especially, an ML-based algorithm trained using simulation data might show a poor classification performance in real fault data if the real fault data is not seen in the simulation [22]. This problem still exists because most of the previous research has used simulation data as training data. Rule-based algorithms, on the other hand, are relatively independent of the quality of the training data because they rely on the knowledge of experts. However, expert knowledge alone sometimes makes it difficult to recognize subtle cases that are close to the threshold.

From the ''data'' perspective, most models trained using simulation data have limitations on the performance of real fault data because simulation cannot perfectly reproduce the real environment. Specifically, both rule-based and ML-based algorithms perform poorly on real fault data. Previous research, that validated using real fault data [5], [14], [15], [22], either showed performance degradation or limitations in the reliability of the validation dataset. For example, the works in [5] and [22] showed 100% accuracy with simulation data, but they showed 96.55% and 96% accuracy with real fault data, respectively. In particular, [22] compared fault classification models to previous studies and showed that an average classification performance degrades about 33% with real fault data than fault data obtained by simulations. In [14] and [15], their classification models achieved 100% accuracy with real fault data. However, the total number of real fault data is very small (five and six) and their fault category is simple (LLG and LG).

As the input ''feature'' point of view, most of the previous research used preprocessed features rather than instantaneous values (IV), and wavelet transform (WT)-based decomposition techniques are the mainstream. So, various preprocessing techniques such as DWT, MODWT, CWT, and RMS have been used in previous research. However, most performance evaluations have been conducted using simulation data, making it uncertain how these features will perform in real fault data. Previous research showed that WT-based features are generally not robust to noise [24], [25], [26]. Therefore, to improve the classification accuracy in real fault data, a detailed analysis of input features using real fault data is required.

B. CONTRIBUTIONS

In this work, we aim to develop a robust fault classification algorithm for real fault data. To this end, we propose a two-stage algorithm that combines rule-based and ML-based ones. First, the proposed algorithm uses a rule-based method for initial fault type classification. If the fault data lies near the borderline for classification, the ML-based algorithm is employed. Otherwise, the classification result by the rule-based algorithm is the final result. By combining the strengths of the two algorithms, the proposed two-stage algorithm outperforms individual algorithms. Through a case study, we show that the proposed algorithm successfully classifies 100% of the real fault data even when it is trained by simulation data.

The main contributions of this paper are summarized as follows:

- Two-Stage Classification Algorithm: The paper introduces a novel two-stage algorithm for fault classification that synergistically combines rule-based and ML-based algorithms. The rule-based and ML-based algorithms are used for initial classification and final decisions for the data in boundary conditions, respectively.
- Feature Analysis: This work gives a detailed analysis of different features such as IV, RMS, MODWT, and CWT, used in fault classification algorithms. Among them, RMS is a robust feature, so it enhances the accuracy and reliability of the classification.
- Performance Evaluation by ML-based algorithms: This paper categorizes ML-based algorithms into tree-based and NN-based methods and compares their performance. Case study shows that NN-based algorithms perform better than tree-based algorithms.
- Validation using different datasets: We validate the performance of the proposed algorithm using simulation data, simulation data with noise, and real fault data. Through this validation, it is confirmed that the proposed algorithm can be practically applied to real fault data.

The paper is organized as follows: Section II outlines the background on ML algorithms and features utilized in this work. Section III introduces the RMS-based rule-based algorithms and then proposes the two-stage algorithm for fault classification. Section IV presents a case study to validate the proposed algorithm including feature extraction, parameter selection for the rule-based algorithm, and the performance of the proposed algorithms. The paper concludes with Section V.

II. BACKGROUND

A. REVIEW OF ML ALGORITHMS

This section reviews two types of ML algorithms used for fault classification in this work: tree-based and NN-based algorithms.

TABLE 1. Literature review of fault type classification models.

Ref.	Algorithm Type	Detailed Algorithm	Feature	Real data for test
[5]	Rule-based	Binary logic	MODWT-E	✔
[6]		Fuzzy logic	Coefficients of WT	✘
[7]		Fuzzy logic	Coefficients of WT	✘
[9]	Tree-based ML	DT	Odd harmonics	✘
[10]		DT, KNN	Coefficients of DWT	✘
[11]		CatBoost	Coefficients of IV	✘
[12]	NN-based ML	ANN	Coefficients of CWT	✘
[13]		ANN	Coefficients of BF	✘
[14]		ANN	Coefficients of ITD	✔
[15]		ANN	Coefficients of DWT	✔
[16], [17]		CNN	IV	✘
[18], [19]		CNN	Coefficients of CWT	✘
[20]		RNN	RMS	✘
[21]		RNN	IV	✘
[22]		RNN & ANN	IV, RMS	✔
[23]		RNN & CNN	FRC	✘
Proposed	Rule-based & NN-based ML	Threshold based & ANN	RMS	✔

1) TREE-BASED ALGORITHMS

The core concept behind tree-based algorithms is to construct a decision tree using training data. Representative tree-based models include decision trees, random forests, and gradient-boosting trees. The decision tree, i.e., the basic element of the tree-based algorithms, is built by splitting features to maximize information gain. The formula for information gain IG is as follows:

$$\mathrm{IG}(D_p, fe) = I(D_p) - \left(\frac{N_l}{N_p} I(D_l) + \frac{N_r}{N_p} I(D_r)\right), \quad (1)$$

where fe refers to the feature upon which the data is split at any node in the tree. And D_p, D_l and D_r denote the datasets of the parent node, and the left and right child nodes after the split, respectively. Also, N_p, N_l, and N_r are the number of samples in the parent node, and the left and right child nodes, respectively. The function $I(\cdot)$ measures the impurity of a dataset, and the impurity is normally measured by either entropy $I_H(\cdot)$ or Gini impurity $I_G(\cdot)$ [27]. Entropy as impurity is defined as

$$I_H(D) = -\sum_{i \in D} p_i \log_2(p_i), \quad (2)$$

where p_i represents the proportion of class i within the dataset. On the other hand, Gini impurity is

$$I_G(D) = 1 - \sum_{i \in D} p_i^2. \quad (3)$$

Because of the criteria to maximize information gain, decision trees effectively understand the features of the data and create predictive models. Random forests and gradient-boosting trees further reduce errors and enhance performance by using multiple trees.

2) NN-BASED ALGORITHMS

The neural network is a cornerstone of NN-based ML algorithms. It is designed to estimate the parameters of a function using training data. Neurons in layer i pass signals to the next layer of neurons from input vector $\boldsymbol{x}$ which is expressed as

$$f^{(i)}(\boldsymbol{x}) = \phi(\boldsymbol{w}^i \boldsymbol{x} + \boldsymbol{b}^i), \quad (4)$$

where $\boldsymbol{w}^i$ and $\boldsymbol{b}^i$ represents the weight and bias vectors of layer i, respectively. And $\phi(\cdot)$ is the activation function. The output y of the NN for input $\boldsymbol{x}$ is the output of the composition of the functions corresponding to each layer. That is,

$$y = (f^{(L)} \circ f^{(L-1)} \circ \cdots \circ f^{(1)})(\boldsymbol{x}), \quad (5)$$

where L is the number of layers and the circle symbol ($\circ$) denotes function composition.

For classification problems, the loss function uses the cross-entropy loss $L(\cdot)$ as given as

$$L(y, \hat{y}) = -\sum_{c=1}^{C} y_i \log(\hat{y}_c), \quad (6)$$

where y and $\hat{y}$ denote the true distribution and the estimated distribution, respectively. And y_i is 1 for the correct class and 0 otherwise, and $\hat{y}_c$ is the estimated probability for class c.

By the backpropagation, the weights are updated as follows

$$W := W - \eta \frac{\partial L}{\partial W}, \quad (7)$$

where W and η denote the weight matrix and the learning rate, respectively. The process of applying this gradient to update the weights is repeated across all layers from the output back to the input layer, adjusting the network's weights

in a way that minimizes the loss, hence improving the model's predictions.

B. CANDIDATE FEATURES FOR FAULT CLASSIFICATION

The accuracy of fault classification algorithms can vary based on the characteristics of preprocessed features derived from IV of fault data. Therefore, we focus on evaluating the accuracy of fault classification models using preprocessed features as inputs, such as RMS, DWT [10], [15] and CWT [12], [17], [18]. This section reviews three preprocessing features for fault classification: RMS, MODWT, and CWT.

1) MOVING WINDOW-BASED RMS

RMS is one of the most common features of voltage and current. In this paper, we propose the use of moving window-based RMS. The moving window-based RMS formula considering the window size and stride is given as

$$\mathrm{RMS}_i = \sqrt{\frac{1}{N}\sum_{j=0}^{N-1} x^2(i\cdot S + j)}, \tag{8}$$

where N and S denote the size of the window and stride,[1] respectively. Also, x is the data sample, and i indicates the starting index of the window. So, $i \cdot S$ stands for the index shift due to the stride.

2) MODWT

MODWT [5] is an extension of DWT, and it has strength in time series analysis. While DWT requires a length of the time series data to be a power of two, MODWT is not restricted by the length of the data and allows for overlapping in the wavelet filtering process. As a result, MODWT is good for analyzing non-stationary temporal signals where capturing transient features is essential. The k-th wavelet coefficient value of the MODWT $\omega_m(k)$ is calculated as

$$\omega_m(k) = \sum_{i=1}^{K} h(i)x(k+i-K), \tag{9}$$

where h and K are the wavelet filter and the amount of coefficient of the wavelet filter, respectively.

3) CWT

CWT is designed to decompose a signal into wavelets, which are essentially small waves located at different intervals. These wavelets vary in scale and position, allowing for a precise measurement of how the signal's frequency content changes over time. The wavelet coefficient value of the CWT $\omega_c(\cdot,\cdot)$ for a signal $x(t)$ is defined as

$$\omega_c(s,\tau) = \frac{1}{\sqrt{s}}\int_{-\infty}^{\infty} x(t)\cdot\psi^*\left(\frac{t-\tau}{s}\right)dt, \tag{10}$$

[1]The number of data points skipped each time the window is moved.

where s and τ are the scale factor that stretches or compresses the wavelet and the translation parameter that shifts the wavelet along the time axis, respectively. A function $\psi(\cdot)$ is the wavelet function used to generate all other wavelets by scaling and translating, and $\psi^*(\cdot)$ is its complex conjugate. In this work, we use morlet wavelet [28] as a wavelet function.

III. TWO-STAGE FAULT CLASSIFICATION ALGORITHM

This section introduces the proposed two-stage algorithm that integrates rule-based and ML-based algorithms to classify fault data. In Stage 1 (Section III-A), the proposed algorithm first applies a rule-based classification method based on the RMS ratio. After verifying the fault coordinate location, it applies the ML-based algorithm, i.e., Stage 2 (Section III-B). The proposed two-stage fault classification algorithm is explained in detail in Section III-C. Note that we use fault current data only because the current is the best indication of the fault characteristic.

A. STAGE 1: RULE-BASED ALGORITHM USING RMS RAITO

The first stage algorithm is based on a reference work [5]. This section reviews the algorithm in [5] and describes our contributions to this algorithm. Our contributions are i) changing the input feature from wavelet coefficient energies to RMS ratio, ii) optimizing the error threshold ϵ of the fault coordinate, and iii) creating an additional plane to categorize LLL and LLLG.

1) RMS RATIO

As the input of the rule-based algorithm, the RMS ratio of phase current is used instead of the wavelet coefficient energies in previous work [5] because of the robustness of RMS for noise [24]. In Section IV-B1, we analyze the robustness of RMS. RMS current is defined as

$$I_j^{RMS} = \sqrt{\frac{1}{T_e - T_s}\sum_{t=T_s}^{T_e} I_j^2(t)}, \quad j \in \{A, B, C, G\}, \tag{11}$$

where $I_j(t)$ is the IV of current at t and j is the phase index of three phases current (A, B, C) and neutral current (G). Given the non-periodic nature of the fault data, the interval from T_s to T_e is extracted for RMS calculation, rather than employing a single cycle typical of periodic functions.

After obtaining RMS current for each phase, we define the RMS ratio of A, B, C, and G as

$$r_j = \frac{I_j^{RMS}}{I_A^{RMS} + I_B^{RMS} + I_C^{RMS}}, \quad j \in \{A, B, C, G\}. \tag{12}$$

By definition, the sum of the RMS ratio for three phases should be 1, i.e., $r_A + r_B + r_C = 1$.

2) COORDINATES BASED FAULT CLASSIFICATION

The three-dimensional fault coordinate system comprises three orthogonal axes: A, B, and C. The coordinates A, B, and C represent each RMS ratio as defined by Eq. (12). On the

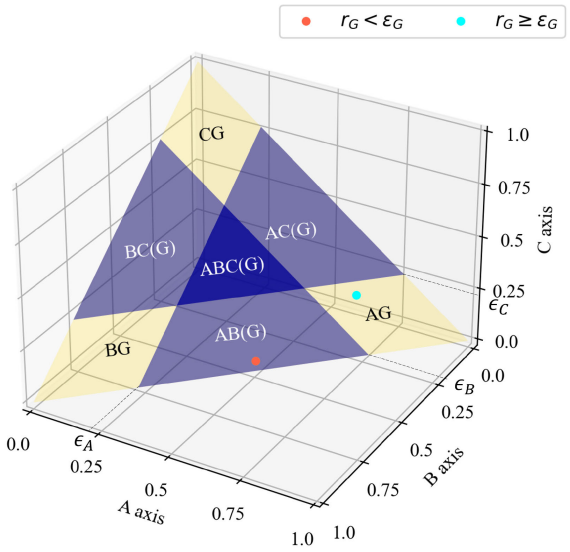


FIGURE 1. Three dimensional fault coordinate.

other hand, coordinate G denoted as r_G, is represented by colors based on the threshold ϵ_G. If $r_G < \epsilon_G$, the color of the dot in the coordinate is red. On the other hand, it is green. Because the sum of the RMS ratio for three phases should be 1, these fault coordinates lie on a three-dimensional plane known as the fault triangle, illustrated in Fig. 1. The placement of A, B, and C on the fault plane allows for the identification of the phases impacted by the fault, while the color G indicates ground involvement in the fault.

As shown in Fig. 1, fault planes which are represented as fault type are separated based on threshold ϵ_j. Therefore, if the RMS ratio exceeds a threshold value of ϵ_j, phase j is classified as a fault. In [5], ϵ_j for all $j \in \{A, B, C\}$ was naively set to $1/3$, which is a reliable value to distinguish between LG, LL, and LLG faults. However, LLL faults have to belong to all planes, which limits the accurate fault classification. Therefore, instead of setting $\epsilon_j = 1/3$, this work chooses the proper threshold value based on simulation data to create an additional plane to categorize LLL and LLLG faults. The detailed parameter settings based on the simulation dataset for the rule-based algorithm are described in Section IV-B. With the proper ϵ_j, the fault planes consist of 7 fault planes so that 11 faults can be classified, including ground fault cases represented by colors as shown in Fig. 1.

B. STAGE 2: ML-BASED ALGORITHM

In the second stage, the proposed algorithm uses an ML-based classification method. Two tree-based algorithms (XGBoost [29] and LGBM [30]) and two NN-based algorithms (ANN [31], CNN [32], and LSTM [33]) are considered in this work. Potential candidates for input features of these ML-based algorithms are IV, RMS, MODWT, and CWT. The classification result of the algorithms is one of 11 fault types (three LG, three LL, three LLG, one LLL, and one LLLG).

We also propose to use a voting method for ML-based algorithms. Voting is one of the ensemble learning techniques that combines several different models to determine the final prediction. The aggregation of predictions from multiple models tends to reduce the variance associated with individual predictions because the performance of ML-based algorithms highly depends on the initial value. This makes the ensemble's output more predictable and reliable. After training, the ML-based algorithm provides multiple classification results and aggregates the results, i.e., perform voting. In voting, the final result is decided based on the class that receives the most votes.

C. TWO-STAGE FAULT CLASSIFICATION ALGORITHM

Both rule-based and ML-based algorithms have advantages and disadvantages. The rule-based algorithm shows reasonable and explainable results, but it cannot distinguish subtle differences. On the other hand, ML-based algorithms have the advantage of distinguishing subtle differences, but they always have the potential for overfitting. Therefore, the proposed two-stage fault classification algorithm applies the rule-based algorithm first, and then the ML-based algorithm is applied if the fault coordinates are located in a plane near the threshold as shown in the red and green points of Fig 2. In Fig. 2, the conditions ($\left|r_j - \epsilon_j\right| = \Delta_j$) of ABC phases and neutral current are represented by the dotted line and color, respectively. The condition of "$\left|r_j - \epsilon_j\right| \leq \Delta_j$" is called a "boundary condition" in this work.

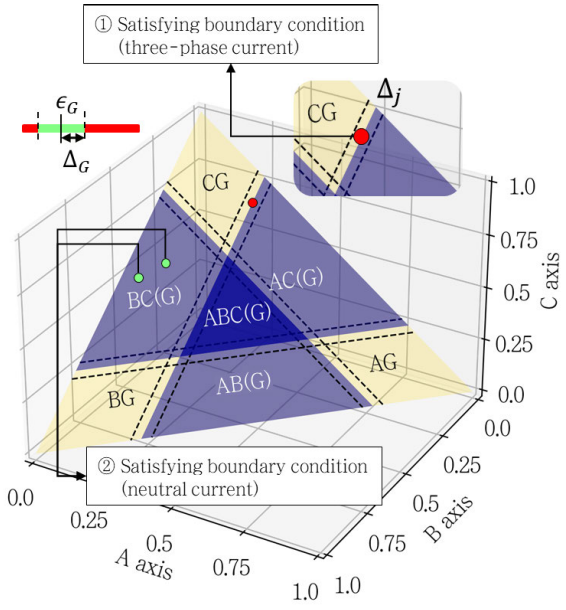


FIGURE 2. Condition for progressing to stage 2.

The detailed process of the proposed two-stage classification algorithm is outlined in Algorithm 1.

The operation of Algorithm 1 is iterative for each fault data. The algorithm is explained as follows: The rule-based algorithm runs first. (Line 2) calculate RMS ratio r_j ($j =$

Algorithm 1 Two-Stage Algorithm

1: Initialize fault type $F = \varnothing$
Stage 1: Rule-based algorithm
2: Calculate RMS ratio r_A, r_B, r_C, r_G by Eq. (12)
3: **for** $j \in \{A, B, C, G\}$ **do**
4: **if** $r_j \geq \epsilon_j$ **then**
5: $F \leftarrow F \cup \{j\}$
6: **end if**
7: **end for**
8: **if** for any $|r_j - \epsilon_j| \leq \Delta_j$ **then**
Stage 2: ML-based algorithm
9: Re-initialize fault type $F = \varnothing$
10: Initialize an empty list of predictions $P = \varnothing$
11: **for** $i \in \{1, 2, \ldots, M\}$ **do**
12: $y_i^{pred} \leftarrow$ Prediction from ML model i
13: $P \leftarrow P \cup \left\{ y_i^{pred} \right\}$
14: **end for**
15: $F \leftarrow$ Choose the most frequent case in P
16: **end if**
17: **return** F=0

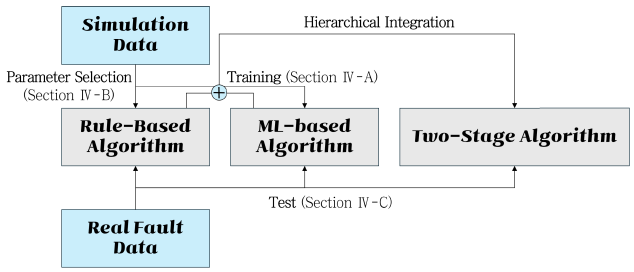


FIGURE 3. Schematic diagram of experiment.

A, B, C, G) by Eq. (12). (Lines 3-7) If any value of the RMS ratio r_j exceeds threshold ϵ_j, the phase is added to fault type set F. After checking the condition for all phases, the initial fault type is decided. (Line 8) At least one of the phases satisfies the boundary condition, (Lines 9-15) The ML-based algorithm is executed M times and the final classification result is decided by voting method. Otherwise, the initial fault type is the final classification result.

IV. CASE STUDY

In this section, we compare the classification performance of the proposed two-stage algorithm with single-stage algorithms and another two-stage algorithm. Fig 3 shows a schematic diagram of the experiment. Section IV-A describes the power system simulation and hyperparameter settings for ML algorithms. Section IV-B shows the process of setting up features and thresholds for a rule-based algorithm. Finally, in Section IV-C, the classification result for each algorithm is demonstrated.

The experiments were conducted on a PC equipped with a 13th Gen Intel(R) Core(TM) i9-13900KF processor running at 3.00 GHz, 64 GB of RAM, and an NVIDIA GeForce RTX-4090 GPU. The machine learning tasks were implemented using Python with scikit-learn and TensorFlow packages.

TABLE 2. The parameter of PSCAD simulation.

Fault type	LG (3), LL (3), LLG (3), LLL (1)
Length (km)	10 - 100 (10 km interval)
Angle (°)	0 - 180° (30° interval)

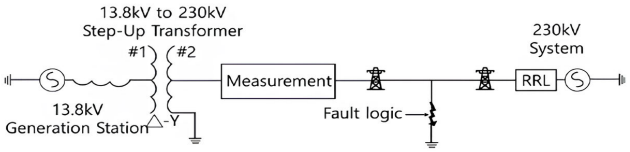


FIGURE 4. Power system model.

TABLE 3. Configuration of real fault data.

Fault type	Number
AG	5
BG	11
CG	9
AB	3
BC	7
ABG	3
ACG	3
ABC	11

A. EXPERIMENT SETUP

1) POWER SYSTEM SIMULATION

To simulate and generate line faults in transmission lines, we modeled the power system using the PSCAD program as shown in Fig. 4. Actual load and transformer parameters from the South Korean transmission lines were used to closely mimic real system faults where real fault data is stored [22]. We generated a total of 700 fault data, i.e., 7 (angles) × 10 (locations) × 10 (fault types[2]), by organizing fault types, transmission line lengths, and phase angles as shown in Table 2. The generated fault waveforms are 64 samples per cycle and 2 cycles after the fault. Then, the basic input of IV data is preprocessed for input to rule-based and ML-based algorithms. Among the generated 700 fault data, we use 90% of the data for training, while the remaining 10% data, along with 52 real field data samples, were used for test.

2) REAL FAULT DATA

We have 52 real fault data with fault types of LG, LL, LLG, and LLL, obtained by Korea Electric Power

[2]While LLLG fault can occur in the real power system, it is difficult to simulate LLLG fault with characteristics that distinguish them from LLL fault, especially in the simulation of transmission lines. We generated fault data except for LLLG type. Accordingly, the ML-based algorithm that learns with simulation data can classify into 10 types of faults, excluding LLLG type.

TABLE 4. List of features and their size.

Algorithm	Feature	Size	Size of CNN
ML	IV	512	4×128
	RMS	260	4×65
	MODWT	512	4×128
	CWT	1024	4×256
Rule-based	RMS ratio	4	-

TABLE 5. Hyperparameters of ML-based algorithm.

ML-based algorithm	Parameters	Value
XGBoost	Max depth of a tree	6
	Learning rate	0.3
LGBM	# of leaves	31
	# of trees for boosting	100
	Learning rate	0.1
ANN	# of hidden layers	2
	Size of hidden layers	32
	Activation function	ReLu
	Dropout rate	0.25
	Learning rate	0.001
CNN	# of convolutional layers	2
	Kernel shape	(1, 32)
	Max pooling shape	(1, 4)
	Number of dense layers	1
	Size of dense layers	32
	Activation function	ReLu
	Dropout rate	0.25
	Learning rate	0.001
LSTM	# of LSTM layers	2
	Size of LSTM layers	32
	# of Dense layers	1
	Size of Dense layers	32
	Activation function	ReLu
	Dropout rate	0.25
	Learning rate	0.001

Corporation (KEPCO). The real-time fault events were recorded during different fault types on various phases of different transmission lines in Korea. The fault signals were recorded in COMTRADE format and its sampling rates vary. We resampled the data to 64 samples per cycle for consistency. Table 3 shows the fault type of the 52 data. These data are not used in training but used in tests.

3) ML ALGORITHMS SETUP

We analyze how features influence the classification accuracy of real fault data, aiming to optimize model performance by selecting the most effective features. The sizes of each feature and the hyperparameters of ML-based algorithms are listed in Table 4 and Table 5, respectively. The number of homogeneous models applied to the voting method M is set as 10. The value of Δ_j as the boundary condition threshold is designed to be 25% of ϵ_j.

To make a robust algorithm, which is one of the main purposes of this work, we set the hyperparameters of the ML algorithms to minimize overfitting. Specifically, NN-based algorithms incorporate dropout layers. In addition, to ensure a fair comparison, the number of neurons in the NN-based algorithms, such as ANN, CNN, and LSTM, was kept consistent, and the tree-based algorithms used default models.[3]

B. RULE-BASED ALGORITHM SETUP

1) RMS RATIO FOR FEATURE

To illustrate the robustness of features, we use signal-to-noise ratio (SNR) of which the mathematical definition in dB is

$$\mathrm{SNR}_{\mathrm{dB}} = 20 \log_{10} \left(\frac{P_s}{P_n} \right), \tag{13}$$

where P_s and P_n denote the powers of original and noise signals, respectively. The change of MODWT-E and RMS ratio values depending on SNR are evaluated by comparing with the original data: cosine similarity and mean square error (MSE). Cosine similarity CS measures the similarity of two data sets with a higher number indicating more similarity. It is defined as

$$\mathrm{CS} = \frac{1}{N} \sum_{i=1}^{N} \frac{\boldsymbol{x}_s^i \cdot \boldsymbol{x}_n^i}{\left|\boldsymbol{x}_s^i\right| \left|\boldsymbol{x}_n^i\right|}, \tag{14}$$

where $\boldsymbol{x}_s$ and $\boldsymbol{x}_n$ denote the vectors of the original and noise signals of each phase, respectively, and N denotes the size of vectors. Because MSE is an error indicator, lower MSE stands for more similarity. The definition of MSE is

$$\mathrm{MSE} = \frac{1}{4N} \sum_{i=1}^{N} \left| \boldsymbol{x}_s^i - \boldsymbol{x}_n^i \right|^2. \tag{15}$$

The change of ratio for cosine similarity and MSE according to SNR are shown in Tables 6 and 7, respectively. As the SNR decreases, the MSE of MODWT-E spikes, and the cosine similarity plummets, whereas the RMS ratio shows relatively stable in both metrics. Also, we compare features of the original and noisy data. Figs. 5 and 6 show examples of the RMS ratio comparing original and noisy data for MODWT-E and RMS, respectively. According to the results, the MODWT-E ratio shows that the faulted phase is clearly characterized in the original data but becomes unclear in the data with an SNR of 25, while the RMS shows that the original and noisy data are indistinguishable.

These results show that RMS is more robust to noise than MODWT-E. Therefore, we choose RMS as the main feature of the rule-based algorithm.

[3]These hyperparameters were not selected by the real fault data, i.e., test data, but by trial and error based on simulation data, i.e., training data. Therefore, the accuracy performance obtained by simulation data is not guaranteed with real fault data. as shown in the following sections.

TABLE 6. Comparison of cosine similarity according to SNR.

SNR	20	25	30	35	40	45	50	55	60
MODWT-E ratio	0.6998	0.8106	0.9019	0.9554	0.9791	0.9940	0.9990	0.9998	0.9999
RMS ratio	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

TABLE 7. Comparison of MSE according to SNR.

SNR	20	25	30	35	40	45	50	55	60
MODWT-E ratio	14.7292	8.5394	4.3060	2.0998	0.7353	0.2050	0.0394	0.0067	0.0014
RMS ratio	0.0002	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

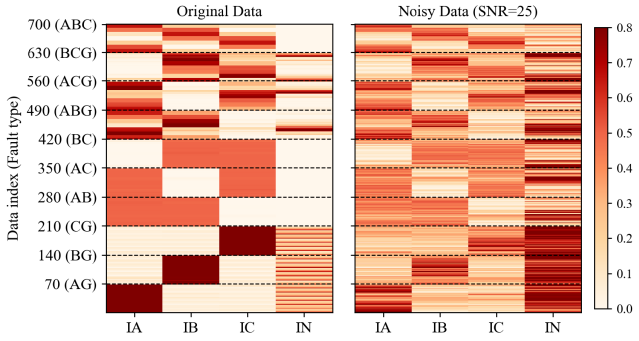


FIGURE 5. Comparison of MODWT-E ratio between original data and noisy data. The cell value of the heat map means the MODWT-E ratio.

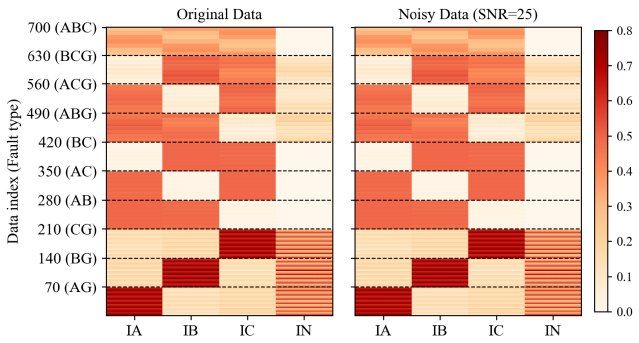


FIGURE 6. Comparison of RMS ratio between original data and noisy data. The cell value of the heat map means the RMS ratio.

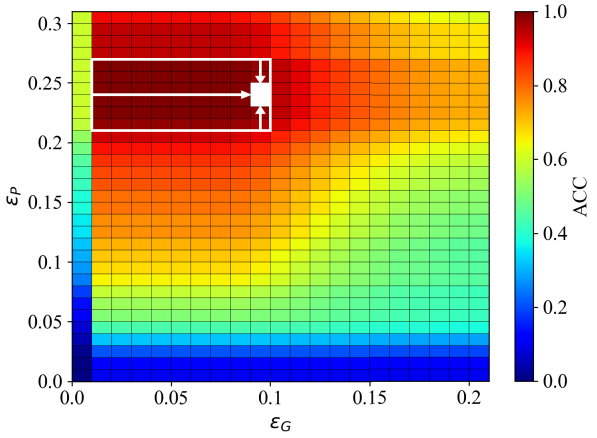


FIGURE 7. Accuracy according to (ϵ_P, ϵ_G). White big and small boxes are the range with accuracy above 0.95 and selected values, respectively.

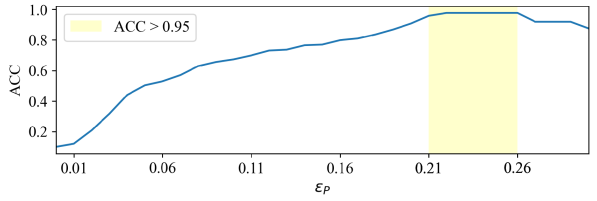


FIGURE 8. Relationship between ϵ_P and accuracy.

2) THRESHOLDS SELECTION

This section explains the process of two thresholds selection for the rule-based algorithm: ϵ_j where $j \in \{A, B, C\}$, called ϵ_P and ϵ_G. We first find the range of (ϵ_j, ϵ_G) where the classification accuracy of the simulation data is above 0.95, i.e., white big box in Fig. 7. Considering the discrepancy between simulation data and real fault data, a margin of 0.05 is added to the range. Then, we select (ϵ_P, ϵ_G) at the value of the white small box in Fig. 7. The reason for this is as follows.

- ϵ_G: The ϵ_G is the threshold value to detect fault type including ground. In fault classification, the neutral current is used for distinguishing between ground fault and short circuit. The neutral current in the real fault involving the three-phase imbalance characteristic is relatively large compared to the simulation because the simulation assumes a three-phase balance environment. Considering the three-phase imbalance of the real environment, we select the maximum value of ϵ_G as 0.09 among the possible range, i.e., white big box in Fig. 7. The selecting process of ϵ_G is visualized with horizontal arrows in the figure.
- ϵ_P: We select ϵ_P by analyzing relationship between ϵ_P and accuracy performance. Fig. 8 shows the accuracy performance as a function of ϵ_P and the range with $\epsilon_G = 0.09$. To choose ϵ_P as a reasonable value, we needed to experiment to see which labels would be disadvantageous to predict if we deviated from the optimal range. Therefore, we analyzed the point-wise results for $\epsilon_P = 0.20$, $\epsilon_P = 0.27$, which are out of the optimal range, and $\epsilon_P = 0.235$, which is in the optimal range as a reference value. Fig. 9 shows the classification

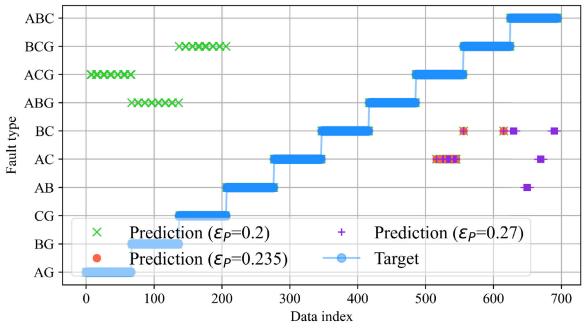


FIGURE 9. Point-wise result of fault classification with three different values of ϵ_P using simulation data. Among the three values, $\epsilon_P = 0.235$ shows a balanced performance.

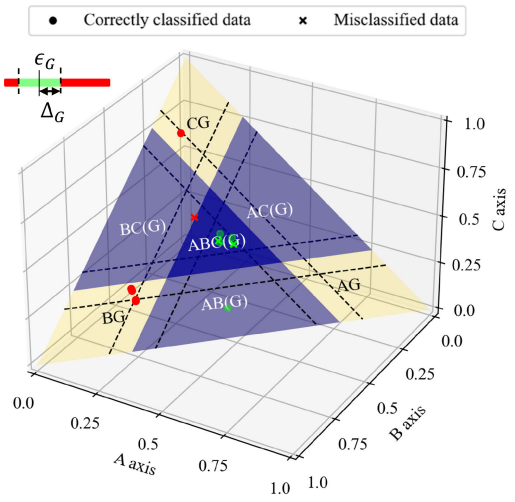


FIGURE 10. Cases for lying boundary condition in real fault dataset. Red and green dots are data points meeting three-phase and neutral current boundary conditions, respectively.

results. It shows that the smaller the value ($\epsilon_P = 0.2$), the higher the percentage of misclassified LG fault, and conversely, the higher the value ($\epsilon_P = 0.27$), the higher the percentage of misclassified LLL fault. Considering the result that the relatively unpredictable label has a trade-off based on the value of ϵ_P, it seems reasonable to select ϵ_P as the average ($\epsilon_P = 0.235$) of the optimal range based on simulation data for fault type classification of real fault data. The selecting process of ϵ_P is visualized with vertical arrows in Fig. 7.

C. PERFORMANCE EVALUATIONS AND ANALYSIS

This section first shows the performance of two single-stage algorithms: rule-based and ML-based algorithms. After analyzing their limitations, the proposed two-stage algorithm's performance is presented.

1) PERFORMANCE EVALUATION OF RULE-BASED ALGORITHM

We evaluated the proposed rule-based algorithm by setting $\epsilon_P = 0.235$ and $\epsilon_G = 0.09$ as shown in Section IV-B2. Because tree-based algorithms also work with thresholds, the performance of two representative tree-based algorithms, i.e., LGBM and XGBoost, are evaluated to highlight the effectiveness of the expert knowledge-based algorithm, i.e., rule-based algorithm. For a fair comparison, we use the same input feature (RMS ratio) for both ML-based and rule-based algorithms.

The classification accuracy of the rule-based algorithm is 0.9571 and 0.9423 for simulation and real fault data, respectively. The accuracy of 0.9423 for real fault data means that the algorithm successfully classified 49 cases out of 52 cases and that it misclassified the other three cases. The two tree-based algorithms show an accuracy of 1 in simulation data, while their accuracy severely degrades to 0.5961 and 0.6538 for XGBoost and LGBM in real fault data, respectively. These results confirm that tree-based algorithms trained by simulation data tend to be overfitting, so the algorithms perform poorly with different data sets. On the other hand, because the rule-based algorithm uses expert knowledge, it is more robust to data uncertainty.

2) ANALYSIS OF MISCLASSIFIED DATA BY RULE-BASED ALGORITHM

If the value of the RMS ratio lies in the boundary condition, the decision made by the rule-based algorithm could be wrong with a high probability. It is one of the core reasons for the proposed two-stage classification algorithm. We visualize the cases that satisfy the boundary condition as shown in Fig. 10. In detail, twelve cases lie in the boundary condition in the real fault dataset. The twelve cases include all the misclassified results, i.e., three cases, by the rule-based algorithm.

Table 8 shows more detailed results of misclassified data for real fault data. The first row shows that a CG fault is misclassified as a BCG fault. This is because the fault currents of A, B, and C are small and similar. Also, the rule-based algorithm misclassifies LLL fault as LLLG fault, which means that the neutral current of LLL fault in real fault data is relatively large compared to simulation data. Therefore, the proposed rule-based algorithm is not possible to classify correctly in these cases.

3) PERFORMANCE EVALUATION OF ML-BASED ALGORITHMS

In this section, we compare the performance of four ML-based algorithms: XGBoost, LGBM, ANN, CNN, and LSTM. The candidate features for inputs of the ML-based algorithms are IV, moving window RMS (hereinafter referred to as RMS), MODWT, and CWT explained in Section II-B.

Tables 9 and 10 show the accuracy results using simulation data and real data, respectively. All five ML-based algorithms are good at the classification of the four features in simulation data. In particular, LGBM and CNN show perfect accuracy

TABLE 8. Result of misclassified real fault data by rule-based algorithm.

r_A	r_B	r_C	r_G	phase satisfying boundary condition	target	prediction by rule-based
0.2315	0.3352	0.4333	0.5221	A	CG	BCG
0.3272	0.3335	0.3393	0.1104	G	ABC	ABCG
0.3726	0.3055	0.3220	0.0932	G	ABC	ABCG

TABLE 9. Comparison accuracy on simulation test data.

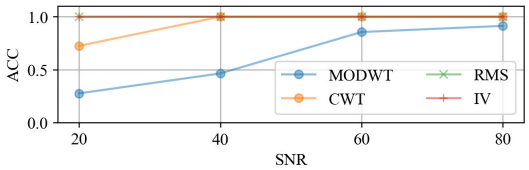
	IV	RMS	MODWT	CWT
XGBoost	1.0000	0.9855	1.0000	1.0000
LGBM	1.0000	1.0000	1.0000	1.0000
ANN	1.0000	1.0000	0.9855	0.9855
CNN	1.0000	1.0000	1.0000	1.0000
LSTM	1.0000	1.0000	0.9275	1.000

TABLE 10. Comparison accuracy on real fault data.

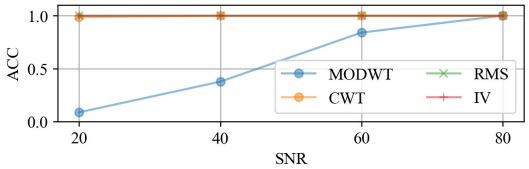
	IV	RMS	MODWT	CWT
XGBoost	0.3077	0.6346	0.3269	0.4231
LGBM	0.4038	0.6346	0.3846	0.4615
ANN	0.5385	**0.9615**	0.5000	0.8462
CNN	**0.9615**	0.8462	0.4615	0.9038
LSTM	0.5000	0.9231	0.5000	0.4038

for all features. So, we can conclude that the differences among the algorithms and the features are negligible in the simulation data. For real fault data, NN-based algorithms show better classification performance than those of tree-based algorithms as discussed in Section IV-C1. Except CNN, RMS shows the best performance for all ML-based algorithms in real fault data. Because CNN was originally designed for image data, it shows better performance with highly fluctuating features like IV. Nevertheless, CNN with RMS also shows a good performance. It confirms that RMS is robust to noise as discussed in Section IV-B1. Note that an accuracy of 0.9615, the best performance among all ML-based algorithms, means that two cases out of 52 cases are misclassified.

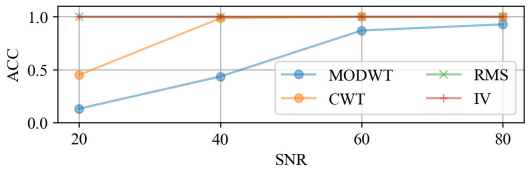
Because real fault data includes noise, we examine the robustness against noise for various features. To compare the robustness against noise, we use NN-based algorithms, which showed a good performance in real fault data. The range of noise levels is SNR of 20, 40, 60, and 80 dB. Fig. 11 shows the accuracy performance of NN-based classification algorithms according to noise level. WT-based features, i.e., MODWT and CWT, decrease relatively fast in accuracy performance as the SNR decreases. This is because noise at the boundaries of the signal can produce large coefficients with WT. RMS shows the best performance in the three NN-based algorithms. This result is consistent with the performance degradation in real fault data shown in Table 10.



(a) Result of ANN



(b) Result of CNN



(c) Result of LSTM

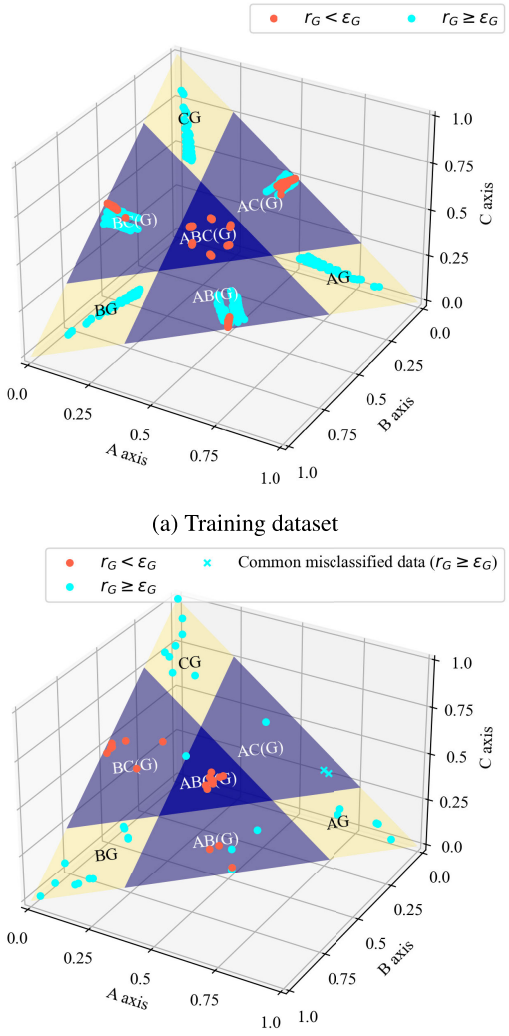
FIGURE 11. Comparison of robustness to noise among features (MODWT, CWT, RMS, and IV).

TABLE 11. Comparison accuracy on simulation test data.

	IV	RMS	MODWT	CWT
XGBoost	1.000	1.000	1.000	1.000
LGBM	1.000	1.000	1.000	1.000
ANN	1.000	1.000	0.9855	1.000
CNN	1.000	1.000	1.000	1.000
LSTM	1.000	1.000	0.9855	1.000

4) ANALYSIS OF MISCLASSIFIED DATA BY ML-BASED ALGORITHMS

This section explains the two misclassified cases by ML-based algorithms. Fig. 12 shows the distribution of the training dataset, i.e., the simulation dataset, and the real fault dataset. The "x" markers in Fig. 12b indicate the two misclassified data in the real fault data by CNN-IV and ANN-RMS based algorithms. Given that the simulated training data by label has a clustered distribution, it is obvious that two misclassified data points are far from any cluster. We can conclude that these misclassification results come from the fundamental overfitting problem of ML-based algorithms.



(a) Training dataset

(b) Real fault test dataset

FIGURE 12. Distribution of RMS ratio. It is observed that the distributions of the simulation data and the real fault data are different.

5) PERFORMANCE EVALUATION OF TWO-STAGE ALGORITHMS

The classification results of the proposed two-stage algorithms using simulation data and real fault data are summarized in Table 11 and Table 12, respectively. The proposed algorithm shows 100% accuracy with simulation data in all cases except for the ANN-MODWT and LSTM-MODWT based algorithms. Also, the results for real fault data show that the two-stage algorithm outperforms the single-stage methods of XGBoost, LGBM, ANN, CNN, and LSTM by 94.3%, 83.7%, 28.4%, 18.2%, and 62.8% on average, respectively.

Among the 20 ML-feature combinations, six cases outperform all single-stage algorithms: ANN-RMS, ANN-CWT, CNN-IV, CNN-RMS, CNN-CWT, and LSTM-RMS. Among ML-based algorithms, only NN-based algorithms are meaningful to integrate with the rule-based algorithm, which

TABLE 12. Comparison accuracy on real fault data. Cases that outperformed compared to the rule-based and ML-based algorithms are bolded.

	IV	RMS	MODWT	CWT
XGBoost	0.7885	0.8269	0.8462	0.8269
LGBM	0.8462	0.8654	0.8846	0.8654
ANN	0.8462	**1.0000**	0.8269	**0.9808**
CNN	**1.0000**	**0.9808**	0.8077	**0.9615**
LSTM	0.8654	**0.9615**	0.8462	0.8269

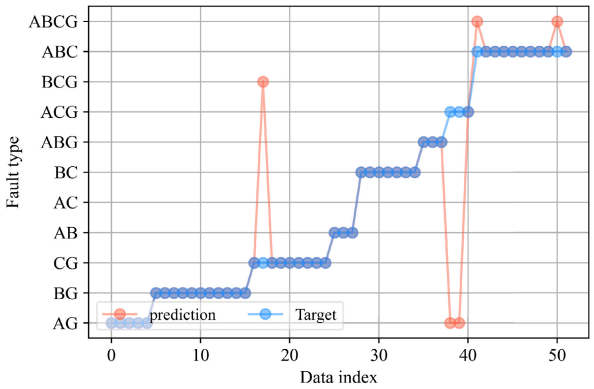


FIGURE 13. Point-wise classification result of previous algorithm.

is consistent with the results in Section IV-C3. In particular, the ANN-RMS and CNN-IV cases show the perfect accuracy performance, while none of the single-stage algorithms achieve it.

The average execution times for rule-based, ML-based, and two-stage algorithms are 0.11 ms, 84 ms, and 32 ms, respectively. It includes the time from data preprocessing to the final fault type decision for the whole 52 real fault data. The rule-based algorithm, which has a simple preprocessing compared to ML, shows the fastest computation speed, followed by the proposed two-stage algorithm. The proposed algorithm only executes the ML stage if the boundary condition is met, resulting in the second-highest speed.

6) PERFORMANCE OF ANOTHER HYBRID ALGORITHM

To demonstrate the effectiveness of the proposed two-stage algorithm, we demonstrate another hybrid algorithm's performance. To the best of our knowledge, there is no two-stage classification algorithm. Therefore, we add a hybrid approach to previous works [34].

The previous work uses single-phase current as an input feature to determine the fault of each phase, thereby deriving the final fault type. Leveraging this structural characteristic, we implement a hybrid approach that horizontally integrates rule-based and ML-based methods. Specifically, the fault states of the three phases are derived using the ML-based method, while the fault state of the neutral current is derived using a rule-based method because the neutral current shows the largest discrepancy between simulation data and real data. As algorithm settings, the ANN-RMS case is chosen for the

ML-based method, and the rule-based is the same algorithm used in this paper.

The accuracy of the previous work for real fault data is 0.9038. That is, five cases out of 52 cases are misclassified. Fig. 13 shows the point-wise classification result of the previous work. The misclassified data of this algorithm are the sum of the misclassified data of rule-based and ML-based algorithms. This result indicates that performance improvement cannot be expected by simply integrating two methods, but also shows the effectiveness of the proposed two-stage algorithm.

V. CONCLUSION

In this work, we proposed a two-stage fault classification algorithm combining rule-based and machine learning (ML)-based methods. The two-stage approach overcomes the limitations of single-stage algorithms: the threshold problem of the rule-based algorithm and the overfitting problem of the ML-based algorithms. The proposed algorithm shows 100% classification accuracy for real fault data without using the real data during the training period. Through diverse experiments, it is confirmed that the RMS is the most robust and reliable feature and that NN-based algorithms such as ANN and CNN outperform tree-based algorithms. Due to the robust performance of the proposed algorithm, it is expected that its application in the field can reduce unnecessary investment and enable efficient operations and management through core reinforcements.

However, the performance of the proposed two-stage algorithm cannot be guaranteed for fault data that deviates significantly from the simulation data. This limitation is a common problem inherent in all fault classification algorithms. To overcome this, we plan to generate more general fault data using generative adversarial networks (GAN). Additionally, we aim to apply the two-stage algorithm to other fault diagnosis scenarios, such as fault localization and fault prediction, to enhance the robustness and utility of the power system.

REFERENCES

[1] B. Kasztenny, B. Campbell, and J. Mazereeuw, "Phase selection for single-pole tripping—Weak infeed conditions and cross-country faults," in *Proc. Annual Western Protective Relay Conf.*, Spokane, Oct. 2000, pp. 1–19.

[2] J. L. Blackburn, *Protective Relaying Principles and Applications*, 2nd ed., New York, NY, USA: Marcel Dekker, 1998.

[3] S. R. Fahim, Y. Sarker, O. K. Islam, S. K. Sarker, Md. F. Ishraque, and S. K. Das, "An intelligent approach of fault classification and localization of a power transmission line," in *Proc. IEEE Int. Conf. Power, Electr., Electron. Ind. Appl. (PEEIACON)*, Nov. 2019, pp. 53–56.

[4] A. Yadav and Y. Dash, "An overview of transmission line protection by artificial neural network: Fault detection, fault classification, fault location, and fault direction discrimination," *Adv. Artif. Neural Syst.*, vol. 2014, pp. 1–20, Dec. 2014.

[5] F. B. Costa, B. A. Souza, and N. S. D. Brito, "Real-time classification of transmission line faults based on maximal overlap discrete wavelet transform," in *Proc. PES TD*, Orlando, FL, USA, May 2012, pp. 1–8.

[6] O. A. S. Youssef, "Combined fuzzy-logic wavelet-based fault classification technique for power system relaying," *IEEE Trans. Power Del.*, vol. 19, no. 2, pp. 582–589, Apr. 2004.

[7] A. K. Pradhan, A. Routray, S. Pati, and D. K. Pradhan, "Wavelet fuzzy combined approach for fault classification of a series-compensated transmission line," *IEEE Trans. Power Del.*, vol. 19, no. 4, pp. 1612–1618, Oct. 2004.

[8] S. R. Samantaray, "A systematic fuzzy rule based approach for fault classification in transmission lines," *Appl. Soft Comput.*, vol. 13, no. 2, pp. 928–938, Feb. 2013.

[9] A. Jamehbozorg and S. M. Shahrtash, "A decision-tree-based method for fault classification in single-circuit transmission lines," *IEEE Trans. Power Del.*, vol. 25, no. 4, pp. 2190–2196, Oct. 2010.

[10] T. S. Abdelgayed, W. G. Morsi, and T. S. Sidhu, "A new harmony search approach for optimal wavelets applied to fault classification," *IEEE Trans. Smart Grid*, vol. 9, no. 2, pp. 521–529, Mar. 2018.

[11] V. N. Ogar, S. Hussain, and K. A. A. Gamage, "Transmission line fault classification of multi-dataset using CatBoost classifier," *Signals*, vol. 3, no. 3, pp. 468–482, Jul. 2022.

[12] Z. He, S. Lin, Y. Deng, X. Li, and Q. Qian, "A rough membership neural network approach for fault classification in transmission lines," *Int. J. Electr. Power Energy Syst.*, vol. 61, pp. 429–439, Oct. 2014.

[13] M. Sanaye-Pasand and H. Khorashadi-Zadeh, "Transmission line fault detection & phase selection using ANN," in *Proc. Int. Conf. Power Syst. Transients*, 2003, pp. 1–6.

[14] M. Pazoki, "A new fault classifier in transmission lines using intrinsic time decomposition," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 619–628, Feb. 2018.

[15] K. M. Silva, B. A. Souza, and N. S. D. Brito, "Fault detection and classification in transmission lines based on wavelet transform and ANN," *IEEE Trans. Power Del.*, vol. 21, no. 4, pp. 2058–2063, Oct. 2006.

[16] J. Fang, K. Chen, C. Liu, and J. He, "An explainable and robust method for fault classification and location on transmission lines," *IEEE Trans. Ind. Informat.*, vol. 19, no. 10, pp. 10182–10191, Aug. 2023.

[17] P. Rai, N. D. Londhe, and R. Raj, "Fault classification in power system distribution network integrated with distributed generators using CNN," *Electric Power Syst. Res.*, vol. 192, Mar. 2021, Art. no. 106914.

[18] Y. Xi, W. Zhang, F. Zhou, X. Tang, Z. Li, X. Zeng, and P. Zhang, "Transmission line fault detection and classification based on SA-MobileNetV3," *Energy Rep.*, vol. 9, pp. 955–968, Dec. 2023.

[19] S. R. Fahim, S. K. Sarker, S. M. Muyeen, S. K. Das, and I. Kamwa, "A deep learning based intelligent approach in detection and classification of transmission line faults," *Int. J. Electr. Power Energy Syst.*, vol. 133, Dec. 2021, Art. no. 107102.

[20] F. Rafique, L. Fu, and R. Mai, "End to end machine learning for fault detection and classification in power transmission lines," *Electric Power Syst. Res.*, vol. 199, Oct. 2021, Art. no. 107430.

[21] S. Belagoune, N. Bali, A. Bakdi, B. Baadji, and K. Atif, "Deep learning through LSTM classification and regression for transmission line fault detection, diagnosis and location in large-scale multi-machine power systems," *Measurement*, vol. 177, Jun. 2021, Art. no. 109330.

[22] T. Kim, S. Lim, K. Song, and S.-G. Yoon, "LSTM-based fault classification model in transmission lines for real fault data," *Trans. Korean Inst. Electr. Engineers*, vol. 73, no. 3, pp. 585–592, Mar. 2024.

[23] A. Moradzadeh, H. Teimourzadeh, B. Mohammadi-Ivatloo, and K. Pourhossein, "Hybrid CNN-LSTM approaches for identification of type and locations of transmission line faults," *Int. J. Electr. Power Energy Syst.*, vol. 135, Feb. 2022, Art. no. 107563.

[24] H.-T. Yang and C.-C. Liao, "A de-noising scheme for enhancing wavelet-based power quality monitoring system," *IEEE Trans. Power Del.*, vol. 16, no. 3, pp. 353–360, Jul. 2001.

[25] L. Angrisani, P. Daponte, M. D'Apuzzo, and A. Testa, "A measurement method based on the wavelet transform for power quality analysis," *IEEE Trans. Power Del.*, vol. 13, no. 4, pp. 990–998, Jul. 1998.

[26] Z. K. Peng and F. L. Chu, "Application of the wavelet transform in machine condition monitoring and fault diagnostics: A review with bibliography," *Mech. Syst. Signal Process.*, vol. 18, no. 2, pp. 199–221, Mar. 2004.

[27] C. Kern, T. Klausch, and F. Kreuter, "Tree-based machine learning methods for survey research," *Surv. Res. Methods*, vol. 13, no. 1, p. 73, Apr. 2019.

[28] M. X. Cohen, "A better way to define and describe Morlet wavelets for time-frequency analysis," *NeuroImage*, vol. 199, pp. 81–86, Oct. 2019.

[29] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[30] G. Ke, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–9.
[31] C. M. Bishop, *Neural Networks for Pattern Recognition*. London, U.K.: Oxford Univ. Press, 1995.
[32] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[34] K. Moloi, N. W. Ndlela, and I. E. Davidson, "Fault classification and localization scheme for power distribution network," *Appl. Sci.*, vol. 12, no. 23, p. 11903, Nov. 2022.

SE-HEON LIM (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering from Soongsil University, Seoul, South Korea, in 2018, where she is currently pursuing the Ph.D. degree. Her research interests include energy big data and distribution system operation via machine learning applications.

TAEGEUN KIM (Graduate Student Member, IEEE) received the B.S. degree in electrical and electronics engineering from Kangwon University, Chuncheon, South Korea, in 2020. He is currently pursuing the Ph.D. degree with Soongsil University, Seoul, South Korea. His research interests include energy big data and load forecasting via machine learning applications.

KYEONG-YEONG LEE (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering from Soongsil University, Seoul. South Korea, in 2021, where he is currently pursuing the M.S. degree. His research interests include energy big data and power system inertia estimation via machine learning applications.

KYUNG-MIN SONG received the B.S. degree in electrical engineering from Soongsil University, Seoul, South Korea, in 2022, where he is currently pursuing M.S. degree. His research interests include energy big data and data and load forecasting via machine learning applications.

SUNG-GUK YOON (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2006 and 2012, respectively. He was a Postdoctoral Researcher with Seoul National University, from 2012 to 2014. Since 2014, he has been with Soongsil University, where he is currently an Associate Professor. His research interests include energy big data, game theory for power systems, and communication networks.