A Faults Recognition Method of Transmission Line Insulators Based on Generalized Neural Networks

Chuanmin Ge *

Department of Mechanical Engineering, North China Electric Power University, Baoding, China, 071003

* Corresponding Author Email: ncepu_ge2025@163.com

Abstract. Regular recognition of transmission line insulators (TLI) and rapid diagnosis of their condition are crucial for the stable operation of power systems. To enhance the speed and accuracy of fault diagnosis for TLI, deep learning algorithms are applied to the recognition of TLI faults. However, traditional neural networks suffer from slow convergence speed and low accuracy in fault diagnosis. This paper proposes a recognition method for TLI based on generalized neural networks (GNN). A database for TLI is established by using aerial images, with image coordinates extracted as the sample input set. Then, by studying the characteristics and applications of GNN, a suitable network of GNN is developed to complete the selection and design of the recognition method for TLI. The simulation experiments show that GNN is able to accurately and quickly identify faulty TLI. Compared with the traditional method, the GNN converges faster, and the loss rate decreases by 27%.

Keywords: Generalized Neural Networks, Transmission Line Insulators, Faults Recognition.

1. Introduction

TLIs are indispensable components of power systems for supporting power lines and providing electrical insulation. Therefore, regular recognition of TLI and rapid diagnosis of their condition are critical to the stable operation of the power system [1]. The key to recognizing faults of TLI lies in the rapid recognition of the condition of TLI strings in aerial images. With the continuous breakthroughs in deep learning in the field of computer vision, many researchers have applied deep learning algorithms to the recognition of TLI faults in transmission lines [2]. The use of deep learning algorithms to recognize the location of TLI has become a hot topic of current research [3].

Regarding the problem of locating and recognizing the status of TLI, the current main research approaches are generally divided into the following two steps: 1) locating insulator strings in aerial images, 2) recognizing the status of TLI. Reference [4] proposes a defect detection method for catenary insulators based on contour features and grey similarity matching, but this algorithm suffers from the problem of low detection accuracy. Reference [5] presents a high-precision detection algorithm for transmission line defects based on improved RetinaNet. Reference [6] proposes a machine learning model combining deep and shallow layers, where a deep convolutional neural network and a shallow least squares support vector machine (LSSVM) model are used to extract image features and classify images respectively. Reference [7] utilizes Faster R-CNN for identifying insulator structures and faults, and employs a deep convolutional neural network to automatically extract key features, which can effectively address the issue of missing key features implicit in original images. Reference [8], based on YOLOv5 and YOLOv7 algorithms, uses images collected by UAVs as training materials to successfully detect foreign object faults in insulators. Reference [9] employs a convolutional neural network model to extract and localize features of insulator strings, and a cascade self-organizing feature map network to complete defect recognition of insulator strings. Reference [10] proposes to calculate the aspect ratio of insulators through clustering methods to locate insulators in images, and optimize the clustering algorithm to achieve precise recognition of targets. The method of first identification and then judgment is relatively complex in process; the structure of traditional neural networks is relatively fixed, and training neural networks is not only time-consuming but also leads to excessive loss of feature information.

In this paper, a recognition method of TLI is proposed based on GNN, which improves training efficiency and accuracy while reducing excessive loss of feature information. Firstly, the aerial images are used to establish a database for TLI location and status recognition, and the image coordinates are extracted as the sample input. The structure of GNN is established to train and obtain the optimal weights. Finally, compared with the traditional method, examples are trained and tested in MATLAB (2022b) environment to verify the accuracy and real-time performance of the proposed method.

2. The analysis of TLI faults

2.1. The introduction of TLI faults

TLIs are an important component of power transmission and distribution equipment in power systems. Due to their prolonged exposure to harsh natural environments, TLI inevitably suffer from erosion caused by temperature and humidity fluctuations, rain, ice damage, lightning strikes, and other factors. Additionally, defects in the equipment itself can lead to issues such as cracking, string detachment, and flashover. These problems significantly impair the operational performance of TLI and adversely affect the operational reliability and service life of the entire transmission line.

Using drones to collect samples of suspension TLI on power transmission lines, the aerial photography image set of TLI is used as a dataset to recognize their features at different locations, with bounding box annotations as the output. The numerical input for the bounding box window can be represented by a set of four-dimensional vectors, denoted as $G = (G_x, G_y, G_w, G_h)$, where (G_x, G_y) represents the position of the upper-left corner, and(w, h) represents its width and height.

The sample annotation results are shown in Table.1. The first column is the sample name, the second column is the number of targets, the third and fourth columns are the x and y coordinates of the upper-left corner of the annotated rectangle, and the fifth and sixth columns are the x and y coordinates of the lower-right corner. The data comes from the article. [11].

Table.1 The partly labelled results

Image	Coordinate 1	Coordinate 2	Coordinate 3	Coordinate 4	Amount
001.jpg	173	46	232	344	34.99
002.jpg	183	47	232	353	41.14
003.jpg	266	46	301	328	41.81
004.jpg	269	49	292	297	36.08
005.jpg	268	87	315	324	41.82

2.2. Pre-processing

To reduce data dimensions, highlight effective information, and enable the model to focus more on core data features, thereby improving operational efficiency and prediction accuracy, normalization operations are required after feature extraction. This paper uses the maximum-minimum normalization principle to pre-process the data, with the formula in (1).

$$x' = \frac{x - min\ (x)}{max(x) - min\ (x)} \tag{1}$$

where x represents the original data, and x' represents the normalised data. After processing with this formula, the data will be mapped to a range between 0 and 1. This normalization method makes data easier to process during model training.

3. Comparison of GNNs with traditional neural networks

3.1. Convolutional neural network（CNN）

3.1.1 The structure of CNN

It is primarily composed of the following layers:

(1) Data Input Layer

The data input layer of a CNN is primarily responsible for data preprocessing, adjusting raw data to make it easier for the convolutional computation layer functions to recognize.

(2) Convolutional Computation Layer

The convolutional computation layer is the most critical component of neural network computation.

The following is the detailed formula for convolutional computation:

a. Input matrix: Assume that the size of the input matrix is $W_{in} \times H_{in} \times D_{in}$, where W_{in} is the width, H_{in} is the height, and D_{in} is the depth (number of channels).

b. Convolution kernel: The size of the convolution kernel is $K \times K \times D_{in}$, where K is the size of the convolution kernel and D_{in} is the depth of the input matrix.

c. Step size: If the step size is S, then the convolution kernel slides S pixels horizontally and vertically each time.

d. Zero padding: Zero padding is the operation of adding zero-value pixels to the boundaries of the input matrix. If the zero padding is P, then P pixels are added to each side of the input.

e. Output matrix: The size of the output matrix is $W_{out} \times H_{out} \times D_{out}$, where W_{out} is the output width, H_{out} is the output height, and D_{out} is the output depth.

f. The output width and height are calculated by (2)-(3).

$$W_{out} = \left[\frac{W_{in}-K+2P}{S}\right] + 1 \tag{2}$$

$$H_{out} = \left[\frac{H_{in}-K+2P}{S}\right] + 1 \tag{3}$$

g. The formula for calculating output depth is shown by (4).

$$D_{out} = \text{number of convolution kernels} \tag{4}$$

The convolution process is shown in Figure.1.

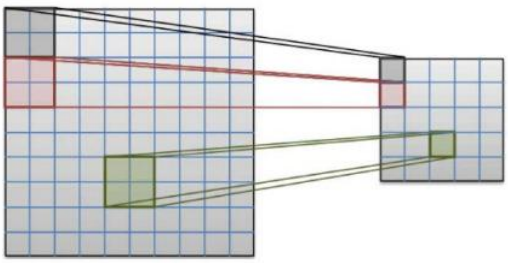

Figure. 1 The image of the convolution process.

(3) Activation Layer

After completing the convolution calculation, the entire network needs to undergo a non-linear transformation through the activation function, which converts the previous input into a new non-linear feature. Different activation functions can be used for different layers within networks. The following are some commonly used activation functions. Figure. 2 shows the image ReLU function.

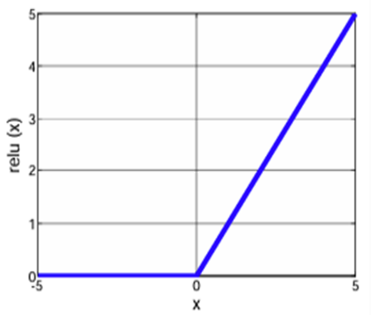

Figure. 2 The image of ReLU function.

The formula is expressed by (6).

$$R(x) = max(0, x) \tag{5}$$

The ReLU function has a slope of 1 when $x > 0$ and a slope of 0 when $x < 0$. The ReLU function introduces non-linearity, enabling neural networks to learn more complex non-linear mappings, thereby enhancing the model's expressive power. Additionally, the ReLU function outputs 0 for positive values, resulting in faster computation speeds. Compared to the Sigmoid function, it is less prone to the vanishing gradient problem.

(4) Pooling Layer

The pooling layer follows immediately after the convolutional layer. Its primary function is to prevent overfitting caused by an excessive number of parameters. Convolution and pooling operations work in tandem: the next convolutional layer and pooling layer perform convolution and pooling operations on the results from the previous layer, repeating this process iteratively until the final feature map is generated. The formula for calculating the feature map output is shown by (7)-(8).

$$W_{out} = \left[\frac{W_{in}-K}{S}\right] + 1 \tag{6}$$

$$H_{out} = \left[\frac{H_{in}-K}{S}\right] + 1 \tag{7}$$

The pooling schematic is shown in Figure. 3.

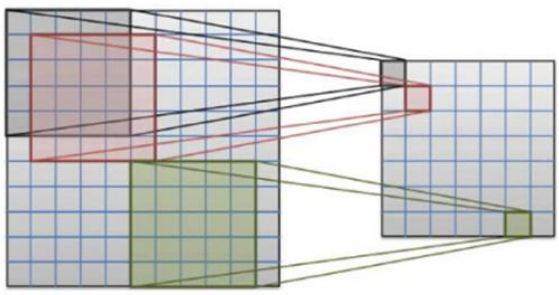

Figure. 3 The image of pooling process.

(5) Fully connected layer and classification network

Fully connected layers and classification networks are located at the end of a CNN. After repeatedly performing convolution and pooling operations, several feature maps are obtained. The fully connected layer is responsible for connecting all extracted features. The output of the fully connected layer is an $n \times 1$ vector. Several fully connected layers perform dimension reduction operations on the vector and correspond to the classifier.

(6) Loss Layer

The loss layer is used to define and calculate the difference or loss between the model output and the true value. During the training process, the ultimate goal of the model is to optimize the loss value, minimizing it to better fit the data. Different loss functions have been selected for different scenarios.

3.1.2 Limitation of CNN

The above is an introduction to CNNs, which have achieved great success in image recognition and classification tasks, but they also have some shortcomings and limitations. CNNs suffer from high computational costs and lengthy training times, requiring substantial computational resources for training and fitting, which can be time-consuming. Additionally, they face issues such as poor interpretability and data gaps.

3.2. Modeling of GNN

3.2.1 The structure of GNN

GNN consists of four layers: the input layer, the pattern layer, the summation layer, and the output layer. Its structural diagram is shown in Figure. 4.

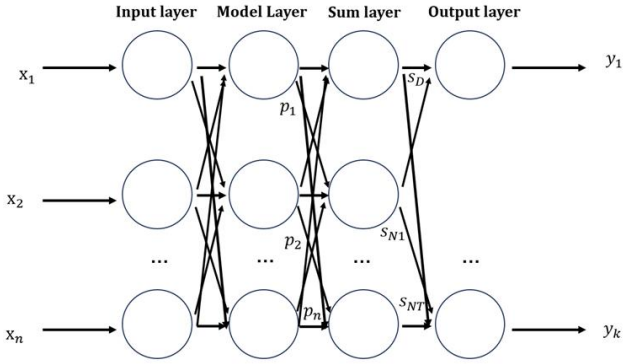

Figure. 4 The structure of GNN.

(1) Input layer

The number of neurons is equal to the dimension of the input vector in the training samples. Each neuron is a simple distribution unit that directly passes input variables to the pattern layer.

(2) Pattern layer

The number of neurons is equal to the number of training samples n. Each neuron corresponds to a different sample. The transfer function of the pattern layer neurons is shown by (9).

$$p_i = exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right] \quad i = 1,2,3\ldots,n \tag{8}$$

The output of neuron i is the exponential form of the square of the Euclidean distance between the input variable and its corresponding sample X: ${D_I}^2 = (X - X_i)^T(X - X_i)$. In this equation, X is the network input variable; X_i is the learning sample corresponding to the i-th neuron.

(3) Summation layer

Two types of neurons are used for summation in the summation layer.

The first type of formula is $\sum_{i=1}^{n} exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$, which calculates the arithmetic sum of the outputs of all pattern layer neurons. The connection weights between the pattern layer and each neuron are 1, and the transfer function is shown by (10).

$$S_D = \sum_{i=1}^{n} g_i \tag{9}$$

Another type of formula is $\sum_{i=1}^{n} Y_i exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right]$, which performs a weighted sum of all neurons in the pattern layer. The connection weight between the i-th neuron in the pattern layer and the j-th summing neuron in the summing layer is the j-th element in the i-th output sample Y_i, and the transfer function is expressed by (11).

$$S_{Ni} = \sum_{i=1}^{n} w_{ij} g_i\,, \quad j = 1,2,\ldots,k \tag{10}$$

(4) Output layer

The number of neurons in the output layer is equal to the dimension k of the output vector in the learning sample. Each neuron divides the output of the summation layer, and the output of neuron *j* corresponds to the j-th element of the estimated result $\hat{Y}$（X）,the formula is expressed by (12).

$$y_j = \frac{S_{Nj}}{S_D} \quad j = 1,2,3 \ldots, k \tag{11}$$

3.2.2 Solution Process

Step 1: Data preparation and preprocessing

Select the dataset of historical records and real-time collection in 1.1, the input part is usually the feature variables, while the output part is the target variables. The numerical output of the border window can be represented by a set of four-dimensional vectors, denoted as $P = (P_x, P_y, P_w, P_h)$, where（P_x, P_y）represents the position of the upper left corner and （w, h） are its width and height. Its corresponding truth value is denoted by $G = (G_x, G_y, G_w, G_h)$ and the meaning of each parameter is similar to that of P . In order to make a better match between the border and the actual target position, it involves translation and scale scaling of the border, i.e., given (P_x, P_y, P_w, P_h), a mapping fischosen such that $f(P_x, P_y, P_w, P_h) \approx (G_x, G_y, G_w, G_h)$. The formulas for the displacement and scaling factors are expressed by (13)-(14).

$$(\Delta x, \Delta y),\ \Delta x = P_w d_x(P),\ \Delta y = P_h d_y(P) \tag{12}$$

$$(S_w, S_h),\ S_w = P_w d_x(P),\ S_h = P_h d_h(P) \tag{13}$$

The target estimate is as shown in the formula (15).

$$\widetilde{G_x} = P_w d_x(P) + P_x,\ \widetilde{G_y} = P_h d_y(P) + P_y,\ \widetilde{G_w} = P_w e^{d_w(P)},\ \widetilde{G_h} = P_h e^{d_h(P)} \tag{14}$$

It can be seen that it is necessary to learn the four transformations $d_x(P)$， $d_y(P), d_w(P), d_h(P)$. For the input image, extract the feature vector $\emptyset_s(P)$, predict the change as$d_*(P) = {w_*}^T \emptyset_s(P)$, where $*$ represents *x*, *y*, *w*, *h*, and w_* is the parameter to be learned through regression. After feature extraction, a normalization operation is performed on the dataset. The principle of maximum-minimum normalization is chosen for this experiment. After processing by this formula, the data will be mapped to the range between 0 and 1.

Step 2: Construction of GNN model and training.

Use the newgrnn function to build a GNN model, and this paper adopts the classical 4-layer feature extraction model. After training on the validation set to get the pre-training model, use the specified dataset to fine-tune the training function and its parameters, so that the output of the network is as close as possible to the target variable. After the training is completed, the trained network is used to predict the training set, and the prediction result Y is obtained.

Step 3: Model validation.

After model training is complete, it is usually necessary to further adjust the model parameters to optimize model performance. For GNN, the smoothing parameter is a key parameter that directly affects the model's complexity and prediction accuracy. The optimal smoothing factor is selected through grid search, and ultimately selects the parameter combination with the best performance.

4. Simulation Analysis

4.1. The setup of the simulation

The hardware environment used for the experiment is Intel(R)Core (TM) i5-12400@3.50Hz processor, and the simulation environment is configured under MATLAB (2022b), using Windows 10 operating system. The parameters shown in Table 2 are to set up the main network.

Table 2 Main parameter settings

Parameter	value
Number of CPUs	1
Iteration count	80
learning rate	0.001
Weight decay	0.006

The original dataset consists of 100 aerial images of TLI captured via UAVs, and each aerial image with an original size of 6000 × 4000 is intercepted as a part local 512 × 512 sized sample, and then the samples are disrupted and normalized to make them randomly distributed. The original image coordinates of TLI are extracted as the dataset, 20% of the data (20 groups) are selected as the validation set, and the remaining 80% of the data (80 sheets) are used as the training setting.

The partly labelled results are shown in the Table 3 below, where the first column is the sample name, the second column is the number of targets, the third and fourth columns are the x and y coordinates of the points in the upper left corner of the labelled rectangular box, and the fifth and sixth columns are the x and y coordinates of the points in the lower right corner, respectively.

Table 3 Sample annotation results

Image	Coordinate1	Coordinate2	Coordinate3	Coordinate4	Amount
001.jpg	173	46	232	344	34.99
002.jpg	183	47	232	353	41.14
003.jpg	266	46	301	328	41.81
004.jpg	269	49	292	297	36.08
005.jpg	268	87	315	324	41.82
006.jpg	204	52	248	307	35.21
007.jpg	270	21	300	372	38.86
008.jpg	207	20	240	343	36.59
009.jpg	123	11	173	351	32.71
010.jpg	107	13	153	329	38.46
011.jpg	138	45	178	344	26.02
012.jpg	196	38	250	347	28.03
013.jpg	382	91	420	353	46.37
014.jpg	259	52	302	367	33.91
015.jpg	254	50	300	364	32.44
016.jpg	257	49	301	363	34.05
017.jpg	257	46	300	343	31.29
018.jpg	217	39	259	352	38.01
019.jpg	294	19	322	341	49.31
020.jpg	269	49	292	297	29.23

To verify the effectiveness of the proposed method, the saliency recognition model of TLI based on CNN in Case 1, the recognition model of TLI based on GNN in Case 2.

4.2. Training results

(1) Operation results of Case 1

The accuracy of the recognition method using CNN is about 80%, the average recognition time is 600ms/frame, the loss rate is 10%, and there is obvious gradient instability when deepening the number of network layers to improve the performance.

(2) Operation results of Case 2

The image of results is shown in Figure.5. After training using the GNN training model, the root mean square error (RMSE) of the true value and the predicted value of the training setting is 2.6, and the average recognition time is 300s/frame, which is a good result when the model is applied to the recognition of TLI.

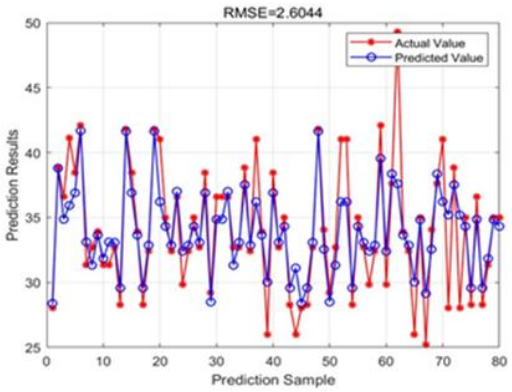

Figure. 5 GNN calculation results image

(3) Loss amounts

When using the model in the paper for training, the change of the loss value of each training process is obtained to compare with the traditional model, the loss function value results of GNN are shown in the Figure. 6.

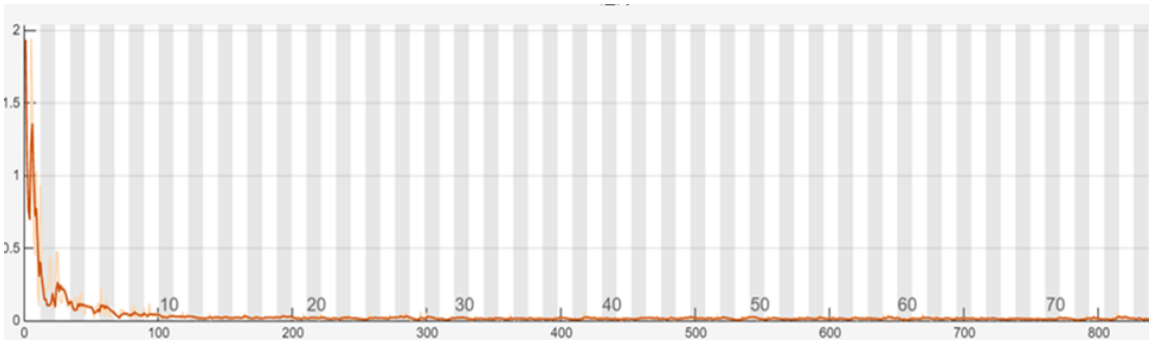

Figure. 6 Loss function image

The loss function value is inversely proportional to the performance of the model, after the same number of trainings, the loss value of the GNN model is smaller than the CNN.

From the above results, it can be concluded that the loss function curve derived using the GNN model is smoother and the loss rate is reduced by 27% compared to the CNN. The use of GNN for TLI image feature extraction can reduce the loss of feature information and improve the recognition accuracy. Meanwhile, the GNN can quickly balance the parameter updates of each layer during the training process, effectively avoiding the problems of gradient dispersion and overfitting, and the training convergence speed is faster than that of CNN.

5. Summary

To solve the problems of slow convergence speed and excessive loss of feature information using CNNs, this paper proposes a recognition method based on GNNs, which provides a decision-making tool for traditional inspection methods. Compared with the traditional use of CNNs for insulator fault diagnosis, the model proposed in this paper has a faster convergence speed, less loss of feature information, and good practical results. The loss rate of the model proposed in this paper is reduced by 27%. This model has not been tested for TLI faults in complex backgrounds such as snow and houses, and can be tested and analyzed in the future without using backgrounds, while gradually expanding to other types of faults in power inspection, such as strand breakage and tower corrosion.

References

[1] Zhang X，Zhang Y，Liu J,et al.InsuDet: A Fault Detection Method for Insulators of Overhead Tr ansmission Lines Using Convolutional Neural Networks[J] EEE Transactions on Instrumentation a nd Measurement，2021,70:1-12.

[2] Jafari M. and Sarkar P.Buffeting and self-excited load measurements to evaluate ice and dry galloping of yawed power transmission lines. journal of structural engineering[J], 2021, 147(11): 1-14.

[3] Tao X，Zhang D，Wang Z, et al. Detection of Power Line Insulator Defects Using Aerial Images Analyzed with Convolutional Neural Networks[J]. IEEE Transactions on Systems, Man, and Cybe rnetics: Systems，2020,50(4):1486-1498.

[4] Tan P, Li X, Xu J, et al. Catenary insulator defect detection based on contour features and gray similarity matching[J]. Journal of Zhejiang University-SCIENCE A, 2020, 21(1): 64-73.

[5] Liu J, Jia R, Li W, et al. High precision detection algorithm based on improved RetinaNet for defect recognition of transmission lines[J]. Energy Reports, 2020, 6: 2430-2440.

[6] Lu J., Ye Y., Xu X., et al. Application research of convolution neural network in image classification of icing monitoring in power grid[J]. EURASIP Journal on Image and Video Processing, 2019, 2019(49): 1-11.

[7] Liu X, Jiang H, Chen J, et al. Insulator detection in aerial images based on faster regions with convolutional neural network[C]//2018 IEEE 14th international conference on control and automation (ICCA). IEEE, 2018: 1082-1086.

[8] Wang Q, Liao Z, Xu M. Wire Insulator Fault and Foreign Body Detection Algorithm Based on YOLO v5 and YOLO v7[C]//2023 IEEE International Conference on Electrical, Automation and Computer Engineering (ICEACE). Changchun, China, 2023: 1412-1417.

[9] Yan B, Chen Q, Ye R, et al. Insulator detection and recognition of explosion based on convolutional neural networks[J]. International Journal of Wavelets Multiresolution and Information Processing, 2019, 17(2): 81-103.

[10] ZHENG H B, SUN Y H, LIU X H, et al. Infrared Image Detection of Substation Insulators Using an Improved Fusion Single Shot Multibox Detector[J]. IEEE Transactions on Power Delivery, 2021, 36(6): 3351-3359.

[11] GUOYING LU, BENHAO LI, YINING CHEN,Wu Y , Liu J , et al. Precision in Aerial Surveillance: Integrating YOLOv8 With PConv and CoT for Accurate Insulator Defect Detection [J]. IEEE Access, 2025:49062-49075