Detection and Classification of Transmission Line Faults Based on Unsupervised Feature Learning and Convolutional Sparse Autoencoder

Kunjin Chen, Jun Hu, and Jinliang He, *Fellow, IEEE*

***Abstract*—We present in this paper a novel method for fault detection and classification in power transmission lines based on convolutional sparse autoencoder. Contrary to conventional methods, the proposed method automatically learns features from a dataset of voltage and current signals, on the basis of which a framework for fault detection and classification is created. Convolutional feature mapping and mean pooling are implemented in order to generate feature vectors with local translation-invariance for half-cycle multi-channel signal segments. Fault detection and classification are achieved by a softmax classifier using the feature vectors. Further, the proposed method is tested under different sampling frequencies and signal types. The generalizability of the proposed method is also verified by adding noise and measurement errors to the data. Results show that the proposed method is fast and accurate in detecting and classifying faults, and is practical for online transmission line protection for its high robustness and generalizability.**

***Index Terms*—Convolutional sparse autoencoder (CSAE), fault detection, fault classification, transmission lines, unsupervised learning.**

I. Introduction

FAULT detection and classification are two important aspects of power transmission line protection. Over the years, researchers have been seeking to realize fast and accurate detection and classification of faults in transmission lines using various methods, so that the faulted system can be protected from possible destructive effects caused by the fault. Further, the information provided by fault detection and classification can greatly facilitate the location of fault, thus reducing the fault clearing time.

The extraction of features from the voltage and current signals, which is implemented purposefully, helps researchers better understand the nature and characteristics of the fault detection and classification tasks, and they can thus fulfill these tasks in a more consistent and effective manner [1]. Transforming signals from time domain to frequency domain is frequently used for feature extraction. Discrete Fourier transform (DFT), a widely adopted tool for signal analysis, is used in the forms of full cycle discrete Fourier transform (FCDFT) [2], [3] and half cycle discrete Fourier transform (HCDFT) [4], [5]. Discrete wavelet transform (DWT) is used by researchers to obtain information in certain frequency ranges [6]–[8]. The DWT coefficients of different frequency ranges (decomposition levels) are usually used to generate features. S-transform (ST) is used as it reveals local spectral characteristics [9], [10]. A number of features can be extracted out of the S-matrix produced by ST. In addition to frequency domain-based methods, researchers also adopt modal transformations such as Clarke transformation to extract useful features [11], [12]. Further, dimensionality reduction methods such as principal component analysis (PCA) can be used to produce more suitable inputs for certain fault classification methods [13], [14]. Though the above-mentioned feature extraction techniques have been applied to different types of transmission line systems with different configurations, much prior knowledge of the specific system configuration is required and the process of determining the implementation details oftentimes needs repeated modification and adjustment. Thus, implementing these techniques can be quite time-consuming and lacks generalizability. Take DWT as an example, researchers first need to determine which mother wavelet and which decomposition levels to use before they can extract the features. As there are a large number of mother wavelets (e.g., Coiflets, Meyer, Daubechies and Symlets) to choose from and the number of decomposition levels is affected by the sampling frequency, it is hard to tell which combination of the options shall be chosen, not to mention the fact that some mother wavelet families have a series of wavelets and different features can be extracted out of the coefficients of different decomposition levels (e.g., energy and maximum of coefficients at one decomposition level) [1].

On the basis of feature extraction, the task of fault detection can be fulfilled by setting thresholds for the extracted features. Similar to fault detection, the task of fault classification can also be done by setting a series of if-then conditions with preset thresholds [15], [16]. Other methods mainly adopt artificial intelligence-based models. Artificial neural networks (ANNs) including feedforward neural network (FNN), radial basis function network (RBFN) and

Manuscript received March 14, 2016; revised June 21, 2016; accepted August 7, 2016. Date of publication August 10, 2016; date of current version April 19, 2018. This work was supported in part by the State Key Development Program (973 Program) for Basic Research of China under Grant 2013CB228206, and in part by the National Natural Science Foundation of China under Grant 51429701. Paper no. TSG-00329-2016. *(Corresponding author: Jinliang He.)*

The authors are with the State Key Laboratory of Power Systems, Department of Electrical Engineering, Tsinghua University, Beijing 100084, China (e-mail: hejl@tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2016.2598881

probabilistic neural network (PNN) are extensively used to classify different types of faults [7], [13], [17], [18]. Support vector machines (SVMs) are used for their structural risk-minimizing nature [19], [20]. Fuzzy logic based methods such as fuzzy-neuro approaches and adaptive-network-based fuzzy inference systems (ANFISs) are also adopted by some researchers [4], [21]. Other methods used for fault classification include decision trees (DTs) and random forests (RFs) [22], [23]. Although the aforementioned classification models are well-developed and have been proven effective, the combination of a certain feature extraction technique and a certain classification model is almost arbitrary, thus taking researchers much time to evaluate the performance of different classification models.

As pointed out previously, the above-mentioned methods require hand-designed features specific for system configurations and parameters. Though favorable performance may be achieved, the process of feature design and feature selection is often time-consuming and lacks generalizability. Thus, it is desirable to implement some feature extraction methods that does not require much prior knowledge, so that the method can be generalized to different cases without making significant modifications. Recent progress in the development of machine learning has been receiving an ever-growing attention in that the models and algorithms are increasingly able to automatically extract features with multiple levels of abstraction from large amounts of data [24]. In the fields such as computer vision, speech recognition, and natural language processing, researchers have been able to build end-to-end models with much better performances than traditional models which require hand-designed features [25]–[27]. One of the key elements in the implementation of the models is that unlabeled data can be used by one-layer structures such as sparse autoencoder (SAE) to help extract features and pre-train the models used for classification tasks [28]. Further, the features extracted by SAE can be used by convolutional neural networks (CNNs) which far outperform classical methods (e.g., methods based on scale-invariant feature transform (SIFT) and SVM) in the image classification task [25]. In summary, adopting the unsupervised feature learning process has a threefold advantage: (i) the time-consuming process of feature design can be replaced by automatic feature learning with increased generalizability, (ii) large amounts of unlabeled data collected by online monitoring devices can be fully utilized, and (iii) the feature extracting approach is fully compatible with powerful machine learning models including CNN.

This paper presents a novel method for fault detection and classification inspired by the above-mentioned studies and concepts. The unsupervised feature learning by SAE is used to automatically extract features from voltage and current signals, as introduced in [29]. Instead of utilizing the signals individually, we combine three phase voltage and current signals as a multi-channel signal. A related work using CNN to analyze multi-channel sequence is proposed in [30]. Rather than directly using the CNN model, we propose a framework based on convolutional sparse autoencoder (CSAE) [31] and softmax classifier to fulfill the tasks of fault detection and classification in power transmission lines (adding a softmax classifier as the last layer of the network is standard practice for implementing CNN [25]). The training strategy using data from different time ranges and the computationally efficient testing strategy with a filtering operation are also proposed. The performance of the proposed method is tested with different sampling frequencies and signal types. Discussion of the proposed method in the presence of noise and measurement errors is then provided with some slight modifications to the original implementation. Further, we compare the proposed method with existing methods. The application of the proposed method in smart grids is also presented.

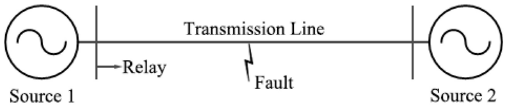


Fig. 1. The studied system with sources at both ends.

TABLE I
SYSTEM PARAMETERS USED FOR THE SIMULATION IN MATLAB/SIMULINK

System Parameter	Values or Types
Fault Distance (km)	20, 40, 60, 80, 100, 120, 140, 160, 180
Fault Resistance (Ω)	0.01, 5, 15, 20, 30, 40, 50
Fault Inception Angle (degrees)	0, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330
Pre-fault Power Angle (degrees)	10, 20, 30
Fault Type	a-g, b-g, c-g, ab, ac, bc, ab-g, ac-g, bc-g, abc-g, non-faulty

II. UNSUPERVISED LEARNING OF SPARSE MULTI-CHANNEL FEATURES FROM VOLTAGE AND CURRENT SIGNALS

A. System Studied and Data Acquisition

A simple three-phase power system is studied in this paper as shown in Fig. 1. The length of the 220 kV transmission line is 200 km and the system frequency is 50 Hz. The transmission line connects two sources and has positive sequence impedance $Z_1 = 4.76 + j59.75\ \Omega$ and zero sequence impedance $Z_0 = 77.70 + j204.26\ \Omega$. The system is modeled in MATLAB/Simulink, with which the data used in this paper is simulated. The three phase voltage and current signals are collected by the relay employed at source 1 at the sampling frequency of 20 kHz.

By varying the tunable system parameters, a dataset of voltage and current signals is generated. The system parameters used for simulation are listed in Table I. Concretely, the fault distance is the distance between fault point and the relay, and the pre-fault power angle is the phase difference between source 1 and source 2 when the fault occurs. As we try all combination of the parameters, a total of 24948 data samples are collected in the dataset. Moreover, as the sampling frequency is 20 kHz, we are able to test the effect of sampling frequencies that are less than or equal to 20 kHz.

B. Unsupervised Feature Learning by Sparse Autoencoder

In this paper, we use the SAE introduced in [32] to achieve unsupervised feature learning. Concretely, a SAE has a visible layer, a hidden layer and a reconstruction layer, and the training process ensures that the output vector corresponding to the reconstruction layer restores the input vector as much as possible for each unlabeled data sample $\boldsymbol{x} \in \mathbb{R}^n$ in the training dataset. Thus, when the training process is properly completed, the hidden nodes within the hidden layer are expected to give effective feature representations of the data in the training dataset. Given an input vector $\boldsymbol{x}$, the output vector $h(\boldsymbol{x})$ of an SAE is calculated as

$$h(\boldsymbol{x}) = \boldsymbol{W}_2 f(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2 \tag{1}$$

where $f(z) = \frac{1}{(1+\exp(z))}$ is the nonlinear sigmoid activation function, $\boldsymbol{W}_1$ is the $n_h \times n$ weight matrix associating the visible layer and the hidden layer, $\boldsymbol{b}_1$ is the bias vector for the visible layer, $\boldsymbol{W}_2$ is the $n \times n_h$ weight matrix associating the hidden layer and the reconstruction layer, and $\boldsymbol{b}_2$ is the bias vector for the hidden layer. As our goal is to minimize the difference between $\boldsymbol{x}$ and $h(\boldsymbol{x})$, a cost function capable of measuring this difference for the entire training dataset is needed. Concretely, the cost function J consists three terms, namely the squared error term, the weight decay term and the sparsity penalty term [32]. For a training dataset with m data samples, the detailed definition of the cost function is

$$J = \frac{1}{2m}\sum_{i=1}^{m}\left\| h\left(\boldsymbol{x}^{(i)}\right) - \boldsymbol{x}^{(i)}\right\|^2 + \frac{\lambda}{2}\sum_{i=1}^{2}\|\boldsymbol{W}_i\|_F^2 + \beta\sum_{j=1}^{n_h} D\left(\rho\|\widetilde{\rho}_j\right) \tag{2}$$

where the first term measures the total squared error between the input and output data and the second term is the weight decay term used to limit the magnitude of the weights so that the autoencoder is not prone to overfitting. The third term is the sparsity penalty term, in which β controls the weight of this term and $D(\rho\|\widetilde{\rho}_j)$ is the Kullback-Leibler divergence between $\widetilde{\rho}_j$ and ρ. More specifically, $\widetilde{\rho}_j$ is the average activation of hidden node j with regard to all input data in the training dataset, ρ is the sparsity parameter and $D(\rho\|\widetilde{\rho}_j)$ is calculated as [33]:

$$D\left(\rho\|\widetilde{\rho}_j\right) = \rho\log\left(\frac{\rho}{\widetilde{\rho}_j}\right) + (1-\rho)\log\left(\frac{1-\rho}{1-\widetilde{\rho}_j}\right) \tag{3}$$

By setting a very small sparsity parameter ρ (usually not more than 0.1), we can make sure that for a given input vector $\boldsymbol{x}$, the activation level of the majority of the hidden nodes is close to zero, while a small proportion of the hidden nodes are highly activated. This indicates that we can easily find some highly relevant feature representations of the input vector $\boldsymbol{x}$ by looking at the activations of the hidden nodes in the hidden layer.

To train the SAE, we optimize the cost function J by iteratively updating the weight and bias values using backpropagation algorithm [34]. After a certain number of iterations, J is expected to converge to a satisfactory local optimum, and the unsupervised feature learning by SAE is achieved.

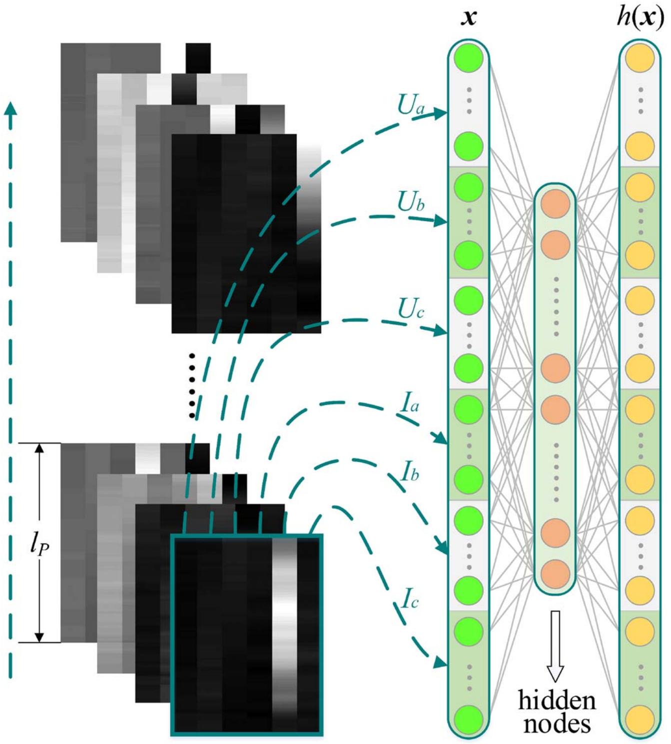


Fig. 2. Schematic diagram of learning multi-channel features of voltage and current signals by SAE.

The schematic diagram of the unsupervised feature learning procedure is illustrated in Fig. 2. In this paper, the voltage and current signals are displayed using grayscale images, such that the correlations across the channels can be clearly observed. First of all, we randomly cut out a large number of patches from the training dataset. A zero component analysis (ZCA) whitening transform is then applied to the patches, the theoretical foundations of which can be found in [35] and [36]. Concretely, for a given $d \times m$ matrix $\boldsymbol{X}$ containing m d-dimensional data samples, we use $\boldsymbol{U} = (\boldsymbol{XX}^{\mathrm{T}})^{-1/2} = \boldsymbol{PD}^{-1/2}\boldsymbol{P}^{\mathrm{T}}$ ($\boldsymbol{XX}^{\mathrm{T}}$ can always be represented as $\boldsymbol{PDP}^{\mathrm{T}}$ using some orthogonal matrix $\boldsymbol{P}$ and diagonal matrix $\boldsymbol{D}$) to transform $\boldsymbol{X}$ to $\boldsymbol{X}_Z$:

$$\boldsymbol{X}_Z = \boldsymbol{UX} \tag{4}$$

We then replace $\boldsymbol{X}$ with $\boldsymbol{X}_Z$, so that the dimensions are uncorrelated with one another and the dimensions all have the same variance [36]. After applying ZCA to the patches cut out from the training dataset, the pixels within the patches become uncorrelated and have the same variance, namely 1.

For each channel, the brightest pixel reaches the positive maximum (crest), whereas the darkest pixel reaches the negative maximum (trough). As we use all six channels of voltage and current signals simultaneously to extract the features, the size of each patch is $6 \times l_P$, l_P being the length of the patches. Thus, both $\boldsymbol{x}$ and $h(\boldsymbol{x})$ in Fig. 2 are $6l_P \times 1$ vectors. Specifically, at the sampling frequency of 20 kHz, if l_P is set to 30, then a patch covers a time span of 1.5 ms. After obtaining the patches from the signals, an SAE is trained in accordance with the above-mentioned method, and the hidden nodes can,

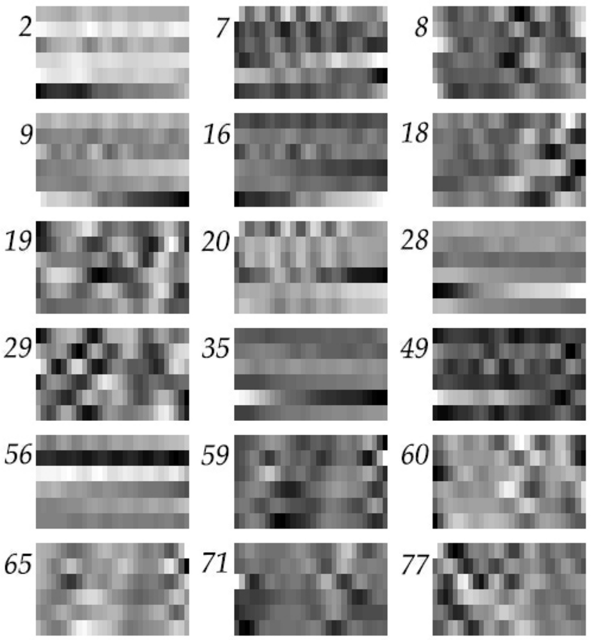


Fig. 3. Examples of extracted 6 × 30 features.

therefore, learn the features which, when combined, represent the intrinsic local characteristics of the multi-channel signals within the dataset. As introduced in [32], the element corresponding to the jth hidden node of the ith feature visualization vector is calculated as

$$f_i^j = \frac{W_1^{i,j}}{\sqrt{\sum_{j=1}^{n} W_1^{i,j}}} \tag{5}$$

where f_i^j is the jth element of the ith feature visualization vector $\boldsymbol{f}_i$, and $W_1^{i,j}$ is the element at the ith row and jth column of $\boldsymbol{W}_1$. We can then reshape the feature visualization vectors into $6 \times l_P$ matrices for visualization. Examples of 100 extracted 6×30 features ($l_P = 30$) corresponding to 100 hidden nodes are displayed in Fig. 3. The features are extracted from 250000 patches cut from the training dataset. Specifically, the channels correspond to three phase voltage and three phase current signals, respectively (top to bottom). For instance, the second half of feature 18 indicates a severe fluctuation in the voltage and current signals. Similar fluctuation patterns can also be seen in some other features, such as feature 7, 8, 19 and 29. Feature 2, 28, 35 and 56 are comparatively more moderate, in which we can see gradual changes in certain channels. In Section III, these features are used to facilitate the implementation of convolutional sparse autoencoder (CSAE).

III. Detection and Classification of Faults Based on Convolutional Sparse Autoencoder

A. The Framework for Fault Detection and Classification

In this paper, we propose and implement a framework based on CSAE to complete both the fault detection and classification tasks. A fault diagnosis system is built on the basis of this framework. Concretely, the system output is expected to be "non-faulty" when no fault occurs. A fault is detected when the output of the system changes to a specific fault type.

The framework for fault detection and classification based on CSAE is demonstrated in Fig. 4. Given a multi-channel signal, a $6 \times l_W$ window moves along the signal and an output is given for each windowed signal segment. Concretely, when the moving window arrives at the ith column of the multi-channel signal, pixels within the windowed signal segment form a $6 \times l_W$ matrix, whose first column and last column are denoted as $\boldsymbol{p}_{i-l_W+1}$ and $\boldsymbol{p}_i$, respectively. Correspondingly, the system output (the classified fault type) of this matrix is denoted as $t^{(i)}$. After the output $t^{(i)}$ is given, the window moves one column forward, so that $\boldsymbol{p}_{i-l_W+1}$ is excluded from the matrix and $\boldsymbol{p}_{i+1}$ is included as the last column. In the case of online monitoring, such a procedure is uninterruptedly repeated.

For each windowed multi-channel signal segment ($6 \times l_W$ matrix), we first use the features extracted by the SAE to map the $6 \times l_W$ matrix into convolved feature vectors. Each feature $\boldsymbol{F}_r$ ($r = 1, 2, \ldots, k$) is a $6 \times l_P$ matrix, and all the features move forward one column a time through the window while calculating dot products with all the patches they encounter. We restrain the features within the two ends of the window as they move along, so the size of each convolved feature vector is, therefore, $1 \times (l_W - l_P + 1)$. Also note that the features have been ZCA whitened previously so that the same whitening process is also applied to the patches prior to calculating the dot products. Thus, with k features available, we can get k convolved feature vectors in this feature mapping process, namely $\boldsymbol{m}_1$ to $\boldsymbol{m}_k$. Despite the fact that we need to obtain k convolved feature vectors with the size $1 \times (l_W - l_P + 1)$ for each window, which involves many calculations when completed alone, the computational burden can be greatly reduced when we compute feature mapping for multiple successive windows. Simply put, for the current window, all but the last elements of the convolved feature vectors can be directly obtained from the previous window. Thus, we only need to calculate the last element of the convolved feature vectors, which takes only k convolutional operations. This significant reduction in computational burden undoubtedly facilitates the online implementation of the proposed method.

After feature mapping, the convolved feature vectors then go through the pooling stage to generate shortened feature representations. With the help of this pooling operation, the model is less prone to overfitting and becomes more translation-invariant [37]. In this paper, we implement the simple mean pooling by calculating the mean values of the $1 \times s_p$ disjoint segments within the convolved feature vectors, s_p being the number of adjacent elements to be pooled together. It should be noted that it is acceptable if the length of the convolved feature vectors is not divisible by s_p, in which case the last few elements of the vectors are abandoned. After pooling all the k convolved feature vectors, we get k pooled convolved feature vectors, namely $\boldsymbol{d}_1$ to $\boldsymbol{d}_k$. The length of the pooled convolved feature vectors, n_p, is determined by rounding down $(l_W - l_P + 1)/s_p$, that is,

$$n_p = \left\lfloor \frac{l_W - l_P + 1}{s_p} \right\rfloor \tag{6}$$

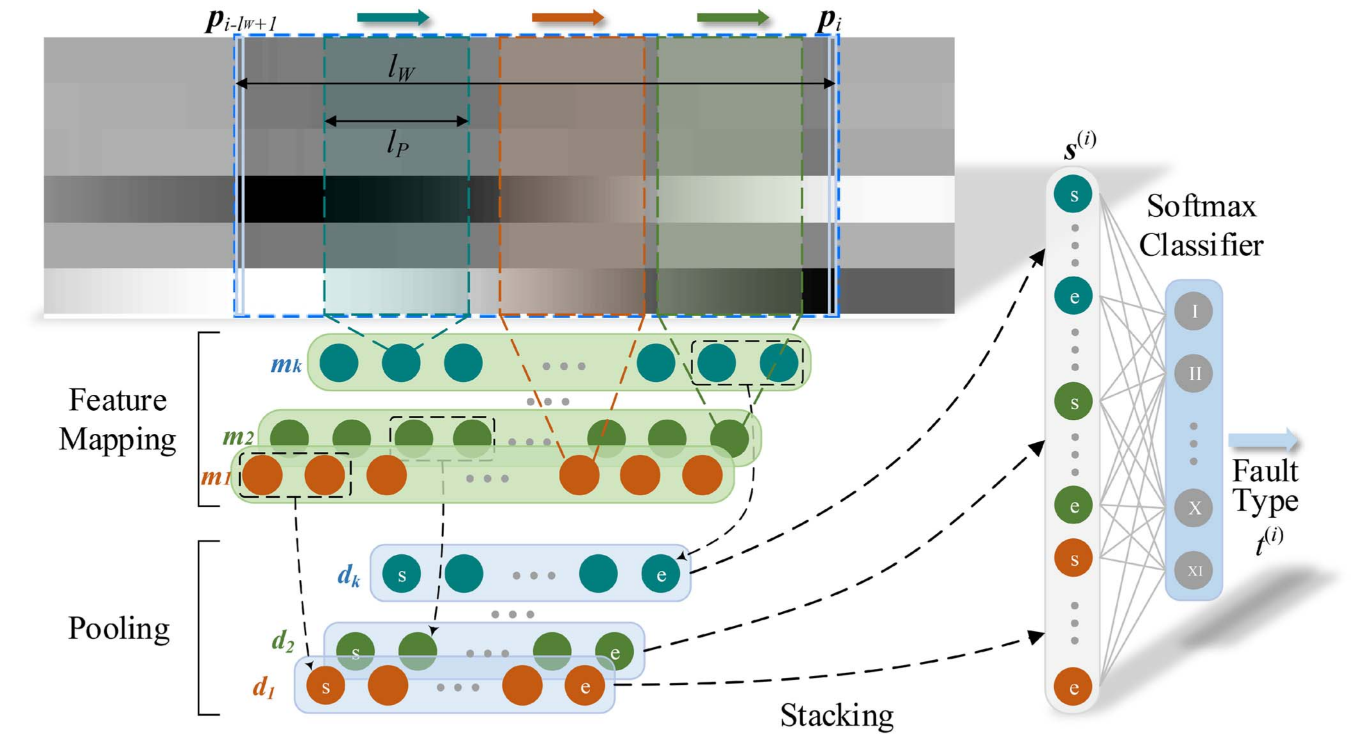


Fig. 4. The framework for fault detection and classification using CSAE and softmax classifier.

Further, the pooled convolved feature vectors are stacked into a long feature vector $\boldsymbol{s}^{(i)}$, which is used as the input vector of a softmax classifier. The length of $\boldsymbol{s}^{(i)}$, n_s, is calculated as

$$n_s = kn_p + 1 \tag{7}$$

where the additional dimension corresponds to the bias term used in the softmax classifier model. Concretely, softmax classifiers are based on the softmax regression model, which is an extension of logistic regression model and is able to solve multi-class problems [38]. For the softmax classifier, the probability of the ith stacked input vector $\boldsymbol{s}^{(i)}$ belonging to class j, $P(Y = j|\boldsymbol{s}^{(i)})$, is calculated as

$$P\left(Y = j|\boldsymbol{s}^{(i)}\right) = \frac{e^{\boldsymbol{\theta}_j^{\mathrm{T}}\boldsymbol{s}^{(i)}}}{\sum_{l=1}^{K} e^{\boldsymbol{\theta}_l^{\mathrm{T}}\boldsymbol{s}^{(i)}}} \tag{8}$$

where Y is the stochastic variable of the output class corresponding to $\boldsymbol{s}^{(i)}$ and $\boldsymbol{\theta}_j \in \mathbb{R}^{n_s}$ is the parameter vector for class $j, j = 1, 2, \ldots, K$. Consequently, for the fault classification problem with 11 fault types, an 11-dimensional vector containing all 11 probabilities is given as the output of the softmax classifier. We then assign $\boldsymbol{x}^{(i)}$ to the fault type with the highest probability:

$$t^{(i)} = \underset{j}{\operatorname{argmax}}\, P\left(Y = j|\boldsymbol{s}^{(i)}\right) \tag{9}$$

Likewise, the softmax classifier is trained using the training dataset by iteratively optimizing the cost function

$$J_s(\boldsymbol{\theta}) = -\left[\sum_{i=1}^{m}\sum_{j=1}^{K} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\boldsymbol{\theta}_j^{\mathrm{T}}\boldsymbol{s}^{(i)}}}{\sum_{l=1}^{K} e^{\boldsymbol{\theta}_l^{\mathrm{T}}\boldsymbol{s}^{(i)}}}\right] + \lambda_s \sum_{i=1}^{K} \|\boldsymbol{\theta}_i\|^2 \tag{10}$$

where $y^{(i)}$ is the actual class label of $\boldsymbol{s}^{(i)}$ and $1\{y^{(i)} = j\}$ is defined as

$$1\left\{y^{(i)} = j\right\} = \begin{cases} 1, & \text{if } y^{(i)} = j \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

A weight decay term is also added to the cost function, whose weight decay parameter is denoted as λ_s.

B. Training and Testing Strategy

In previous studies, the signal segments used to train the features and the classifier generally correspond to the same time range. However, as we depend only on the output of the softmax classifier to decide whether a fault has occurred, using training data corresponding to the same time range is insufficient. In this light, we use data corresponding to several different time ranges to form the training dataset. Further, considering the dynamic process during which the post-fault signal starts to appear at the end of the window and gradually stretch into the window, it is difficult for the classifier to distinguish among different fault types at the early stage due to the lack of information. Thus, we put some data with post-fault signal appearing at the latter half of the window into the training dataset and label them as "non-faulty", so that the classifier can intentionally ignore the data with insufficient fault-related information and only start to classify the faults when enough information is available.

Concretely, we cut off multi-channel signal segments corresponding to 11 different time ranges to form the training dataset and the test dataset. The length of the time windows is 200 with a sampling frequency of 20 kHz (i.e., half cycle). For each multi-channel signal, we denote the column of fault inception as $\boldsymbol{p}_j$ and the time range starting with this column as

$[j, j+199]$. Five time ranges containing pre-fault information are similarly denoted as $[j-120, j+79]$, $[j-80, j+119]$, $[j-60, j+139]$, $[j-40, j+159]$ and $[j-20, j+179]$. Further, the rest of the time ranges with only post-fault information are denoted as $[j+20, j+219]$, $[j+40, j+239]$, $[j+60, j+259]$, $[j+100, j+299]$ and $[j+200, j+399]$. To create the dataset, we use all the 24948 simulated multi-channel signals. A total of 274428 multi-channel signal segments are cut off from the simulated signals. Then, 70% of the segments are randomly assigned to the training dataset (194199 segments), and the rest 30% are assigned to the test dataset (83229 segments). Each segment is given a label indicating its fault type (for simplicity, "non-faulty" is also considered as a fault type). The segments of the time range $[j-120, j+79]$ are all labeled as "non-faulty".

An online test dataset with 200 full-length multi-channel signals for each type is also prepared to validate the real-time performance of the proposed system. With this online test dataset, we can testify the robustness of the system and assess its fault detection speed. Despite the fact that the training dataset only provides information up to 1 cycle after fault inception, the system does not stop giving fault classification results until 2 cycles after fault inception.

In addition, a filtering operation is applied to the output of the online fault diagnosis system. Because the multi-channel signal segments of the time range $[j-120, j+79]$ are labeled as "non-faulty", the boundary between "faulty" and "non-faulty" states becomes indistinct to some extent. Thus, for the sampling frequency of 20 kHz, the system output is filtered in such a way that any change in the output is confirmed only when the changed output remains the same for 25 consecutive sample points (1.25 ms). Therefore, any change in the output that lasts fewer than 25 consecutive sample points is filtered. This filtering operation would certainly delay the detection of faults, but it can greatly improve the online classification performance of the fault diagnosis system. The detailed performance of the proposed method and fault diagnosis system is presented in Section IV.

C. *Selection of Parameters Used in the Models*

1) Sparse Autoencoder: The SAE is used to implement the unsupervised feature learning. Concretely, a total of 250000 patches are cut from the 6×200 signal segments from the training dataset for the learning process. The hidden layer of the SAE has 100 hidden nodes. The sparsity parameter ρ, weight decay parameter λ and sparsity penalty parameter β are 0.1, 0.003 and 5, respectively.

2) Feature Mapping and Pooling: The window length l_W and patch length l_P are 200 and 30, hence the length of each convolved feature vector is 171. Moreover, we set the pooling size s_p to 5. Thus, the length of the pooled convolved feature vectors is 34.

3) Softmax Classifier: As we have 100 features and the length of each pooled convolved feature vector is 34, the input size of the softmax classifier is 3400. The weight decay parameter λ_s is 0.0001. Considering the limited computational ability of the machine used for the experiments, 45000 signal segments are randomly taken out from the complete training dataset with 194199 signal segments.

IV. Resutls and Discussion

A. *Performance of the Proposed Method*

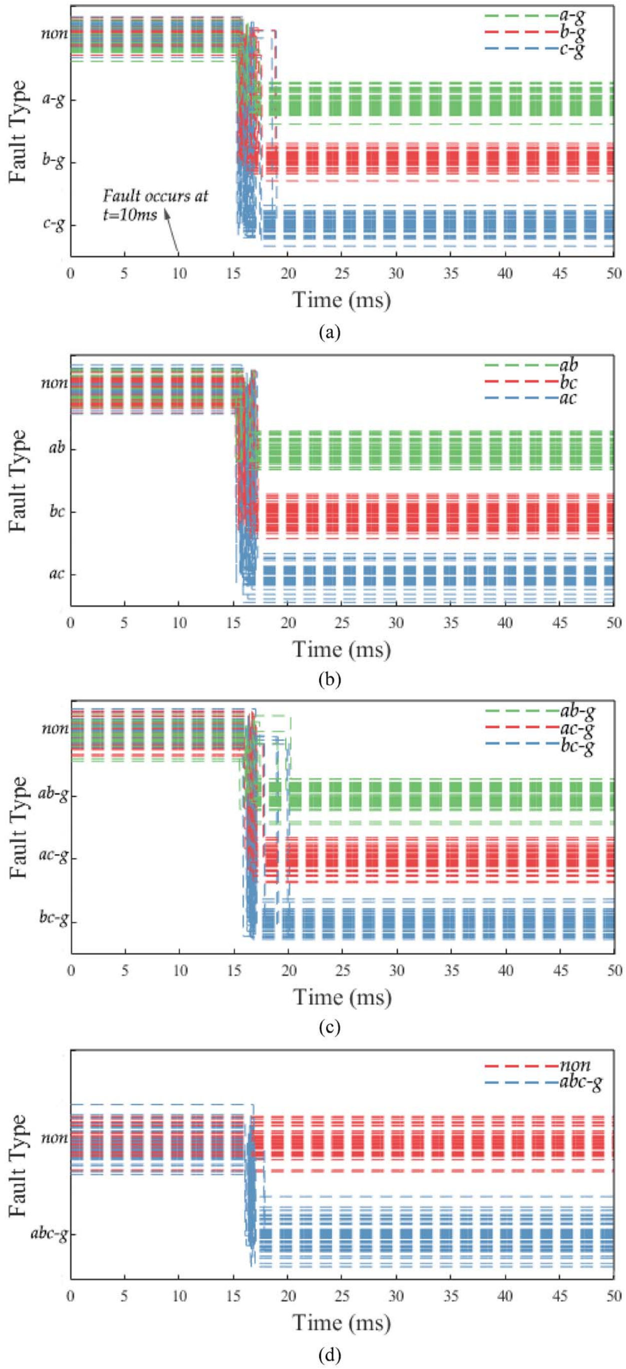
The performance of the proposed method for online fault detection and classification is shown in Fig. 5. We randomly select 50 multi-channel signal samples for each fault type and plot the system output for each signal sample within the time range between 0 ms and 50 ms. For clarity, the signals of four different fault categories are separately plotted in Fig. 5(a), Fig. 5(b), Fig. 5(c), and Fig. 5(d). The "non-faulty" signals are also plotted in Fig. 5(d). The fault inception time is 10 ms, and the system output for each signal sample is expected to be "non" before the fault is detected. To clearly display the system output for each signal sample, we shift all the outputs slightly up or down. The constants added to the outputs obey normal distribution $N(0, 0.01)$ given the difference between any two neighboring fault types on the vertical axis is 1. As the filtering operation is applied to the system outputs, we can see that all signal samples are correctly classified into the 11 fault types once fault detection is securely done. No mistake is observed even in the time range between 30 ms to 50 ms, for which no training data is provided. Further, as is seen in Fig. 5, the fault detection time for all fault types are basically between 5 and 10 ms. Considering the fact that we employ the strategy to classify signal segments whose post-fault signal proportion is lower than 40% as "non-faulty", such response speed is quite satisfactory.

The average time used for fault detection of the online test dataset (200 signals for each fault type) is listed in Table II. As is depicted in the table, the average time used for fault detection is between 6 ms and 7 ms. Generally speaking, the average detection time for ab-g, ac-g, bc-g and abc-g faults is longer than other fault types. The "non-faulty" type is not listed in this table, as the system output of this type is expected to remain unchanged.

To validate the performance of the softmax classifier, the classification accuracy for different fault types is also calculated and depicted in Table III. As mentioned previously, the training dataset contains 45000 multi-channel signal segments. The test dataset used here is also a subset of the complete test dataset with 83229 multi-channel signal segments, which is generated by randomly taking out 20000 segments from the complete dataset. The overall classification accuracy is 99.74%, and the classification accuracies for all 11 types are higher than 99.29%. This result shows that the proposed method is capable of classifying faults with quite high accuracies.

B. *The Effect of Sampling Frequency and Signal Type*

The previous results are obtained when the sampling frequency of the multi-channel signals is 20 kHz. Under the restriction of the data acquisition equipment, the maximum sampling frequency can be much lower than 20 kHz in practice. Further, in cases where voltage and current signals are not available at the same time, we may only use the voltage signals



(a)

(b)

(c)

(d)

Fig. 5. Examples of output signals of the fault detection and classification system for different types of faults. Each fault type has fifty examples whose original outputs are added with different small biases for the purpose of illustration.

TABLE II
Average Time of Fault Detection for Different Fault Types

Fault Type	Average Time of Fault Detection (ms)
a-g	6.22
b-g	6.38
c-g	6.42
ab	6.24
ac	6.58
bc	6.24
ab-g	6.96
ac-g	6.65
bc-g	6.71
abc-g	6.70

TABLE III
Classification Accuracy of the Proposed Method for Different Fault Types

Fault Type	Classification Accuracy (%)
a-g	99.94
b-g	99.29
c-g	99.77
ab	99.88
ac	99.88
bc	99.94
ab-g	99.31
ac-g	99.70
bc-g	99.62
abc-g	99.82
non-faulty	100.00

or the current signals. Thus, we use the same training dataset and test dataset to examine how the proposed method perform with different sampling frequencies and signal types. In addition to the scheme in which both voltage and current signals are used, two other schemes using only the voltage signals or the current signals are implemented. The sampling frequencies used are 1.25 kHz, 2.5 kHz, 5 kHz, 10 kHz, and 20 kHz, respectively. Each classification accuracy result is obtained by repeating the implementation for 5 times, and the mean values of the classification accuracies are then calculated. As different sampling frequencies are used, the window length l_W and patch length l_P are accordingly modified so that the time durations corresponding to l_W and l_P remain the same to the maximum extent (as these parameters has to be integers, we have to round them up or down if necessary). The values of window length l_W for the sampling frequencies (1.25 kHz to 20 kHz) are 13, 25, 50, 100, and 200, whilst the values of patch length l_P are 2, 4, 8, 15, and 30, respectively. Moreover, the values for the pooling size s_p for different frequencies are set to 1, 2, 3, 4, and 5 in order that the pooled convolved feature vectors are comparable in length (note that the pooling size also has to be integers). As a result, the input sizes for the softmax classifier are 1200, 1100, 1400, 2100, and 3400, given that the number of hidden nodes is 100.

The results of classification accuracy with different sampling frequencies and signal types are shown in Fig. 6. First of all, the classification accuracy increases as sampling frequency increases for all 3 schemes. This is in line with the expectation, as a higher sampling frequency provides more information with respect to the specific fault type. What also meets our expectation is that the scheme using both voltage and current signals (Scheme I) has the highest accuracies for all sampling frequencies. Specifically, at the lower sampling frequencies such as 1.25 kHz and 2.5 kHz, the advantage of scheme I over scheme II and scheme III is more significant compared to higher sampling frequencies. Further, it is noticed

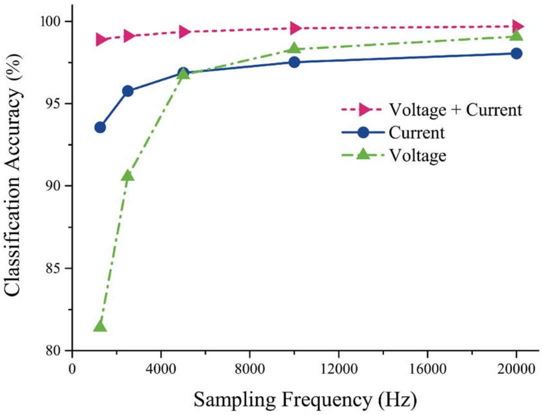


Fig. 6. The classification accuracies with different sampling frequencies for scheme I (voltage and current signals), scheme II (current signals only) and scheme III (voltage signals only).

that the performance of scheme II is better than scheme III at lower sampling frequencies, whereas at higher frequencies scheme III has higher accuracies. At 5 kHz, both schemes have roughly the same performance. One explanation for this is that the current signals contain more low-frequency information about the specific fault type than the voltage signals, while the voltage signals contain more fault-induced transients that are helpful to reveal the fault type. When both signals are used, high classification accuracy can be achieved across the frequency range we consider, as both aspects of fault type-specific information can be fully used. It is worth noticing that as we try to keep some of the parameters consistent for all 3 schemes to ensure that the results are comparable, the performance of scheme II and scheme III may not represent their optimal capability. Nevertheless, we can see that the classification accuracy of scheme I is more than 98% even at lower sampling frequencies, which validates the effectiveness of the proposed method.

C. Performance of the Proposed Method With Noise and Measurement Errors

It is of great significance to ensure that the method used to detect and classify faults can withstand noise and measurement errors. In addition, the generalizability of the feature extracting process needs verification. In this light, a new dataset of voltage and current signals corresponding to different fault types are simulated using PSCAD/EMTDC. Noise and measurement errors are separately considered in the first place, and we then compare the performance of the proposed method with some existing methods in the presence of both noise and measurement errors.

For the simulation setup in PSCAD/EMTDC, a transmission line model similar to the model in Fig. 1 is used. The transmission line adopts the frequency dependent phase-domain model, which is theoretically the most accurate model as the frequency dependence of internal transformation matrices can be represented [39]. In addition, the three phases of the transmission line are untransposed, as opposed to the ideally transposed line used for the model simulated in MATLAB/Simulink. Concretely, the impedance of the sources at both ends is $Z_s = 9.19 + 74.76\ \Omega$, and the transmission line has positive sequence impedance $Z_1 = 5.38 + 84.55\ \Omega$ and zero sequence impedance $Z_0 = 64.82 + 209.52\ \Omega$. The parameters used for simulation are listed in Table IV. Each fault type has 540 simulation results, thus there are 5400 simulation results (the "non-faulty" type is not considered here). Further, 59400 waveform segments are generated (11 different time ranges) and are divided into a training set (41580 segments, 70%) and a test set (17820 segments, 30%). We now refer to the previous dataset as dataset I and refer to the dataset introduced in this section as dataset II. In the following implementations of the proposed method, we use dataset I to extract the features and train the softmax classifier with dataset II. It should be noted that the feature extraction stage is unsupervised as the emphasis of this stage is to learn useful and universal feature representations for the convolutional operations. Thus, with more data available, the effectiveness of this stage is expected to be improved. As the dataset used for training the softmax classifier has to be labeled, it is acceptable if the size of the dataset is relatively small, as has been discussed in [29].

TABLE IV
PARAMETERS USED FOR THE SIMULATION IN PSCAD/EMTDC TO GENERATE DATASET II

System Parameter	Values
Fault Distance (km)	50, 70, 90, 110, 130, 150
Fault Resistance (Ω)	0.01, 5, 15, 30, 50
Fault Inception Angle (degrees)	0, 60, 120, 180, 240, 300
Pre-fault Power Angle (degrees)	10, 20, 30

Denoising Sparse Autoencoder (DSAE) is implemented to enhance the performance of the proposed method by extracting noise-tolerant features. In [40], additive Gaussian noise is used to extract useful robust feature representations and the denoising mechanism is discussed extensively. In this paper, we corrupt the input data of DSAE with white Gaussian noise (WGN) to produce data with a specific signal to noise ratio (SNR) and train the DSAE capable of reconstructing the uncorrupted data. More specifically, we corrupt the input data $\boldsymbol{x}$ with WGN and produce $\widetilde{\boldsymbol{x}}$. The reconstruction of $\boldsymbol{x}$, $h(\widetilde{\boldsymbol{x}})$, is then calculated using (1). Consequently, the squared error term $\sum_{i=1}^{m} \|h(\boldsymbol{x}^{(i)}) - \boldsymbol{x}^{(i)}\|^2$ in (2) is replaced by $\sum_{i=1}^{m} \|h(\widetilde{\boldsymbol{x}}^{(i)}) - \boldsymbol{x}^{(i)}\|^2$. We then use the features extracted by DSAE for the convolutional operations. Comparison of classification accuracies of features extracted by SAE and DSAE is presented in Fig. 7. We can see that the classification accuracies of DSAE-based implementations are above 98%, while those of SAE-based implementations drop much faster as SNR decreases. This result indicates that feature extraction by DSAE greatly reduces the impact of WGN, given that we have prior knowledge of the SNR of the signals collected.

Further, two types of representative measurement errors are considered in this paper. The first type of error (type I error) is "consecutive zero", i.e., a section of the signal becomes zero. The second type of error (type II error) is "consecutive high value", which refers to the phenomenon in which the signal rises to a high value (either positive or negative) and keeps the value for a little while. To cope with measurement errors,

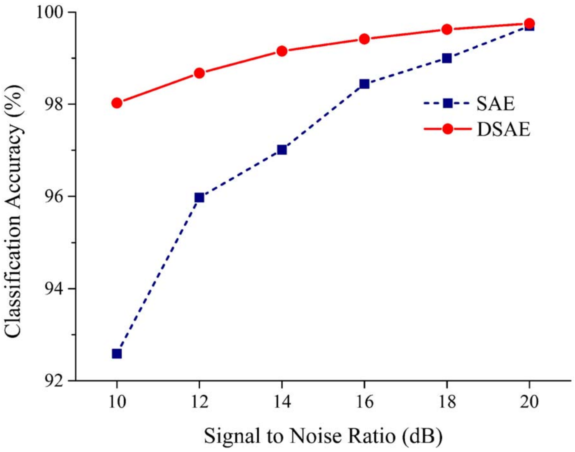


Fig. 7. The classification accuracies using features extracted by SAE and DSAE under different SNR values.

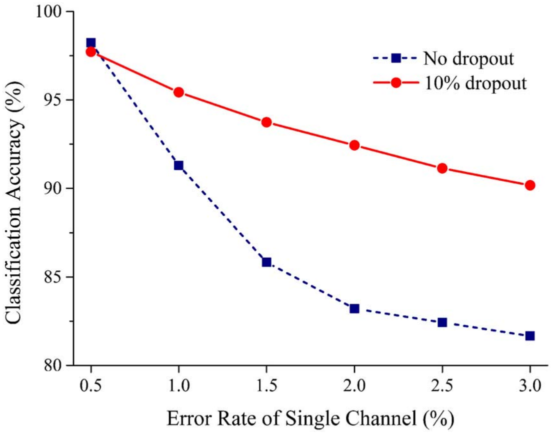


Fig. 8. The classification accuracies using features extracted by SAE with and without dropout under different error rates.

dropout is added during the training process of SAE. Dropout has been proven useful to prevent neural networks from overfitting [41], which means that models trained with dropout are more tolerant to errors within the data. Concretely, dropout is implemented by randomly setting activations of hidden nodes to 0 at a given probability. In this paper, we apply dropout to the connections between the input layer and the hidden layer, that is, we replace (1) with

$$h(\boldsymbol{x}) = \boldsymbol{W}_2\big(\boldsymbol{r}^{\mathrm{T}} f(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1)\big) + \boldsymbol{b}_2 \tag{12}$$

where $\boldsymbol{r}$ is the dropout masking vector whose jth element r_j satisfies $r_j \sim \text{Bernoulli}(1 - p_d)$, p_d being the probability for a given connection to be masked by dropout. Comparison of classification accuracies of training SAE with and without dropout is shown in Fig. 8. The value of p_d is set to 0.1 after trying several different values. Each waveform segment has exactly one error randomly added to one of its six channels, and both type I and II errors account for 50% of the errors. For type II error, the high value is set to two times of the rated amplitude of the signal (errors with positive and negative high values have the same proportion). The range of error rate (the proportion of error in a single channel) is 0.5% to 3%. It is clear from Fig. 8 that a dropout rate of 10% ($p_d = 0.1$) greatly improves the classification accuracies of the proposed method when measurement errors are taken into consideration.

We now combine the implementation of both DSAE and dropout and compare the proposed method with existing methods that have been proven effective in fault classification. For clarity, the proposed method with the implementation of DSAE and dropout is referred to as CDSAE (convolutional DSAE), which is essentially a slightly modified version of the original CSAE method. The SNR of signals with additive WGN is set to 14, and the error rate is 1.5% (3 consecutive sampling points with the sampling frequency of 20 kHz). The features used for convolutional operations are extracted by a DSAE with 10% dropout rate using dataset I, while the softmax classifier is trained and tested using dataset II. All signals in the datasets are corrupted with WGN, resulting in an SNR of 14. Other parameters remain the same as introduced in Section III. We also use DWT to extract features for fault classification. Concretely, we decompose the waveforms into eight detail levels and one approximate level (with the sampling frequency of 20 kHz, the approximate level covers 0-78.125 Hz) using Db4 mother wavelet. Energy (summation of squared coefficients) [42] and maximum (the maximum of absolute values of the coefficients) [43] features are calculated from the coefficients in each decomposition level. For classification, SVM and ANN are used with both types of features as inputs. The implementation of SVM with RBF kernel is introduced in [20]. Appropriate values of parameters γ and C are determined using 10-fold cross-validation and grid search. The structure of the ANNs has one hidden layer with fully connected neurons. With regard to energy features, $\gamma = 0.05$ and $C = 20$ are used for the SVM and $\lambda = 0.001$ is used for the 54-200-11 ANN (λ is the weight decay parameter of the ANN model for regularization). As for maximum features, $\gamma = 0.01$ and $C = 20$ are used for the SVM and $\lambda = 0.001$ is used for the ANN with the same structure.

The results of classification accuracies are listed in Table V. It is demonstrated in the table that the proposed CDSAE method outperforms the other methods. With the implementation of DSAE with dropout, the performance of the proposed method is satisfactory in the presence of noise and measurement errors. Also note that the features used for convolutional operations for dataset II are extracted from dataset I, which confirms the generalizability of the proposed method for transmission line systems that are similar in configuration but have different parameters and system dynamics. Though the features extracted by DSAE with dropout are inevitably different from the ones extracted by SAE without dropout, the generalizability is still verified, as both implementations only use dataset I during the feature extraction stage.

V. Application of the Proposed Method in Smart Grids

An illustrative diagram of implementing the proposed method in power systems is shown in Fig. 9. With intelligent electronic devices such as remote terminal units installed at the terminals of substations in the monitored region [44],

TABLE V
COMPARISON OF CLASSIFICATION ACCURACIES ON DATASET II OF DIFFERENT METHODS

Method	Classification Accuracy (%)
CDSAE	**92.22**
DWT energy + SVM	87.67
DWT energy + ANN	91.60
DWT maximum + SVM	86.55
DWT maximum + ANN	86.11

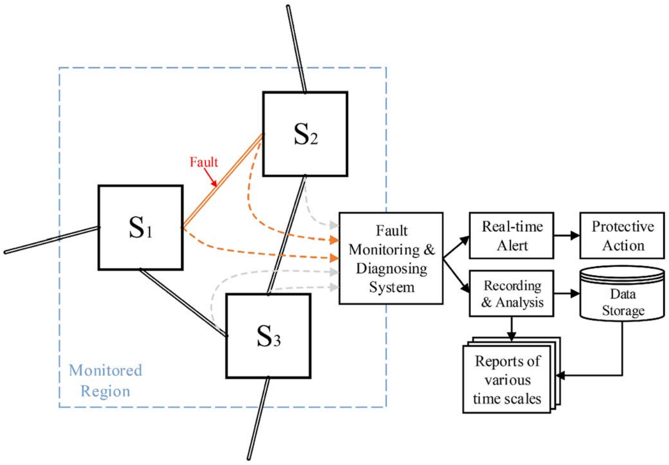


Fig. 9. Application of the proposed method in smart grids.

a fault monitoring and diagnosing system based on the proposed method can be established. The system is able to give real-time alerts at the moment of fault occurrence, and protective actions can be taken if possible. The high generalizability of the proposed method means that it can be widely adopted by power transmission systems as well as power distribution systems.

In addition, data recording and preliminary data analysis run continuously and store the data and analysis results in the database. Reports on a daily, weekly, monthly, seasonal and yearly basis can be generated via further analysis using stored data, which may help grid operators assess the reliability of the grid in the monitored region and evaluate the necessity of transforming or upgrading the grid infrastructure. Further analysis can be done with the integration of fault locating methods.

VI. CONCLUSION

This paper presents a new method for detection and classification of power transmission line faults. Three-phase voltage and current signals are combined as a multi-channel signal, and unsupervised feature learning from a dataset of signal segments is achieved by SAE. A CSAE based framework capable of detecting and classifying faults is proposed with novel training and testing strategies, which greatly reduces the computational burden and improves the performance. Results show that the proposed method detects faults within 7 ms after fault inception, and classifies faults with accuracies close to 100% for all fault types. Tests with different sampling frequencies and signal types show that using both voltage and current signals guarantees favorable performance across the considered frequency range. The proposed method is further modified to ensure that the performance is favorable in the presence of noise and measurement errors. Comparison of the proposed method with existing methods show that the proposed method is robust to noise and measurement errors with high generalizability.

As we use simulated data for training and testing, future implementation of the method may consider using real data collected by various devices deployed in the power grid. Moreover, in cases where only voltage or current signals are available, the parameters such as window length and patch length need to be tuned so that the classification accuracy can be improved. In order to build a more comprehensive fault diagnosis system capable of detecting and classifying faults and power quality disturbances, the framework needs further modification in window and patch lengths as well as the overall structure.

REFERENCES

[1] K. Chen, C. Huang, and J. L. He, "Fault detection, classification and location for transmission lines and distribution systems: A review on the methods," *High Voltage*, vol. 1, no. 1, pp. 25–33, 2016.

[2] S.-L. Yu and J.-C. Gu, "Removal of decaying DC in current and voltage signals using a modified Fourier filter algorithm," *IEEE Trans. Power Del.*, vol. 16, no. 3, pp. 372–379, Jul. 2001.

[3] M. T. Hagh, K. Razi, and H. Taghizadeh, "Fault classification and location of power transmission lines using artificial neural network," in *Proc. Int. Power Eng. Conf. (IPEC)*, Singapore, 2007, pp. 1109–1114.

[4] B. Das and J. V. Reddy, "Fuzzy-logic-based fault classification scheme for digital distance protection," *IEEE Trans. Power Del.*, vol. 20, no. 2, pp. 609–616, Apr. 2005.

[5] 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.

[6] A. I. Megahed, A. M. Moussa, and A. E. Bayoumy, "Usage of wavelet transform in the protection of series-compensated transmission lines," *IEEE Trans. Power Del.*, vol. 21, no. 3, pp. 1213–1221, Jul. 2006.

[7] 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.

[8] S. Ekici, S. Yildirim, and M. Poyraz, "Energy and entropy-based feature extraction for locating fault on transmission lines by using neural network and wavelet packet decomposition," *Expert Syst. Appl.*, vol. 34, no. 4, pp. 2937–2944, 2008.

[9] S. R. Samantaray and P. K. Dash, "Pattern recognition based digital relaying for advanced series compensated line," *Int. J. Elect. Power Energy Syst.*, vol. 30, no. 2, pp. 102–112, 2008.

[10] K. R. Krishnanand and P. K. Dash, "A new real-time fast discrete S-transform for cross-differential protection of shunt-compensated power systems," *IEEE Trans. Power Del.*, vol. 28, no. 1, pp. 402–410, Jan. 2013.

[11] J.-A. Jiang, C.-S. Chen, and C.-W. Liu, "A new protection scheme for fault detection, direction discrimination, classification, and location in transmission lines," *IEEE Trans. Power Del.*, vol. 18, no. 1, pp. 34–42, Jan. 2003.

[12] X. Dong, W. Kong, and T. Cui, "Fault classification and faulted-phase selection based on the initial current traveling wave," *IEEE Trans. Power Del.*, vol. 24, no. 2, pp. 552–559, Apr. 2009.

[13] D. Thukaram, H. P. Khincha, and H. P. Vijaynarasimha, "Artificial neural network and support vector machine approach for locating faults in radial distribution systems," *IEEE Trans. Power Del.*, vol. 20, no. 2, pp. 710–721, Apr. 2005.

[14] J.-A. Jiang *et al.*, "A hybrid framework for fault detection, classification, and location—Part I: Concept, structure, and methodology," *IEEE Trans. Power Del.*, vol. 26, no. 3, pp. 1988–1998, Jul. 2011.

[15] A. A. Girgis, A. A. Sallam, and A. K. El-Din, "An adaptive protection scheme for advanced series compensated (ASC) transmission lines," *IEEE Trans. Power Del.*, vol. 13, no. 2, pp. 414–420, Apr. 1998.

[16] O. A. S. Youssef, "Fault classification based on wavelet transforms," in *Proc. IEEE/PES Transm. Distrib. Conf. Expo.*, vol. 1. Atlanta, GA, USA, 2001, pp. 531–536.

[17] R. N. Mahanty and P. B. D. Gupta, "Application of RBF neural network to fault classification and location in transmission lines," *IEE Proc. Gener. Transm. Distrib.*, vol. 151, no. 2, pp. 201–212, Mar. 2004.

[18] J. Upendar, C. P. Gupta, and G. K. Singh, "Discrete wavelet transform and probabilistic neural network based algorithm for classification of fault on transmission systems," in *Proc. Annu. IEEE India Conf. (INDICON)*, vol. 1. Kanpur, India, 2008, pp. 206–211.

[19] P. K. Dash, S. R. Samantaray, and G. Panda, "Fault classification and section identification of an advanced series-compensated transmission line using support vector machine," *IEEE Trans. Power Del.*, vol. 22, no. 1, pp. 67–73, Jan. 2007.

[20] U. B. Parikh, B. Das, and R. Maheshwari, "Fault classification technique for series compensated transmission line using support vector machine," *Int. J. Elect. Power Energy Syst.*, vol. 32, no. 6, pp. 629–636, 2010.

[21] M. J. Reddy and D. K. Mohanta, "Adaptive-neuro-fuzzy inference system approach for transmission line fault classification and location incorporating effects of power swings," *IET Gener. Transm. Distrib.*, vol. 2, no. 2, pp. 235–244, Mar. 2008.

[22] A. Jamehbozorg and S. M. Shahrtash, "A decision tree-based method for fault classification in double-circuit transmission lines," *IEEE Trans. Power Del.*, vol. 25, no. 4, pp. 2184–2189, Oct. 2010.

[23] M. K. Jena, L. N. Tripathy, and S. R. Samantray, "Intelligent relaying of UPFC based transmission lines using decision tree," in *Proc. 1st Int. Conf. Emerg. Trends Appl. Comput. Sci. (ICETACS)*, Shillong, India, 2013, pp. 224–229.

[24] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.

[26] A. Hannun *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. Int. Conf. Learn. Represent. Workshop*, Scottsdale, AZ, USA, 2013, pp. 1301–3781.

[28] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Vancouver, BC, Canada, 2013, pp. 8595–8598.

[29] K. Chen, J. Hu, and J. L. He, "A framework for automatically extracting overvoltage features based on sparse autoencoder," *IEEE Trans. Smart Grid*, to be published, doi: 10.1109/TSG.2016.2558200.

[30] R. Zhang, C. Li, and D. Jia, "A new multi-channels sequence recognition framework using deep convolutional neural network," in *Proc. INNS Conf. Big Data Program*, vol. 53. San Francisco, CA, USA, Aug. 2015, pp. 383–390.

[31] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Spatio-temporal convolutional sparse auto-encoder for sequence classification," in *Proc. BMVC*, Surrey, U.K., 2012, pp. 1–12.

[32] A. Ng, "Sparse autoencoder," CS294A Lecture Notes, Stanford Univ., Stanford, CA, USA, pp. 1–19, 2011.

[33] J. Ngiam *et al.*, "On optimization methods for deep learning," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, Bellevue, WA, USA, 2011, pp. 265–272.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cogn. Model.*, vol. 5, no. 3, p. 1, 1988.

[35] A. J. Bell and T. J. Sejnowski, "The 'independent components' of natural scenes are edge filters," *Vis. Res.*, vol. 37, no. 23, pp. 3327–3338, 1997.

[36] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[37] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997.

[38] C. Hung, J. Nieto, Z. Taylor, J. Underwood, and S. Sukkarieh, "Orchard fruit segmentation using multi-spectral feature learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, 2013, pp. 5314–5320.

[39] M. Z. Daud, P. Ciufo, and S. Perera, "Investigation on the suitability of PSCAD/EMTDC models to study energisation transients of 132 kV underground cables," in *Proc. Aust. Universities Power Eng. Conf. (AUPEC)*, Sydney, NSW, Australia, 2008, pp. 1–6.

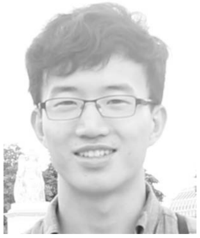
[40] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[41] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jun. 2014.

[42] Z. Q. Bo, R. K. Aggarwal, A. T. Johns, H. Y. Li, and Y. H. Song, "A new approach to phase selection using fault generated high frequency noise and neural networks," *IEEE Trans. Power Del.*, vol. 12, no. 1, pp. 106–115, Jan. 1997.

[43] 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.

[44] M. Kezunovic, "Smart fault location for smart grids," *IEEE Trans. Smart Grid*, vol. 2, no. 1, pp. 11–22, Mar. 2011.



Kunjin Chen was born in Changsha, China, in 1993. He received the B.Sc. degree in electrical engineering from Tsinghua University, Beijing, China, in 2015, where he is currently pursuing the M.Sc. degree with the Department of Electrical Engineering. His research interests include pattern recognition and data mining in power systems.

Jun Hu was born in Ningbo, China, in 1976. He received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the Department of Electrical Engineering, Tsinghua University, Beijing, China, in 1998, 2000, and 2008.

He is currently an Associate Professor with the Department of Electrical Engineering, Tsinghua University. His research fields include overvoltage analysis in power system, sensors and big data, dielectric materials, and surge arrester technology.

Jinliang He (M'02–SM'02–F'08) was born in Changsha, China, in 1966. He received the B.Sc. degree in electrical engineering from the Wuhan University of Hydraulic and Electrical Engineering, Wuhan, China, in 1988, the M.Sc. degree in electrical engineering from Chongqing University, Chongqing, China, in 1991, and the Ph.D. degree in electrical engineering from Tsinghua University, Beijing, China, in 1994.

He became a Lecturer in 1994, and an Associate Professor in 1996, with the Department of Electrical Engineering, Tsinghua University. From 1997 to 1998, he was a Visiting Scientist with Korea Electrotechnology Research Institute, Changwon, South Korea, involved in research on metal oxide varistors and high voltage polymeric metal oxide surge arresters. From 2014 to 2015, he was a Visiting Professor with the Department of Electrical Engineering, Stanford University, Palo Alto, CA, USA. In 2001, he was promoted to a Professor with Tsinghua University. He is currently the Chair with High Voltage Research Institute, Tsinghua University. He has authored five books and 400 technical papers. His research interests include overvoltages and EMC in power systems and electronic systems, lightning protection, grounding technology, power apparatus, and dielectric material.