IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 May 2024, accepted 23 May 2024, date of publication 27 May 2024, date of current version 4 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3406164

RESEARCH ARTICLE

A Temporal Ensembling Based Semi-Supervised Graph Convolutional Network for Power Quality Disturbances Classification

JIAJUN CAI, HUAIZHI WANG, (Member, IEEE), AND HUI JIANG, (Member, IEEE)

Key Laboratory of Optoelectronic Devices and Systems of Ministry of Education and Guangdong Province, College of Physics and Optoelectronic Engineering, Shenzhen University, Shenzhen 518060, China

Corresponding author: Hui Jiang (huijiang@szu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 52177102.

ABSTRACT With the integration of multiple energy sources into the power grid makes power quality disturbances (PQDs) more complex. Dealing with power quality problems requires automatic classification of PQDs. This paper proposes a novel semi-supervised Graph Convolutional Network (GCN) framework based on Temporal Ensembling for PQDs classification. Considering both short-term and long-term features of PQDs, a Visibility Graph (VG) based graph theory model was adopted to process PQDs to highlight features. In the proposed semi-supervised framework, Graph Convolutional Network was designed to extract features from massive PQDs and classify PQDs automatically. Due to the fact that GCN belongs to supervised learning, it is necessary to label the data in advance. However, labeling is costly and easily lead to human mistake. Therefore, this article introduces the Temporal Ensembling algorithm which provides pseudo labels to reduce the amount of labeled data and has tolerance to incorrect labels. Simulation results prove that the proposed method is capable of noise resistance, tolerates incorrect labels, and has high classification performance in both single and composite PQDs.

INDEX TERMS Power quality, graph convolutional network, temporal ensembling, visibility graph.

I. INTRODUCTION

With the integration of diverse renewable energy sources into the power grid and the increasing utilization of numerous power electronic devices within the network [1], there is a noticeable rise in voltage, current, and frequency fluctuations in power systems [2]. These fluctuations pose new challenges for the reliable and stable operation of the power grid. Additionally, as power consumption expands, both electricity consumers and electrical equipment require higher power quality. Consequently, power quality issues have gained greater attention [3]. The automated classification of PQDs is of crucial importance in addressing power quality problems [4].

In traditional approaches to classifying PQDs through artificial intelligence or machine learning, the process generally comprises three stages: feature extraction, feature selection, and feature classification. Sun et al. [5] effectively employed the wavelet transform (WT) to extract performance index features from the negative sequence components of PQDs, leveraging the capabilities of feedforward neural networks for accurate classification. Mahela et al. [6] utilized multi-resolution analysis based on the S transform (ST) to obtain diverse signal features, subsequently employing decision trees for efficient feature classification. Sahani and Dash [7] utilized Hilbert Transform (HT) to derive the characteristics of PQDs, proposing a novel class-specific weighted random vector functional link network classifier for the recognition of PQDs. Huang et al. [8] employed the short-time Fourier transform as a time-frequency analysis tool to examine various types of PQDs. They then introduced a Binary-SVM-based multi-label classification method to classify mixed PQDs. Additionally, Cai et al. [9] applied variational mode decomposition (VMD) to distinguish between stationary and non-stationary PQDs, extracting three statistical features from the instantaneous amplitude

The associate editor coordinating the review of this manuscript and approving it for publication was Alon Kuperman.

of the decomposition mode. Furthermore, they developed a deep stochastic configuration network (DSCN) model for PQDs classification. Samanta et al. [10] proposed a down-sampled Kriging Interpolation (KI)-based empirical mode decomposition (EMD) to enhance the accuracy and speed of EMD operations. Subsequently, they employed the extreme learning machine (ELM) for the classification of PQDs, taking into account all the derived features through the KI-EMD approach. These methods demonstrate increased efficiency and accuracy compared to manual classification. However, with the explosive growth of data in smart grids and the increasingly complex problem of PQDs classification [11], the above methods no longer meet the needs as they separate feature acquisition, selection, and feature classification.

Deep learning (DL), which has recently emerged, can effectively address all the aforementioned problems. DL seamlessly integrates feature extraction and classification, automatically detecting relevant information and categorizing data, while continuously updating its parameters to optimize performance. DL is utilized in addressing virtually every kind of problem. More recently, DL has been frequently employed in tackling PQDs. Balouji and Salor [12] used PQDs' voltage images obtained from the power grid. Shukla et al. [13] used GAF (Gramian Angular Field) to convert PQDs series into an image. Both [12] and [13] classify the PQDs images by using a two-dimensional Convolutional Neural Networks (CNN). Wang and Chen [14] proposed a novel full closed-loop approach to detect and classify PQDs based on a deep convolutional neural network. Wang et al. [15] selected LSTM as the DL architecture to obtain the temporal features of the signal and achieve PQDs classification. Deng et al. [16] utilized Bi-GRU to extract deep features from PQDs and optimized the shortcomings of the GRU algorithm. The proposed methodology can determine the type of each element in the input sequence, which is different from the existing sequence-to-sequence model employing an encoder–decoder network. Wang et al. [17] constructed a parallel network of CNN-GRU, using CNN and GRU to extract short-term and long-term features from PQDs, respectively, and improve classification accuracy. Karim et al. [18] combined CNN and LSTM to extract PQDs features in both time and space, while incorporating SE modules within the CNN to improve training efficiency. In the aforementioned research, CNN excels at extracting short-term features, while RNN excels at extracting long-term features. To simultaneously capture both short-term and long-term features of PQDs, a combination network is required. However, employing both networks simultaneously would not only increase the network's running time but also raise the requirements for equipment.

The VG, originally proposed by Lacasa et al. [19], transforms time series data into a VG representation, by utilizing the graph's topological structure to depict both short-term and long-term features of time series. This approach effectively highlights significant structures and outliers within the sequence. Therefore, it can be used to convert PQDs data to obtain features. However, VG is graph structured data that requires Graph Neural Network (GNN) to obtain its features. In the realm of deep learning, the GCN, introduced by Kipf and Welling [20] in 2016, has become a focal point for research. GCN excels at extracting structural features from graphs. Zheng et al. [21] used GCN to predict traffic flow with the support of historical traffic flow datasets. This method can accurately estimate the future traffic flow for specific time periods and regions. Chen et al. [22] applied GCN to analyze the compound structures, determining their toxicity and predicting toxicity characteristics based on extensive chemical databases. Zheng et al. [23] transformed PQDs from one-dimensional time series into two-dimensional density maps using Gramian Angular Field (GAF). Subsequently, they employed GCN for the classification and recognition of PQDs. However, the data graphs he used were all square mesh graphs composed of pixel points, which were universal for different PQDs and did not fully leverage the advantages of GCN, thus hindering further learning of the network model.

Methods for classifying PQDs primarily rely on supervised learning [12], [13], [14], [15], [16], [17], [18], which requires labeled data throughout the dataset. However, in practical production scenarios, labeling data incurs significant costs. In response to this challenge, researchers have turned their attention to semi-supervised learning. The concept of Temporal Ensembling, introduced by Laine and Aila [24], revolves around self-assembly principles. This algorithm strategically introduces pseudo-labels to unlabeled data, filling in the gaps left by missing real labels and alleviating the demand for fully labeled datasets. Meel and Vishwakarma [25] applied Temporal Ensembling to detect false news and addresses the significant increase in the speed and quantity of false news generated on social media, resulting in time-consuming and expensive data labeling tasks. Ding et al. [26] utilized Temporal Ensembling for detecting COVID-19 lesions, resolving issues related to challenging data acquisition and insufficient annotated data during emergency epidemic situations. Shi et al. [27] employed Temporal Ensembling in tissue pathology image analysis to tackle scenarios with a large number of unlabeled sample images. Temporal Ensembling proves effective not only in handling data with insufficient labeling, but also in addressing potential incorrect labels that may arise during manual labeling processes.

Based on the above analysis, this paper proposes a novel GCN semi-supervised framework based on Temporal Ensembling for PQDs classification. The main contributions of this article are as follows:

1) Convert the large voltage time series containing PQDs information into a VG, which reflects short-term and long-term features of PQDs by showcasing connection modes and relationships between nodes, thereby highlighting significant structures and outliers within the

data. Meanwhile, the feature input of the node adopts a feature matrix composed of One-Hot Encoding and sampled values. This matrix represents the sampling order and values of the sampling points corresponding to the nodes in VG. Subsequently, employ a multi-layer stacked GCN to extract feature information from the graph structure.

2) Temporal Ensembling algorithm is introduced for the classification of PQDs, integrating both supervised and unsupervised learning to facilitate the training of datasets with a limited number of labeled samples, thereby reducing expenses associated with manual data labeling. Simultaneously, it demonstrates robust tolerance for incorrect labels in the dataset, minimizing the impact of such errors. Additionally, the algorithm exhibits resistance to noise.

II. THE METHODS

A. VISIBILITY GRAPH

VG is an algorithm that converts time series into a graph. Through calculations, it represents the short-term and long-term features of the time series by utilizing the topological structure of the graph. This structure includes information such as the connection mode and relationships between nodes, as well as the length of edges. The algorithm enables the application of graph theory tools to characterize time series, facilitating connections between time series analysis, nonlinear dynamics, and graph theory. VG has the following advantages:

1) Important Structural Discovery: In VG, crucial structural characteristics may manifest as distinct shapes, connection methods, or clustering patterns compared to typical nodes. Examples include periodicity, trends, and other distinctive structures. These unique formations often signify important information or features within the dataset, facilitating a deeper understanding of the inherent correlations and structures within it.

2) Outlier Identification: Outliers in VG may manifest as nodes with different connection patterns from surrounding nodes or as nodes occupying special positions in the graph. These outliers can represent abnormal behavior or unusual data points within the data. Observing these distinctive nodes in the graph makes it easier to identify atypical behaviors in the dataset.

The construction of the VG involves representing a time series with bars, where the height of each bar is determined by the corresponding value in the time series. If the line connecting two bars is unobstructed by other bars, forming what is referred to as a visibility line, the following visibility criteria can be established: There are two points (t_i, y_i) and (t_j, y_j). If i and j are adjacent points, visibility between these two points must be satisfied. However, if there is another point (t_k, y_k) between i and j, the specific conditions for visibility that must be satisfied by i and j are illustrated in Figure 1.

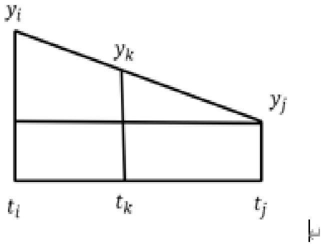


FIGURE 1. Visibility diagram.

According to the similarity principle of triangles, the values in the figure should meet the (1):

$$\frac{y_k - y_j}{y_i - y_j} = \frac{t_j - t_k}{t_j - t_i} \tag{1}$$

It means that k-th point should satisfy the following inequality [27]:

$$y_k < y_j + \left(y_i - y_j\right) * \frac{t_j - t_k}{t_j - t_i} \tag{2}$$

There is visibility between points i and j, that is, there is no vertical bar between i and j that blocks the line of sight between the two points. Figure 2 and Figure 3 are examples of one of the PQDs (swell) represented as a bar chart and converted into a VG:

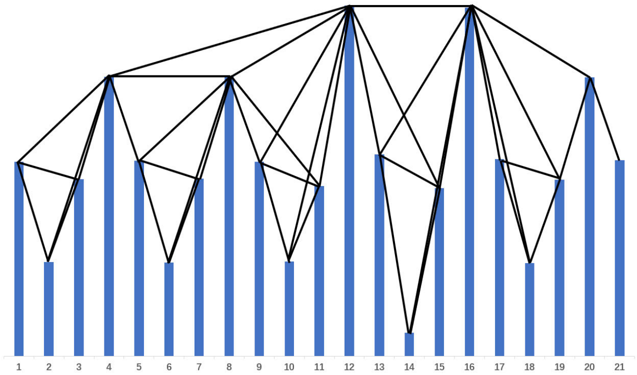


FIGURE 2. Bars of swell.

Points 12 and 16 serve as special nodes in the temporary elevation, where they can see more points and occupy a special position in the transformed VG. They can gather more information and are also a topological feature of the graph. This can identify swell in various PQDs. The VG converted from other PQDs is shown in Figures 4 and 5. Figure 4 shows the single PQDs, and Figure 5 shows the composite PQDs. The outliers of PQDs will appear as special nodes in VG, and the characteristics of each disturbance will be presented in VG with a special topological structure.

B. NODE FEATURE

The input of GCN includes the topological structure of the graph and the feature information of nodes. The One-Hot Encoding is added as the feature value of the node. It is essentially an identity matrix that can be used to record the temporal order of each node in the original PQDs and as

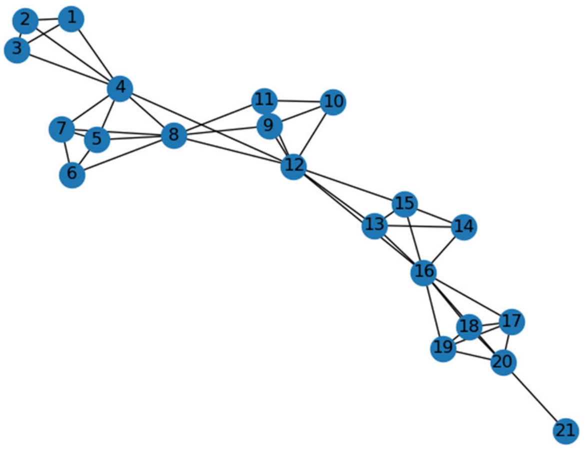


FIGURE 3. VG image converted from Figure 2.

one of the temporal characteristics of PQDs. An identity matrix $I^{n\times n}$ is employed as a one-hot encoding matrix, where n is the length of the time series and the number of nodes in the graph. Additionally, the values of the sampling points p_i are represented as matrix $\boldsymbol{P} = \begin{bmatrix} p_1 \\ p_2 \\ \dots \\ p_{n-1} \\ p_n \end{bmatrix}$ in node order and are combined to form a feature matrix $\boldsymbol{X} = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 & p_1 \\ 0 & 1 & \dots & 0 & 0 & p_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 & p_{n-1} \\ 0 & 0 & \dots & 0 & 1 & p_n \end{bmatrix}$, serving as the feature input for nodes

C. GRAPH CONVOLUTIONAL NETWORK

GNN is a neural network model specifically designed for modeling graph data. It seamlessly integrates deep learning with graph broadcasting operation algorithms and leverages both node feature and graph structure information in the neural network learning process. GNN has demonstrated excellent feasibility and interpretability in deep learning applications, particularly in graph analysis. GCN stands out as the most widely used type of GNN. Given the feature matrix X and adjacency matrix A, GCN performs spectral convolution operations by considering nodes and their first-order neighbors in the graph, as opposed to the convolution operations in CNN, to capture the spatial features of the graph. Additionally, GCN applies hierarchical propagation rules across multiple layers of the network, further enhancing its representation ability.

Graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{X})$ is input of GCN, where $\boldsymbol{V} = (\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3}, \dots, \boldsymbol{v_n})$ is the collection of nodes in the graph, $\boldsymbol{E}$ is the collection of edges in the graph. $\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \\ \dots \\ \boldsymbol{x_3} \\ \boldsymbol{x_n} \end{pmatrix} \in \boldsymbol{R}^{n\times m}$ is feature matrix, where $\boldsymbol{x_i}$ as the feature vector of $\boldsymbol{v_i}$, m is the number of features of node. Define adjacency matrix $\boldsymbol{A} \in \boldsymbol{R}^{n\times n}$, If there are edges between nodes $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$, then $\boldsymbol{A_{ij}} = 1$, otherwise $\boldsymbol{A_{ij}} = 0$.

Graph convolution is a technique that utilizes a local first-order approximation to spectral graph convolution. The propagation of node features within the hidden layer of a specific layer, denoted as layer l, is determined by the following formula.:

$$\boldsymbol{X}^{(l+1)} = \sigma(\widetilde{\boldsymbol{A}}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}) \tag{3}$$

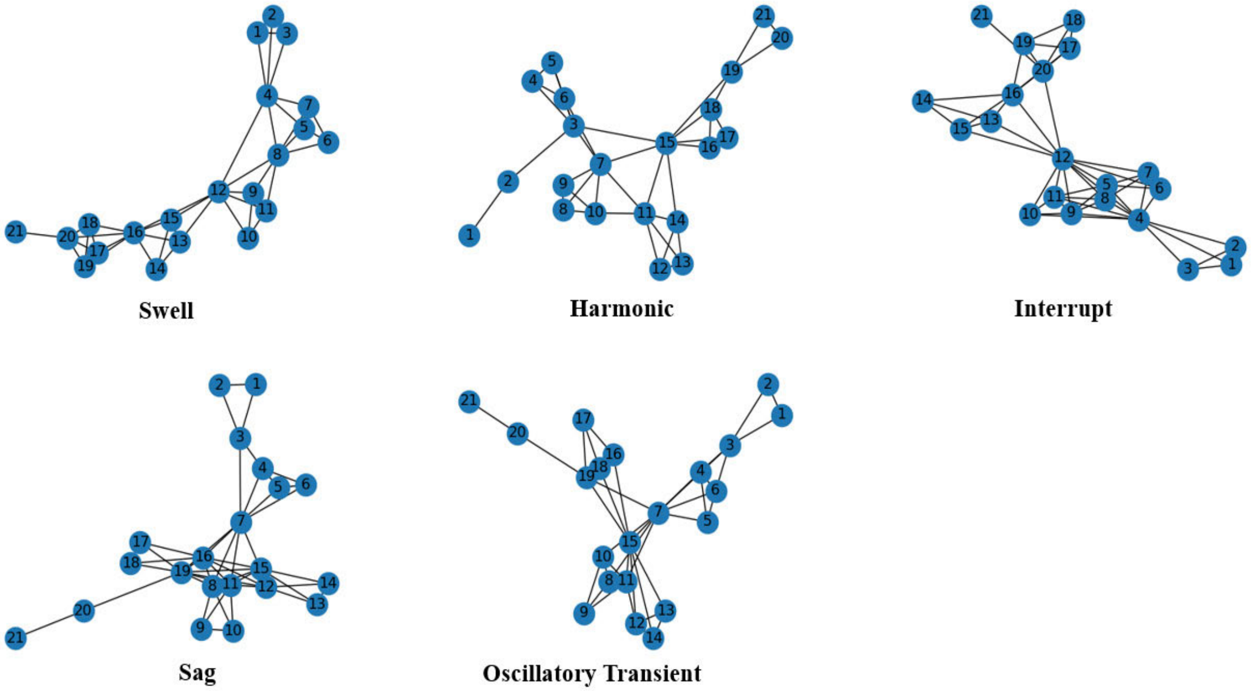
where $\boldsymbol{X}^{(l+1)}$ is output of l-th layer, $\boldsymbol{X}^{(l)}$ is input of l-th layer, $\boldsymbol{W}^{(l)} \in \boldsymbol{R}^{m^{(l)}\times m^{(l+1)}}$ is a learnable parameter matrix determining the size of the l-th hidden layer node feature matrix, $\sigma(\cdot)$ is nonlinear activation function. $\widetilde{\boldsymbol{A}} = \widehat{\boldsymbol{D}}^{-\frac{1}{2}}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{D}}^{-\frac{1}{2}}$ is a symmetric normalized adjacency matrix, where $\widehat{\boldsymbol{A}}$,the self-loop adjacency matrix, is obtained by adding the identity matrix $\boldsymbol{I}$ to $\boldsymbol{A}$, thus introducing self-loops for each node. During the convolutional layer's information aggregation, it includes the information of the node itself. $\widehat{\boldsymbol{D}} = diag(d_1, d_2, d_3, \dots, d_n)$ is a degree matrix, where $d_i = \sum_j^n \widehat{\boldsymbol{A}}_{ij}$.

After convolution, the feature vector of the i-th node is denoted as $\boldsymbol{x}_i^{(l+1)}(x_{i,1}^{(l+1)}, x_{i,2}^{(l+1)}, x_{i,3}^{(l+1)}, \dots, x_{i,n}^{(l+1)})$, where $x_{i,j}^{(l+1)}$ is as follow:

$$x_{i,j}^{(l+1)} = \sum\nolimits_{k=1}^{n} \widetilde{\boldsymbol{A}}_{ik}\boldsymbol{X}_{kj}^{(l)} \tag{4}$$

$x_{i,j}^{(l+1)}$ is the j-th feature value of the i-th node after convolution. If there is no edge between two nodes ($\widetilde{\boldsymbol{A}}_{ik} = 0$), the information of the k-th node is not aggregated to the i-th node after convolution. If $\widetilde{\boldsymbol{A}}_{ik} \neq 0$, indicating an edge between the two nodes, node features are weighted and summed based on the values in the normalized Laplacian matrix $\widetilde{\boldsymbol{A}}$. According to matrix multiplication and formula (3), $\boldsymbol{X}^{(l+1)} \in \boldsymbol{R}^{n\times m^{(l+1)}}$, therefore, the parameter matrix $\boldsymbol{W}^{(l)}$ not only influences the network's training process to approach the training target, but also affects the feature output of nodes to meet the network requirements. The graph features extracted by the network are represented by the feature values of nodes. Base on formulas (3) and (4): the computational complexity of GCN is calculated as O($nnm^l + nm^l m^{l+1}$) based on the computational complexity of matrix multiplication, where n is the number of nodes, m^l is the number of features per node in the l-th layer GCN, m^{l+1} is the number of features per node in the l + 1-th layer GCN and the hidden layer channel size of the l-th layer GCN. The number of hidden layers can be set by oneself, and the input node features only participate in the calculation in the first layer of GCN. Therefore, the main factor affecting the computational complexity of GCN is the number of nodes. The more nodes there are, the more complex the calculation becomes.

In summary, GCN leverages spectral graph convolution to update node features. The convolution process involves



Swell

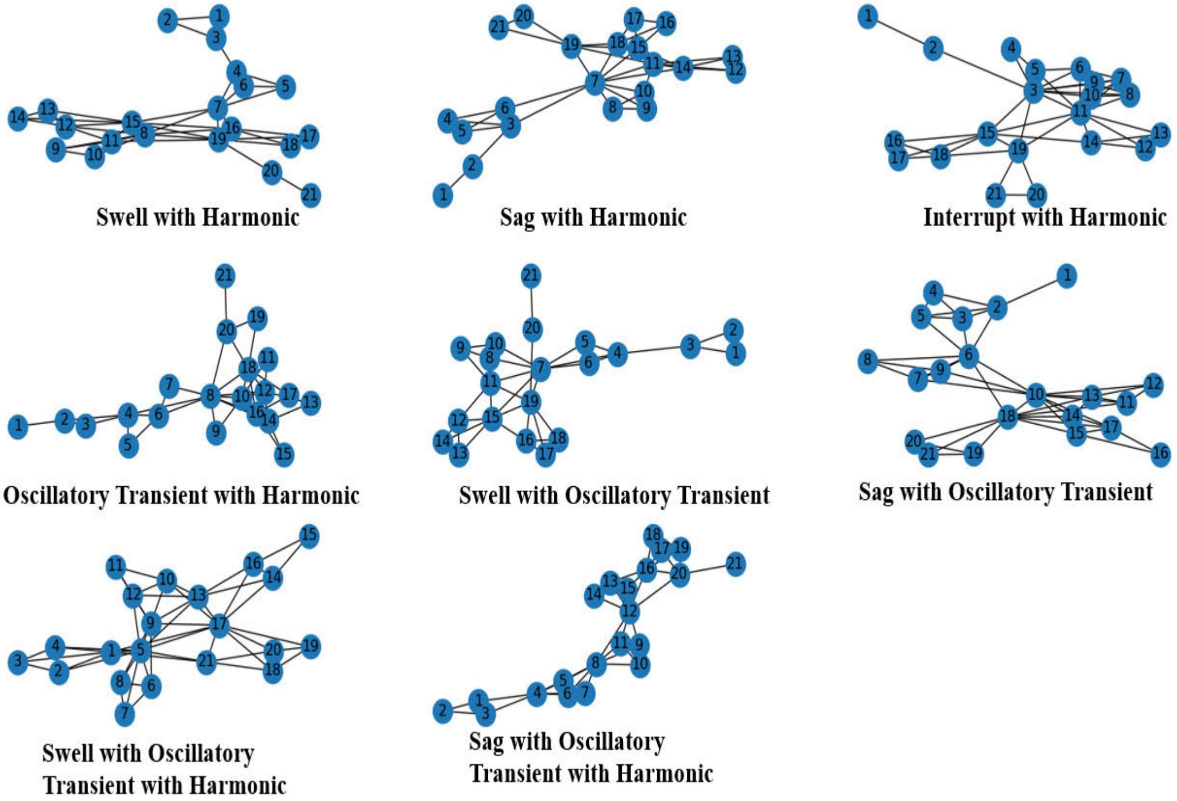
Harmonic

Interrupt

Sag

Oscillatory Transient

FIGURE 4. VG of single PQDs.



Swell with Harmonic

Sag with Harmonic

Interrupt with Harmonic

Oscillatory Transient with Harmonic

Swell with Oscillatory Transient

Sag with Oscillatory Transient

Swell with Oscillatory Transient with Harmonic

Sag with Oscillatory Transient with Harmonic

FIGURE 5. VG of composite PQDs.

aggregating information from neighboring nodes based on the normalized Laplacian matrix $\widetilde{A}$. The parameter matrix W influences both the network training outcomes and the feature outputs of nodes, determining the ability to effectively capture spatial features within the graph. Through this process, GCN excel in extracting meaningful graph features, representing them through the updated feature values of individual nodes.

D. TEMPORAL ENSEMBLING ALGORITHM

Temporal Ensembling is a semi-supervised learning algorithm that combines supervised and unsupervised learning. It can effectively train datasets with limited labeled data and a large amount of unlabeled data, thereby reducing the cost of manually labeling data. Meanwhile, Temporal Ensembling can also handle incorrect labels in the dataset, reducing the impact of such labels.

Temporal Ensembling runs as shown in Algorithm 1:

Algorithm 1

```
Parameter Initialization:
$x_i$ = training data
$L$ = set of training input with known labels
$y_i$ = labels for labeled inputs $x_i$, $i \in L$
$w(t)$ = unsupervised weight ramp-up function
$\alpha$ = ensemble momentum, $0 \leq \alpha \leq 1$
$f_\theta(x)$ = Training parameters of neural networks
$T$ = num epochs
$\boldsymbol{Z} \leftarrow 0_{[N\times C]}$ Δ initialize ensemble predictions
$\tilde{z} \leftarrow 0_{[N\times C]}$ Δ initialize unsupervised target vectors
 for t in [1, T] do
   for each minibatch $B$ do
     $z_i \leftarrow f_{\boldsymbol{\theta}}(x_{i\in B})$ Δ output of neural network
     $Loss_l \leftarrow -\frac{1}{|B|}\sum_{i\in(B\cap L)} \log z_i[y_i]$ Δ supervised loss
     $Loss_u \leftarrow \frac{1}{C|B|}\sum_{i\in B} ||z_i - \tilde{z}_i||^2$ Δ unsupervised loss
     $Loss_{total} \leftarrow Loss_l + w(t) * Loss_u$ Δ total loss
     update $\boldsymbol{\theta}$ Δ update network parameters
 end for
$\boldsymbol{Z} \leftarrow \alpha * \boldsymbol{Z} + (1-\alpha) * z$ Δ update temporal ensemble
$\tilde{z} \leftarrow \frac{\boldsymbol{Z}}{1-\alpha^t}$ Δ bias correction
end for
return $\boldsymbol{\theta}$
```

Temporal Ensembling divides its training data into labeled and unlabeled sets. Throughout the training process, the algorithm computes supervised loss for the labeled data and unsupervised loss for all inputs. At the conclusion of each epoch, the algorithm aggregates the predictions (represented as z) for all current data. It then updates and outputs a temporary ensemble, $\boldsymbol{Z}$, based on the formula (5):

$$\boldsymbol{Z} = \alpha * \boldsymbol{Z} + (1 - \alpha) * z \tag{5}$$

where α is the weight parameter that controls the historical ensemble $\boldsymbol{Z}$ and the current epoch's ensemble z. During the initial training, both $\boldsymbol{Z}$ and z were initialized as zero tensors due to the absence of a historical ensemble from the previous one. After the first epoch of training, $\boldsymbol{Z} = (1 - \alpha) * z$. Using formula (6) to correct the startup deviation:

$$\tilde{z} = \frac{\boldsymbol{Z}}{1 - \alpha^{\boldsymbol{t}}} \tag{6}$$

Therefore, Z is an ensemble representing the weighted average of algorithm outputs from earlier epochs, giving greater weight to the recent epoch's outputs compared to those from distant epochs. t represents the current training epoch. The supervised loss function is denoted as $Loss_l$(7), which calculates cross-entropy [28] for the output of labeled samples and their labels. B represents the dataset of the smallest batch in each training iteration, while L denotes the set of labeled datasets within the entire dataset. The unsupervised loss function, denoted as $Loss_u$(8), calculates the mean square error between the output of the current epoch and the historical ensemble output. C is the number of classifications. The total loss function (9) is the weighted sum of supervised and unsupervised loss functions:

$$Loss_l = CrossEntropy\,(z_i, y_i) = -\frac{1}{|B|}\sum_{i\in(B\cap L)} \log z_i[y_i] \tag{7}$$

$$Loss_u = MSE\,(z_i, Z_i) = \frac{1}{C\,|B|}\sum_{i\in B} ||z_i - \tilde{z}_i||^2 \tag{8}$$

$$Loss_{total} = Loss_l + w(t) * Loss_u \tag{9}$$

The unsupervised component of the loss function is weighted by the time function $w(t)$, which increases along the Gaussian curve from zero, as shown in formula (10):

$$w(t) = \exp[-5 * (1 - \frac{t}{T})] \tag{10}$$

where T is the number of epochs when the unsupervised loss reaches its maximum weight. $w(t)$ increases the proportion of unsupervised components as the training iterates. Due to $w(t)$, the supervised loss factor dominates in the early stages of training, thereby influencing both the learning gradient and the overall loss of the framework. In the later stages, unsupervised loss factors become more significant, ultimately enabling the achievement of semi-supervised learning.

III. POWER QUALITY DISTURBANCE CLASSIFICATION BASED ON GCN WITH TEMPORAL ENSEMBLING

This paper constructs a semi-supervised learning framework as shown in Figure 6 to classify PQDs. Firstly, the collected PQDs are labeled as needed, with the unlabeled PQDs are labeled as −1. The PQDs undergo VG conversion and one-hot encoding before being input into the GCN for feature extraction and classification. The GCN consists of five parts. The first three parts are composed of a graph convolutional layer and a normalization layer (BN) [29], stacked together. The graph convolutional layer is used to extract the features of the graph, and these extracted features are then normalized through the BN layer to improve the efficiency of network training. The activation function of the GCN is the Rectified Linear Unit (ReLU) [30], which can prevent gradient explosion and network overfitting. The latter two parts comprise the global pooling layer and the Linear layer. Through a global pooling layer, the data in each

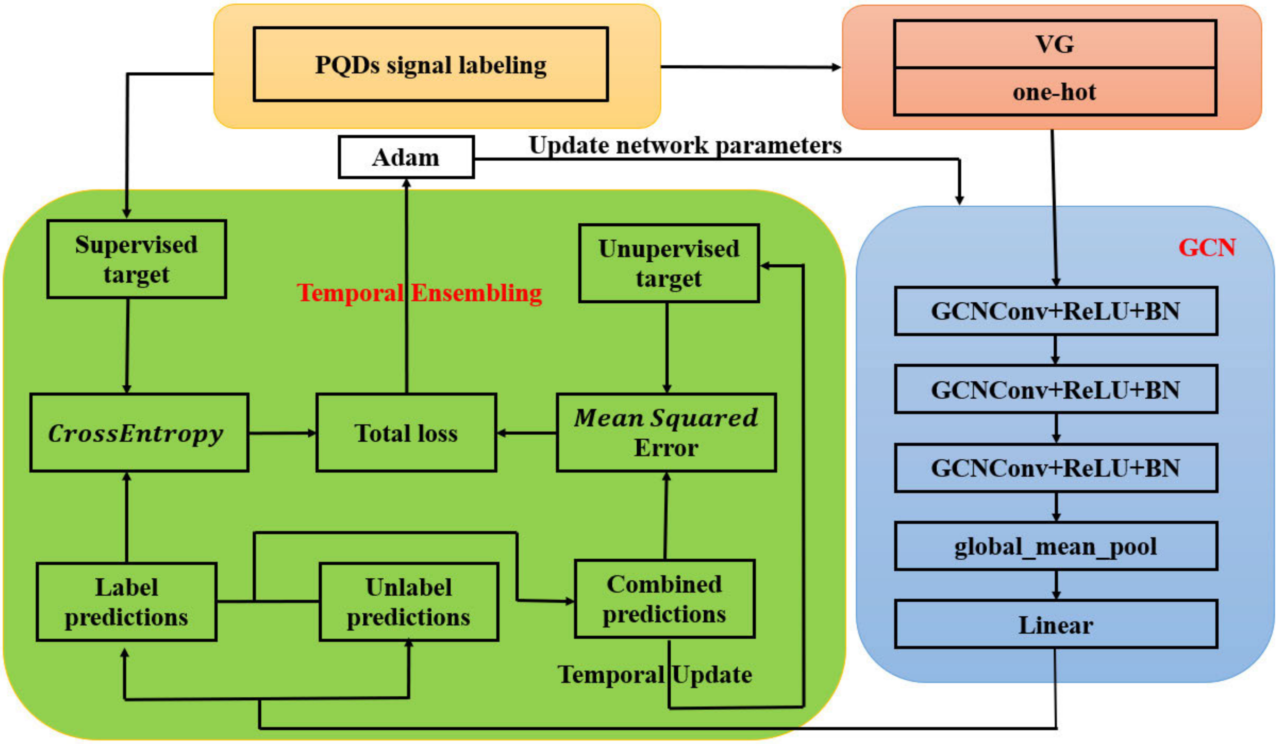


FIGURE 6. Proposed Semi-Supervised Learning Framework.

column of the node feature matrix is pooled. By summing or averaging multiple features of a node, the N (number of nodes) × M (hidden layer channels) matrix is transformed into the 1×M matrix. This transformation enables the linear layer to predict PQDs. The output of GCN is input to Temporal Ensembling and subjected to loss calculation. The supervised loss function calculates the loss based on the predictions of labeled PQDs. All PQDs predictions will be used to calculate unsupervised loss while updating the historical ensemble of Temporal Ensembling. The two types of losses are added by weight to form the final loss of the framework, and the optimizer optimizes GCN based on it.

The hidden layer channel size of each GCN layer is set to 64, and the "esp" parameter and attenuation coefficient momentum in each BN layer is set to 1×10^{-5} and 0.1. The global pooling layer used global_mean_pool. The input parameter for the Linear layer is 64 (the number of hidden layers in the graph convolutional layer), and the output is 13 (the number of types of PQDs). T is set to 20. The supervised loss function adopts the cross-entropy function, and the unsupervised loss function adopts the mean square error function. The optimizer selected is Adam [31], with the learning rate set to 5×10^{-3}. The weight coefficient α set to 0.6.

IV. SIMULATION AND ANALYSIS

A. SIGNAL GENERATION OF PQDS

Due to the strong randomness and complexity of PQDs in distribution networks, there are differences between the data simulated by MATLAB and the measured signals. Therefore, we have integrated a PQDs generation system with signal generation and data acquisition functions. The NI USB-6259 serves as the signal-generating unit, while the NI ELVIS II functions as the data acquisition unit, as illustrated in Figure 7. The front panel of the experimental system is depicted in Figure 8. By programming in LabView, we control the NI USB-6259 to simulate the required PQDs and then output them to the NI ELVIS II for signal acquisition. 13 types of PQDs are generation, comprising single PQDs (Swell, Sag, Harmonic, Interrupt, Oscillatory Transient) and composite PQDs (Swell with Harmonic, Sag with Harmonic, Interrupt with Harmonic, Oscillatory Transient with Harmonic, Swell with Oscillatory Transient, Sag with Oscillatory Transient, Swell with Oscillatory Transient with Harmonic, Sag with Oscillatory Transient with Harmonic). Using MATLAB and generative systems to generate half of the data respectively, in order to improve the generalization ability of the proposed framework. Each PQDs has 160 sampling points, and each type of PQDs collects the same number, with 13000 as the training set, 1300 as the validation set, and 13000 as the testing set.

B. SIMULATION AND ANALYSIS OF PQDS

The computer configuration used for simulation are Intel(R) Core (TM) i5-8300H CPU @ 2.30GH 2.30 GHZ, 16GB DDR4 RAM, NVIDIA Geforce GTX 1050Ti.

Before verifying the advantages of the proposed network, we compared GCN with traditional networks such as CNN,

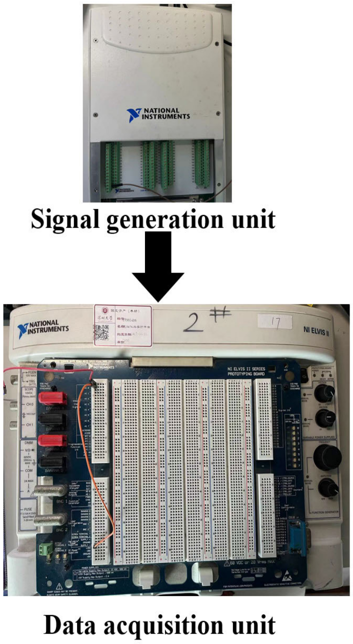
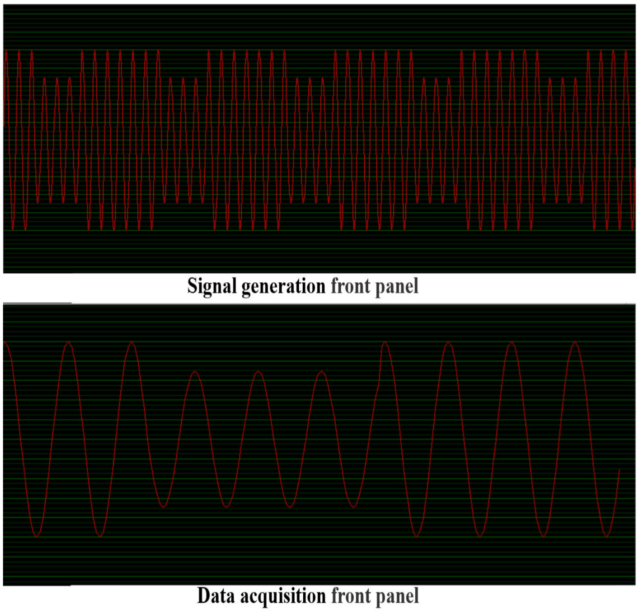


FIGURE 7. Experimental hardware framework.


Signal generation front panel

Data acquisition front panel

FIGURE 8. Experimental software frame.

LSTM, and GRU. The simulation results, which is shown in Table 1, suggest that both GCN and CNN exhibit higher classification accuracy than LSTM and GRU in the absence of noise, with rates of 99.9%, 98.5%, 97.5%, and 97.3%, respectively. In the signal-to-noise ratio of 30dB and 40dB noise, GCN shows better classification performance than traditional deep learning networks, and surface GCN has strong noise resistance. The confusion matrix of GCN is shown in Figure 9 and the types of PQDs in the matrix are given in the order shown in Table 3. The matrix shows that GCN exhibits excellent classification ability in both single and composite PQDs. Therefore, In the case where the VG graph effectively captures the important structures and outliers of PQD as different graph features, GCN can maximize its network performance and perform well in PQD classification.

TABLE 1. Classification accuracy of different methods in different SNRs.

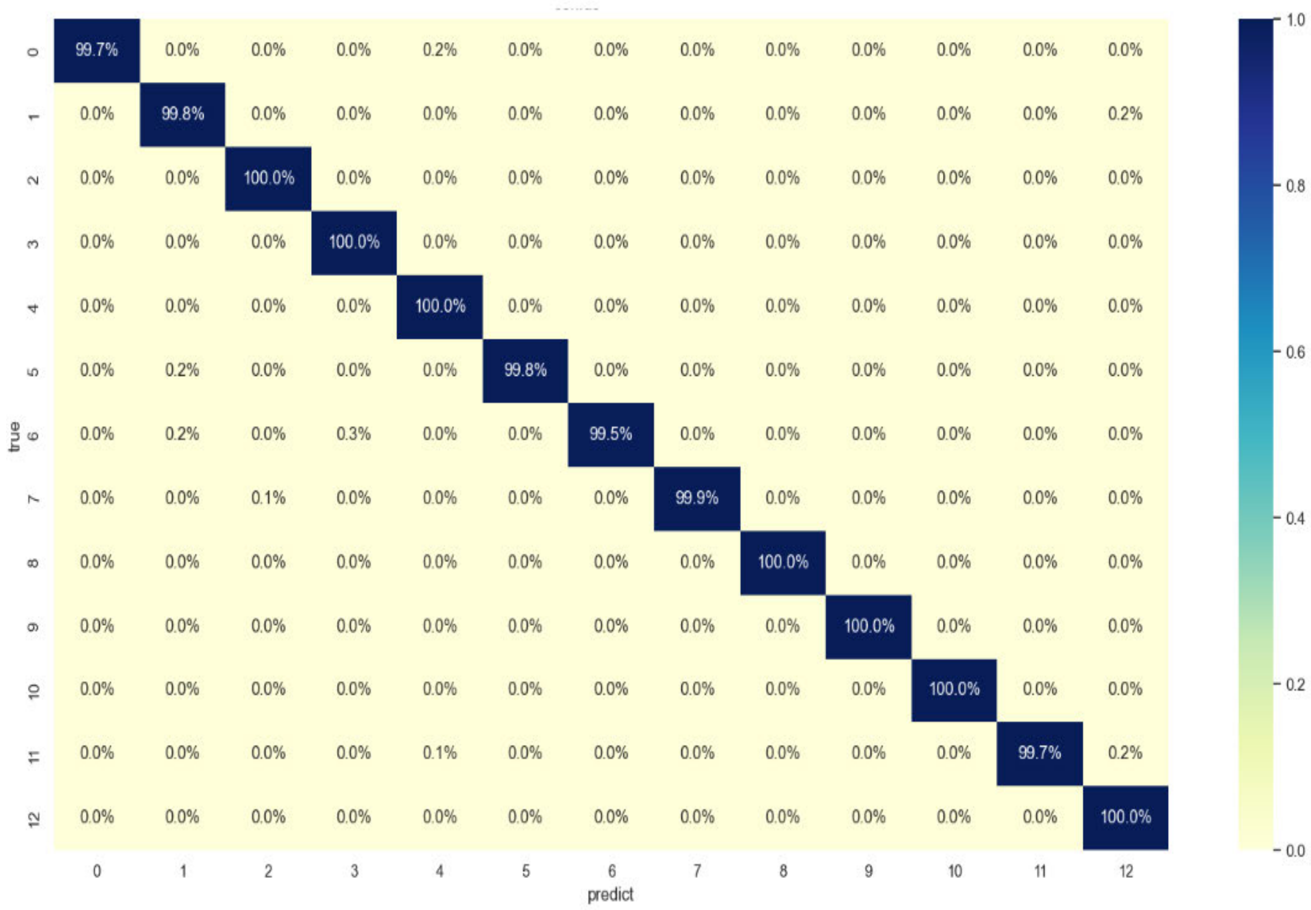
Methods	No noise	30dB	40dB
GCN	0.999	0.993	0.993
CNN	0.985	0.982	0.984
LSTM	0.975	0.953	0.961
GRU	0.973	0.960	0.966

GCN was utilized as the main network. We also compared Temporal Ensembling with others semi-learning algorithms, such as Π-model and mean-teacher. The simulation results, which is shown in Table 2, suggest that Temporal Ensembling and other two algorithms exhibit all have higher classification accuracy in the absence of noise, with rates of 99.4%, 98.1% and 99.2%, respectively. In the signal-to-noise ratio of 30dB and 40dB noise, Temporal Ensembling shows better classification performance than other semi-learning algorithms. In addition, Π-model requires the use of two identical network structures to handle labeled and unlabeled data separately and mean-teacher also needs to use a teacher model and a student model, which means they need to make two predictions each time they train, while Temporal Ensembling only makes one prediction each time they train. Temporal Ensembling is faster in training speed than the other two algorithms.

TABLE 2. Classification accuracy of semi-learning algorithms in different SNRs.

Algorithms	No noise	30dB	40dB
Temporal Ensembling	0.994	0.994	0.993
Π-model	0.981	0.977	0.976
Mean-teacher	0.992	0.989	0.991

Based on the above, GCN was utilized as the main network and Temporal Ensembling as the semi-supervised learning algorithm. Initially, the simulation results of GCN and GCN with Temporal Ensembling were compared. Subsequently, GCN with Temporal Ensembling was simulated under varying noise levels, amounts of labeled data, and amounts of incorrectly labeled data. The results of these simulations are presented in Table 3. Notably, in all scenarios, the simulations achieved exceptionally high classification accuracy, exceeding 99%, with the exception of cases involving a significant



true \ predict	0	1	2	3	4	5	6	7	8	9	10	11	12
0	99.7%	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
1	0.0%	99.8%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.2%
2	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
3	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
4	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
5	0.0%	0.2%	0.0%	0.0%	0.0%	99.8%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
6	0.0%	0.2%	0.0%	0.3%	0.0%	0.0%	99.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
7	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	99.9%	0.0%	0.0%	0.0%	0.0%	0.0%
8	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%
9	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%
10	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%
11	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	99.7%	0.2%
12	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%

FIGURE 9. Confusion matrix of GCN.

proportion of incorrect labels (80%), where the accuracy was 96.7%. A detailed simulation and analysis are presented in the following sections.

1) GCN AND GCN WITH TEMPORAL ENSEMBLING

Figure 10 compares the loss and accuracy during training of two different scenarios: GCN with all labeled data and GCN with Temporal Ensembling where only 10% of the data is labeled.

Both methods exhibit a rapid decrease in the loss function followed by gradual stabilization. When trained with fully labeled data, GCN achieved an accuracy of 99.8%. In contrast, GCN with Temporal Ensembling, which utilized only 10% of labeled data, reached a training accuracy of 99.6%. According to Table 3, the classification accuracy of GCN for PQDs is reported as 99.9%. On the other hand, with the reduced labeled dataset, GCN with Temporal Ensembling achieved a classification accuracy of 99.4%. While the simulation results indicate that GCN outperforms GCN with Temporal Ensembling in terms of accuracy, it is noteworthy that the latter relies on significantly fewer labeled data. Despite a mere 0.5% difference in accuracy, the ability to significantly reduce the amount of labeled data while sacrificing only a small amount of accuracy is deemed acceptable in practical production scenarios.

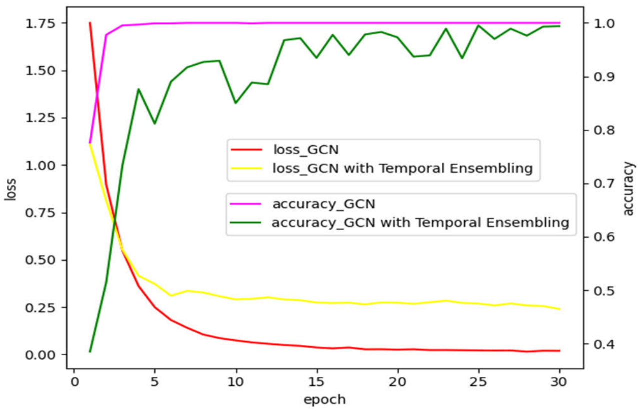


FIGURE 10. Classification accuracy and loss curves of GCN and GCN with Temporal Ensembling.

2) GCN WITH TEMPORAL ENSEMBLING IN DIFFERENT AMOUNT OF LABELED DATA

Adjusting the number of labeled data in the dataset for comparative simulation, the framework is trained using datasets where labeled data accounts for 5%, 10%, and 15%

TABLE 3. Classification accuracy of the proposed method.

PQDs	GCN	GCN with Temporal Ensembling							
		No noise			30dB 10% labeled	40dB 10% labeled	20% incorrectly labeled	50% incorrectly labeled	80% incorrectly labeled
		5% labeled	10% labeled	15% labeled					
Swell	0.997	0.956	0.994	0.997	1.000	0.992	1.000	0.903	0.828
Sag	0.999	0.999	0.999	1.000	1.000	0.999	1.000	0.999	1.000
Harmonic	0.998	0.969	0.979	0.986	0.985	0.969	0.998	0.987	0.975
Interrupt	1.000	0.983	0.997	0.992	1.000	0.997	0.999	1.000	0.989
Oscillatory Transient	1.000	0.967	0.996	0.985	0.998	0.995	1.000	0.976	0.979
Swell with Harmonic	0.999	0.979	0.989	0.987	0.977	0.990	0.997	0.974	0.999
Sag with Harmonic	1.000	0.998	0.984	0.991	0.991	0.980	0.993	0.995	0.964
Interrupt with Harmonic	1.000	0.990	0.999	0.998	0.991	0.988	0.994	0.945	0.987
Oscillatory Transient with Harmonic	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Swell with Oscillatory Transient	0.999	0.974	0.993	0.991	0.988	1.000	0.996	0.999	0.869
Sag with Oscillatory Transient	1.000	0.998	0.987	1.000	0.995	1.000	0.967	0.998	0.995
Swell with Oscillatory Transient with Harmonic	1.000	0.997	1.000	1.000	1.000	1.000	0.999	0.995	0.983
Sag with Oscillatory Transient with Harmonic	1.000	1.000	1.000	1.000	1.000	1.000	0.998	1.000	1.000
All	**0.999**	**0.990**	**0.994**	**0.994**	**0.994**	**0.993**	**0.995**	**0.982**	**0.967**

of the total data, respectively. The loss and accuracy during training are depicted in Figure 11.

When labeled data comprises 15%, 10%, and 5% of the total data, the training accuracy is 99.8%, 99.6%, and 98.5%, respectively, accompanied by a rapid decrease in loss. We then simulate the three trained frameworks using a test set; the classification results are shown in Table 3. The simulation results indicate that, as the proportion of labeled data decreases, the classification accuracy also decreases. Specifically, the accuracy is 99.0% when labeled data comprises 5%, while it is 99.4% when labeled data comprises both 10% and 15%. These frameworks exhibit high classification performance under both single and composite disturbances. Therefore, the proportion of labeled data can be selected for framework training according to actual needs. Although a large amount of labeled data ensures fast training efficiency and high classification accuracy, it also entails a corresponding increase in cost

3) GCN WITH TEMPORAL ENSEMBLING IN DIFFERENT NOISES

Train the framework using PQDs datasets without noise and with SNRs of 30dB and 40dB. Figure 12 shows the training process curves for these conditions.

The training accuracy without noise is 99.6%, while with SNRs of 30dB and 40dB, it is 99.4% and 99.3%, respectively, accompanied by a decrease in loss. Subsequently, we simulate the three trained frameworks using a test set, and the

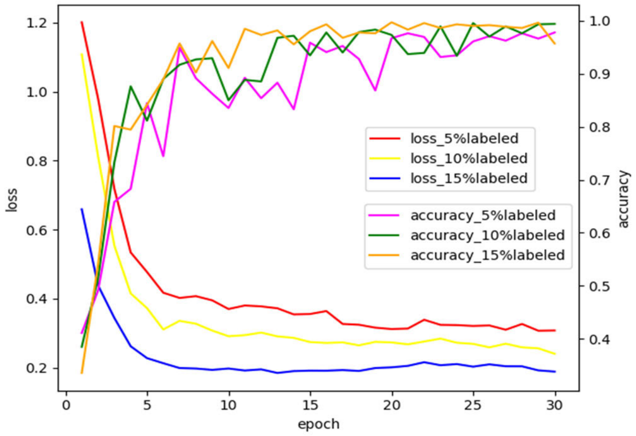


FIGURE 11. Classification accuracy and loss curves under different number of labeled data.

classification results are presented in Table 3. The simulation outcomes reveal that the classification accuracy is 99.4% in the absence of noise, and 99.3% and 99.4% at SNR levels of 40dB and 30dB, respectively. Furthermore, the framework demonstrates high classification performance under both single and composite disturbances, showcasing its robust noise resistance.

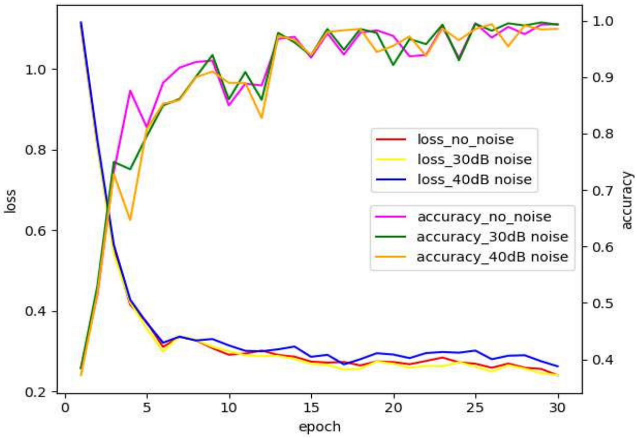


FIGURE 12. Classification accuracy and loss curves under different noise.

4) GCN WITH TEMPORAL ENSEMBLING IN DIFFERENT AMOUNT OF INCORRECT DATA

Temporal Ensembling has robustness against incorrect labels, indicating its ability to effectively cope with incorrect or noisy labels. With the help of Temporal Ensembling, GCN is trained using datasets with incorrect labels. The proposed framework is trained with datasets containing 20%, 50%, and 80% incorrect labels. T is set to 200, and the training process curves are illustrated in Figures 13 and 14.

The final training results revealed a training accuracy of 99.6% when 20% of the labels were incorrect, 98.5% with 50% incorrect labels, and 96.7% with 80% incorrect labels. The corresponding losses decreased to 0.105, 0.092, and 0.075, respectively. The simulation results for the proposed framework are presented in Table 3. When 20% of the labels were incorrect, the classification accuracy was 99.5%. With 50% incorrect labels, the accuracy was 98.2%, exhibiting relatively low performance on swell, specifically with a classification accuracy of 90%. When 80% of the labels were incorrect, the classification accuracy was 96.7%, but it performed poorly on swell and harmonic + swell, achieving classification accuracies of 82.8% and 86.9%, respectively.

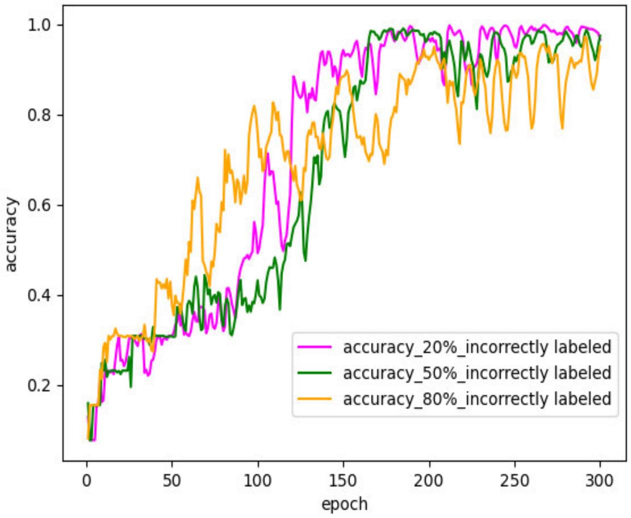


FIGURE 13. Classification accuracy curve under different number of incorrectly labeled data.

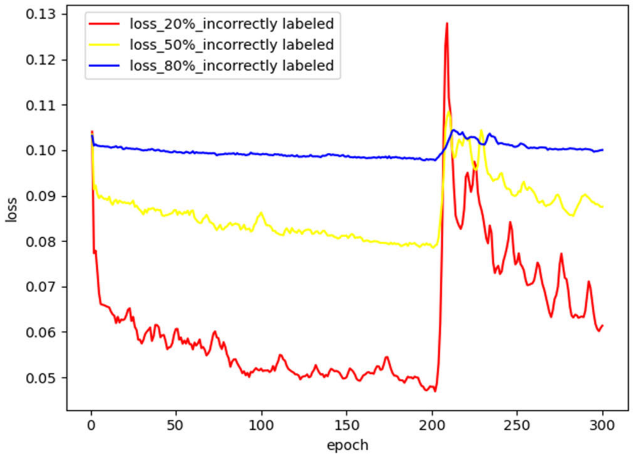


FIGURE 14. Loss curve under different number of incorrectly labeled data.

The figures illustrate that both the decrease in loss and the increase in training accuracy during the early stages of framework training are slow and fluctuate. The supervised loss term enforces the mapping function to have a specific value in the vicinity of labeled input data points. In the initial stages of framework training, the supervised loss function value constitutes a significant portion of the total loss value and the incorrect labels can mislead the supervised loss function, preventing the framework from training correctly. On the other hand, the unsupervised loss term encourages the framework's mapping function to be flat in the vicinity of all input data points. Sufficient correct labels in each class help anchor the clusters to the correct output vectors through the supervised loss term. As the framework progresses, the proportion of unsupervised loss functions increases that make the total loss function enables the optimizer to optimize network parameters towards accurate training results, effectively training the framework. However, incorrect supervised loss functions consistently impact the total loss function, causing the training process curve to fluctuate more noticeably as the number of incorrect labels increases.

The above results are acceptable when a large number of incorrect labels are used. The framework has demonstrated a strong resistance to incorrect labels, as evidenced by our results, allowing it to reduce the effort of human error.

V. CONCLUSION

This paper proposes a Temporal Ensembling based semi-supervised GCN for PQDs classification:

(1) By converting PQDs to VG, the short-term and long-term features f the time series are translated into the topological structure of the graph. The significant structures and outliers in PQDs are represented as characteristic connection methods and specific nodes in the graph, effectively addressing the challenges of feature extraction in PQDs. The features of the nodes are encoded using one-hot encoding, which can be used as one of the temporal features of PQDs. GCN is employed to extract feature information from the VG, encompassing both nodes and their relationships, successfully capturing complex patterns and structural information within the VG. Ultimately, this leads to the accurate classification of PQDs. Through comparative experiments with traditional deep learning networks such as CNN, LSTM, and GRU, it has been demonstrated that, with the assistance of VG, GCN exhibits stronger classification performance and noise resistance.

(2) Temporal Ensembling is used for semi-supervised learning. This algorithm uses the weighted sum of supervised and unsupervised learning loss functions as the loss output of the network framework, and optimizes the parameters of the GCN. It enables semi-supervised training on limited labeled PQDs, thereby reducing production costs. This algorithm can stabilize the training process of the model by introducing a temporal integration strategy. During the training process, the algorithm not only focuses on the current batch of samples, but also takes into account information from previous batches, thereby reducing training instability caused by data distribution differences between batches. the algorithm calculates the weighted sum of each output and historical set as pseudo labels for PQDs in the next round of unsupervised learning. This makes the Temporal Ensembling algorithm can better address these challenges and improve the robustness of the model when dealing with datasets with noise or inaccurate labels.

The proposed method combines the advantages of VG, GCN, and Temporal Ensembling mentioned above to achieve efficient classification of PQDs.

Future work on the classification of PQDs should center on the utilization of models. The proposed deep semi-supervised framework, being relatively lightweight and possessing swift computational speed, holds promise for deployment on embedded devices and integration into practical production.

ACKNOWLEDGMENT

The authors would like to thank the editors and anonymous reviewers who provided valuable input to this article.

REFERENCES

[1] Y. Zhan, H. Cheng, Y. Ding, Q. Lv, and Y. Sun, "Support vector machine classification and recognition of power quality disturbances based on S-transform," *Proc. CSEE*, vol. 25, no. 4, pp. 51–56, Feb. 2005, doi: CNKI:SUN:ZGDC.0.2005-04-009.

[2] R. C. Dougan, M. F. McGranagham, S. Santoso, and H. W. Beaty, *Electrical Power Systems Quality*, 2nd ed. New York, NY, USA: McGraw-Hill, 2004.

[3] M. Valtierra-Rodriguez, R. de Jesus Romero-Troncoso, R. A. Osornio-Rios, and A. Garcia-Perez, "Detection and classification of single and combined power quality disturbances using neural networks," *IEEE Trans. Ind. Electron.*, vol. 61, no. 5, pp. 2473–2482, May 2014, doi: 10.1109/TIE.2013.2272276.

[4] D. Granados-Lieberman, R. J. Romero-Troncoso, R. A. Osornio-Rios, A. Garcia-Perez, and E. Cabal-Yepez, "Techniques and methodologies for power quality analysis and disturbances classification in power systems: A review," *IET Gener., Transmiss. Distrib.*, vol. 5, no. 4, pp. 519–529, Apr. 2011, doi: 10.1049/iet-gtd.2010.0466.

[5] L. B. Sun, Z. S. Wu, and K. K. Yang, "Application of islanding detection and classification of power quality disturbance in hybrid energy system," in *Proc. IOP Conf., Earth Environ. Sci.*, Shanghai, China, Jan. 2018, Art. no. 012020.

[6] O. P. Mahela, A. G. Shaik, B. Khan, R. Mahla, and H. H. Alhelou, "Recognition of complex power quality disturbances using S-transform based ruled decision tree," *IEEE Access*, vol. 8, pp. 173530–173547, 2020, doi: 10.1109/ACCESS.2020.3025190.

[7] M. Sahani and P. K. Dash, "FPGA-based online power quality disturbances monitoring using reduced-sample HHT and class-specific weighted RVFLN," *IEEE Trans. Ind. Informat.*, vol. 15, no. 8, pp. 4614–4623, Aug. 2019, doi: 10.1109/TII.2019.2892873.

[8] J. Huang, H. Qu, and X. Li, "Classification for hybrid power quality disturbance based on STFT and its spectral kurtosis," *Power System Technol.*, vol. 40, no. 10, pp. 1391–3184, 2016, doi: 10.13335/j.1000-3673.pst.2016.10.036.

[9] K. Cai, B. Alalibo, W. Cao, Z. Liu, Z. Wang, and G. Li, "Hybrid approach for detecting and classifying power quality disturbances based on the variational mode decomposition and deep stochastic configuration network," *Energies*, vol. 11, no. 11, p. 3040, Nov. 2018, doi: 10.3390/en11113040.

[10] I. S. Samanta, P. K. Rout, K. Swain, M. Cherukuri, and S. Mishra, "Power quality events recognition using enhanced empirical mode decomposition and optimized extreme learning machine," *Comput. Electr. Eng.*, vol. 100, May 2022, Art. no. 107926, doi: 10.1016/j.compeleceng.2022.107926.

[11] A. K. K. Giri, S. R. Arya, R. Maurya, and B. C. Babu, "Power quality improvement in stand-alone SEIG-based distributed generation system using Lorentzian norm adaptive filter," *IEEE Trans. Ind. Appl.*, vol. 54, no. 5, pp. 5256–5266, Sep. 2018, doi: 10.1109/TIA.2018.2812867.

[12] E. Balouji and O. Salor, "Classification of power quality events using deep learning on event images," in *Proc. 3rd Int. Conf. Pattern Recognit. Image Anal. (IPRIA)*, Shahrekord, Iran, Apr. 2017, pp. 216–221.

[13] J. Shukla, B. K. Panigrahi, and P. K. Ray, "Power quality disturbances classification based on Gramian angular summation field method and convolutional neural networks," *Int. Trans. Elect. Energy Syst.*, vol. 31, no. 12, Dec. 2021, Art. no. e13222, doi: 10.1002/2050-7038.13222.

[14] S. Wang and H. Chen, "A novel deep learning method for the classification of power quality disturbances using deep convolutional neural network," *Appl. Energy*, vol. 235, pp. 1126–1140, Feb. 2019, doi: 10.1016/j.apenergy.2018.09.160.

[15] Q. Wang, X. Liang, and S. Qin, "Research on power quality disturbance analysis and identification based on LSTM," *Energy Rep.*, vol. 8, no. 8, pp. 709–718, Feb. 2022.

[16] Y. Deng, L. Wang, H. Jia, X. Tong, and F. Li, "A sequence-to-sequence deep learning architecture based on bidirectional GRU for type recognition and time location of combined power quality disturbance," *IEEE Trans. Ind. Informat.*, vol. 15, no. 8, pp. 4481–4493, Aug. 2019, doi: 10.1109/TII.2019.2895054.

[17] N. Wang, M. Sun, and X. Xi, "Identification of power quality disturbance characteristic based on deep learning," *Electr. Power Syst. Res.*, vol. 226, Jan. 2024, Art. no. 109897, doi: 10.1016/j.epsr.2023.109897.

[18] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate LSTM-FCNs for time series classification," *Neural Netw.*, vol. 116, pp. 237–245, Aug. 2019, doi: 10.1016/j.neunet.2019.04.014.

[19] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, and J. Nuno, "From time series to complex networks: The visibility graph," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 13, pp. 4972–4975, 2008, doi: 10.1073/pnas.070924710.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[21] H. Zheng, X. Li, Y. Li, Z. Yan, and T. Li, "GCN-GAN: Integrating graph convolutional network and generative adversarial network for traffic flow prediction," *IEEE Access*, vol. 10, pp. 94051–94062, 2022, doi: 10.1109/ACCESS.2022.3204036.

[22] J. Chen, Y. Si, C. Un, and S. Siu, "Chemical toxicity prediction based on semi-supervised learning and graph convolutional neural network," *J. Cheminform.*, vol. 13, no. 1, p. Journal of cheminformatics, Nov. 2021, doi: 10.1186/s13321-021-00570-8.

[23] W. Zheng, R. Lin, J. Wang, and Z. Li, "Classification of power quality disturbances based on GAF and convolutional neural networks," *Power Syst. Protection Control*, vol. 49, no. 11, pp. 97–104, 2021, doi: 10.19783/j.cnki.pspc.200997.

[24] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," 2016, *arXiv:1610.02242*.

[25] P. Meel and D. K. Vishwakarma, "A temporal ensembling based semi-supervised ConvNet for the detection of fake news articles," *Expert Syst. Appl.*, vol. 177, Sep. 2021, Art. no. 115002, doi: 10.1016/j.eswa.2021.115002.

[26] W. Ding, M. Abdel-Basset, and H. Hawash, "RCTE: A reliable and consistent temporal-ensembling framework for semi-supervised segmentation of COVID-19 lesions," *Inf. Sci.*, vol. 578, pp. 559–573, Nov. 2021, doi: 10.1016/j.ins.2021.07.059.

[27] X. Shi, H. Su, F. Xing, Y. Liang, G. Qu, and L. Yang, "Graph temporal ensembling based semi-supervised convolutional neural network with noisy labels for histopathology image analysis," *Med. Image Anal.*, vol. 60, Feb. 2020, Art. no. 101624, doi: 10.1016/j.media.2019.101624.

[28] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948, doi: 10.1002/j.1538-7305.1948.tb01338.x.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[30] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," *J. Mach. Learn. Res.*, vol. 15, pp. 315–323, Jan. 2011.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

JIAJUN CAI was born in Guangdong, China, in 1999. He received the B.Sc. degree in measurement and control technology and instrument from Shenzhen University, Shenzhen, China, in 2022, where he is currently pursuing the M.Sc. degree. His research interests include power quality disturbances and signal processing.

HUAIZHI WANG (Member, IEEE) received the B.Eng. and M.Eng. degrees in control science and engineering from Shenzhen University, Shenzhen, China, in 2009 and 2012, respectively, and the Ph.D. degree in electrical engineering from the South China University of Technology, Guangzhou, China, in 2015. From 2014 to 2015, he was a Research Assistant with the Department of Electrical Engineering, The Hong Kong Polytechnic University, Hong Kong. He is currently an Associate Professor with Shenzhen University. His research interests include cyber-physical power systems, multi-terminal HVDC, and power system transient stability.

HUI JIANG (Member, IEEE) received the B.S. degree from Chongqing University, Chongqing, China, in 1990, and the M.S. and Ph.D. degrees in electrical engineering from Hunan University, Hunan, China, in 1999 and 2005, respectively. From 1990 to 2009, she was a Teacher with the College of Electrical and Information Engineering, Hunan University. Since 2003, she has been an Associate Professor. From 2005 to 2006, she was a Visiting Researcher with Brunel University, London, U.K. Since 2008, she has been a Professor of electrical engineering. Since June 2009, she has been a Professor with the College of Physical and Optoelectronic Engineering, Shenzhen University, Guangdong, China. Her research interests include power system economics, complex system optimization, and power quality analysis and control.