Pattern Analysis and Applications (2024) 27:94
https://doi.org/10.1007/s10044-024-01315-7

INDUSTRIAL AND COMMERCIAL APPLICATION

A lightweight weld defect recognition algorithm based on convolutional neural networks

Wenjie Zhao[1] · **Dan Li**[1] · **Feihu Xu**[1]

Received: 24 August 2023 / Accepted: 22 July 2024 / Published online: 4 August 2024

Abstract
This paper proposes a lightweight weld defect-recognition algorithm based on a convolutional neural network that is appropriate for weld defect recognition in industrial welding. Specifically, the developed scheme relies on the original SqueezeNet model. However, we improve the fire module to reduce the model's parameter cardinality, introduce the ECA module to strengthen the learning of feature channels and improve the feature extraction ability of the overall model. The experimental results highlight that our algorithm's average recognition rate on the overall defects of welding depressions, welding holes, and welding burrs reaches 97.50%. Note that although our model requires substantially fewer parameters, its recognition effect is significantly improved. Our algorithm's feasibility is verified on the test data and challenged against current weld defect identification algorithms, demonstrating its enhanced identification role and application prospect.

Keywords Weld defect recognition · Convolutional neural networks · SqueezeNet model · ECA module

1 Introduction

Welding is often used in industrial processes and is widely used in railways, bridges, chemical equipment manufacturing [1], and other fields. With the continuous improvement of industrial production levels, the requirements for welding quality are becoming more and more stringent. However, due to the limitation of the working environment, various welding defects such as cracks, pores, and cracks are inevitably produced [2]. Welding defects can lead to significant safety risks, so identifying them is important.

Currently, the mainstream defect identification methods include traditional and deep learning methods [3]. The traditional method classifies defects based on statistical information and image features [4], involving defect segmentation, feature extraction, feature selection, and defect identification [5]. Weld defect segmentation [6] mainly extracts the defect areas, such as the improved OTSU method [7]. Feature extraction obtains feature sets that describe defects, such as edge features, area-based features, and texture features [8], and feature selection mainly removes redundant features and noise while retaining useful features. Defect identification involves identifying the type and nature of defects and is the core stage of the entire defect identification system.

Defect identification mostly adopts multi-source information fusion decision-making [9], Bayesian [10], support vector machines, and fuzzy logic. The traditional defect recognition method aims to determine the defect's target point within the image by extracting the feature points and geometric relationships while the external environment is relatively fixed. For instance, Zhou et al. [11] used the Hough transform to identify welds, and Xu [12] developed a SIFT feature point extraction method to identify the weld. However, traditional defect identification methods suffer from poor generalization ability. Thus, it is necessary to increase the cost of the welding process to accurately cooperate with the visual sensor and enhance the welding process accuracy. However, this cooperation has extremely high requirements for the working environment, and once its changes slightly, it must be remodeled, adjusted, and calibrated.

In recent years, deep learning methods based on convolutional neural networks (CNNs) have become a research hotspot in image processing and pattern recognition [13–15]. Unlike the traditional recognition method, a CNN can extract highdimensional nonlinear graph features with good stability and is less susceptible to interference from external local conditions. As a feedforward neural network,

✉ Dan Li
lidan@ahut.edu.cn

[1] Ma'anshan, China

a CNN avoids complex pre-processing, which is evident in traditional recognition algorithms, by employing the original image as the model's input and extracting its features. Due to their high recognition efficiency and strong data representation ability, CNNs have been widely used in various fields [16–18]. For example, Khumaidi et al. [19] utilized a CNN to classify weld defects in images obtained by a webcam. Although the final accuracy reached 95.8%, this method required large sample size to achieve high precision. Jiao et al. [20] extracted 15 characteristic parameters that can be used to characterize the weld surface defect status from the weld image and combined them with a BP neural network to identify the internal weld surface defect. However, the overall recognition accuracy (91%) was inadequate for actual industrial production. Zhou et al. [21] proposed an improved U-Net robust weld recognition algorithm based on fusion attention mechanism. This method improved the feature fusion and loss function, and added a feature classification structure to output the corresponding weld type name. Experimental results show that this method achieved a weld recognition accuracy of 95.6%. However, this method is mainly used for weld detection, and its effectiveness in welding defect recognition remains to be explored. Zhang [22] proposed a welding defect identification algorithm based on principal component analysis to extract crack defects that may occur in the welding area of long-distance pipelines. To enhance recognition accuracy, CNN has become deeper, and some key technologies have been proposed, such as skip connection (SC) [23] and batch normalization (BN) [24], to ease training such deep neural networks.

In classification networks, common deep learning networks are VGG-16 [25], PReLU-nets [26], and ResNet [27]. Although these networks attain a high recognition accuracy, their parameters and calculations are excessive and thus inappropriate for resource-constrained devices. Moreover, in actual welding, the hardware resources used in the weld defect identification network model are limited, and therefore deploying such a network must consider computing power, power, and hardware storage space while ensuring a fairly high recognition accuracy. Therefore, how to compress the deep CNN model's computational complexity and storage space has become a current research hotspot. Indeed, Han et al. [28] used pruning, clustering, and Hoffman coding to compress the mode's storage space. Nevertheless, applying such a network in the mobile terminal of the industrial site requires the model to be lightweight while attaining high recognition accuracy, reducing the calculation burden, and maximizing the computing power requirements.

Spurred by the above analysis, this study improves SqueezeNet to have fewer parameters, higher recognition accuracy and does not require many training samples to identify weld defects. The improved SqueezeNet model performance is analyzed and compared with other existing defect classification algorithms considering precision and accuracy. The results further verify the effectiveness of the proposed model.

2 SqueezeNet network model

The SqueezeNet model is a lightweight network model proposed by Iandola et al. [29] in 2016, based on AlexNet. This paper shrinks the network's convolution kernel and uses the mean pooling layer instead of the fully connected layer [30]. These modifications reduce the parameter cardinality of the original AlexNet to about 1/50 while ensuring the recognition accuracy and the model's memory occupation is only 4.8 MB.

Figure 1 illustrates the SqueezeNet model, which mainly comprises a convolutional layer (Conv), a max pooling layer (Maxpool/2), a fire module (Fire) and a global average pooling (GAP). When the model receives an input image, it initially passes through the first convolutional layer, which utilizes 96 7×7 convolution kernels for preliminary feature extraction. This is immediately followed by a max pooling layer that reduces the spatial dimensions of the feature map. Subsequently, the image progresses through several "Fire" modules in sequence. Figure 2 shows the specific structure of the fire module highlighted in blue in Fig. 1. The Fire module primarily consist of squeeze and expand layers. The squeeze layer employs a 1×1 convolution kernel to decrease the input feature channels, while the expand layer substitutes the 3×3 convolution kernels with 1×1 kernels, effectively reducing the model's parameter count. Additionally, a max pooling layer is inserted after certain Fire modules to further reduce the spatial size. Following the final Fire module, the network includes a large convolutional layer with 1000 output channels, which addresses the image classification task for 1000 categories. At the network's terminus, a global average pooling layer (GAP) highlighted in orange replaces the traditional fully connected layer. This layer calculates the average value of each feature map and produces a fixed-size output, which not only decreases the model's parameter count but also aids in reducing the risk of overfitting. Finally, the softmax layer transforms the output of the global average pooling layer into a probability distribution, predicting the likelihood that the image belongs to each category.

3 Proposed method

This paper improves the classic lightweight network SqueezeNet to build a lightweight CNN model for weld defect recognition. The major improvements are the following. The first is to improve the fire module of the SqueezeNet model and apply depthwise separable convolution to the fire module to reduce the number of

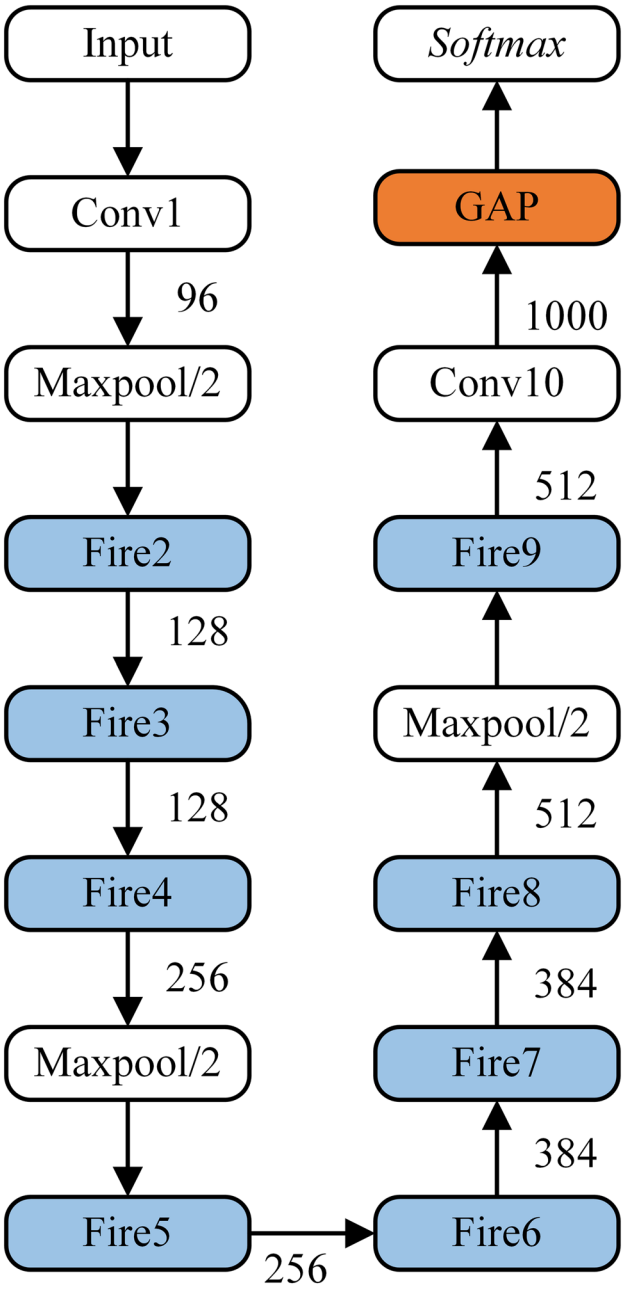

Fig. 1 The network architecture of SqueezeNet

parameters further. Second, the residuals are introduced in the SqueezeNet to solve the deep network degradation problem. Third, the ECA channel attention mechanism module is introduced to extract important features and improve the model's recognition efficiency.

3.1 Improvements to the fire module

In 2017, Google proposed a lightweight network called MobileNet, which has now evolved to v3 [31]. In version v1, the concept of Depthwise Separable Convolution was proposed, which comprises Depthwise Convolution (DW Conv) combined with Pointwise Convolutions (PW Conv). The Fig. 3 depicts the main steps of depthwise separable convolution, including two main stages: depthwise convolution and pointwise convolution. Initially, depthwise convolution operates on each channel of the input feature map separately, using separate convolution kernels, and then the outputs of these convolution kernels are spliced together to form the final output. This approach avoids the need to convolve on all input channels in standard convolution, significantly reducing the number of parameters and computational effort. Following the depthwise convolution, batch normalization is applied to enhance training efficiency and stability. The ReLU activation function is subsequently used to introduce nonlinearity into the network, enabling the model to capture more complex features. Next is pointwise convolution, using a 1×1 convolution kernel to process the output of the depthwise convolution. The primary purpose of this stage is to amalgamate the features from various channels produced during the depthwise convolution. Similar to the earlier stage, each pointwise convolution is followed by batch normalization and ReLU activation to further stabilize network behavior and bolster its nonlinear properties.

The Fire module is the core of SqueezeNet and uses lightweight strategies to reduce the network's parameters. The Fig. 4a below reveals that the traditional Fire module mainly consists of two layers of convolution: a squeeze layer using a 1×1 convolution kernel and an expand layer using a mixture of 1×1 and 3×3 convolution kernels [32]. A ReLU activation function is added after each convolution layer to give the model stronger representation capabilities and introduce nonlinearity.

By replacing the convolution kernel in the Fire module, the model's parameters are reduced to a certain extent. However, as SqueezeNet becomes deeper due to employing many Fire modules, the 3×3 convolution will still produce many parameters, and thus the Fire module must be improved further.

In Fig. 4b, the new Fire module (N-Fire) has two major changes compared to the old. First, the BN (Batch Normalization) layer is added before the original ReLU activation function, which can stabilize the gradient changes of each layer and effectively converge the network during the training process. The second modification adopts the MobileNet concept and replaces the original 3×3 standard convolution of the expanding layer in the Fire module with 3×3 Depthwise Separable Convolution. It can not only ensure the

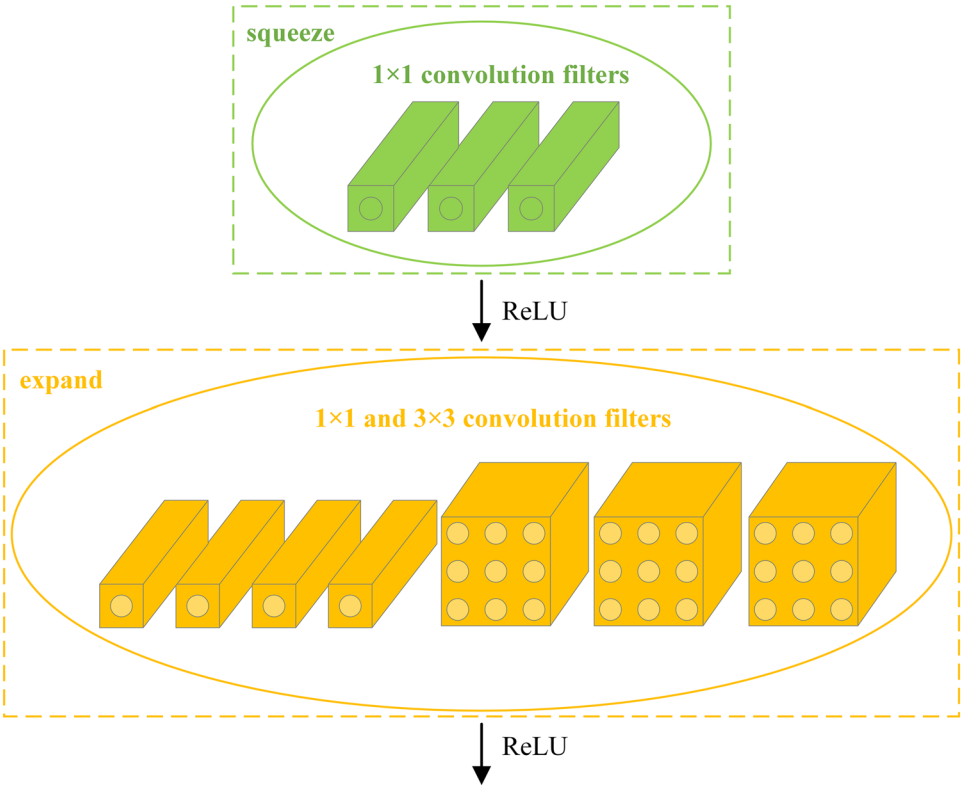

Fig. 2 Structural diagram of fire module in SqueezeNet

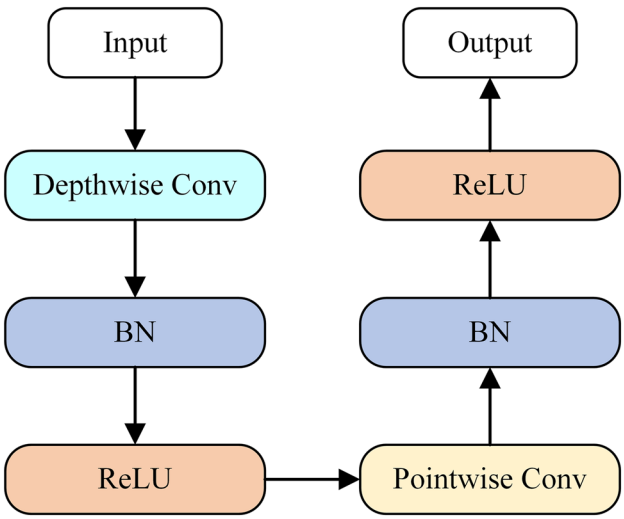

Fig. 3 Flowchart of depthwise separable convolution

performance of the model, but also significantly reduce the number of parameters of the SqueezeNet model.

3.2 Use of residual structures

The original SqueezeNet model is a simple tandem structure similar to the VGG network, aiming to overcome the gradient vanishing and gradient explosion problems caused by deepening the network. In order to assist the network's recognition accuracy from reaching saturation or even suffering from a recognition accuracy decrease, a residual structure is introduced in SqueezeNet.

Figure 5 depicts the architecture of a residual block, commonly used in deep neural networks to address the vanishing gradient problem. The input X first passes through a weight layer followed by ReLU activation, forming the function $F(X)$. It then proceeds to a second weight layer. The output of this sequence is added to the original input X, forming $F(X)+X$, which then passes through another ReLU activation. The addition of X directly to the output of the network layers allows the gradient to flow directly through the network, thereby preserving the gradient magnitude and improving training efficiency.

3.3 Use of ECA modules

The channel attention mechanism plays an important role in improving the performance of deep convolutional neural networks. The ECA module is a channel attention mechanism, which is an ultra-lightweight attention module. By learning feature channels, each channel is divided into different attention values. Then the network can reasonably allocate computing resources and improve the model's accuracy given that number of small parameters increases.

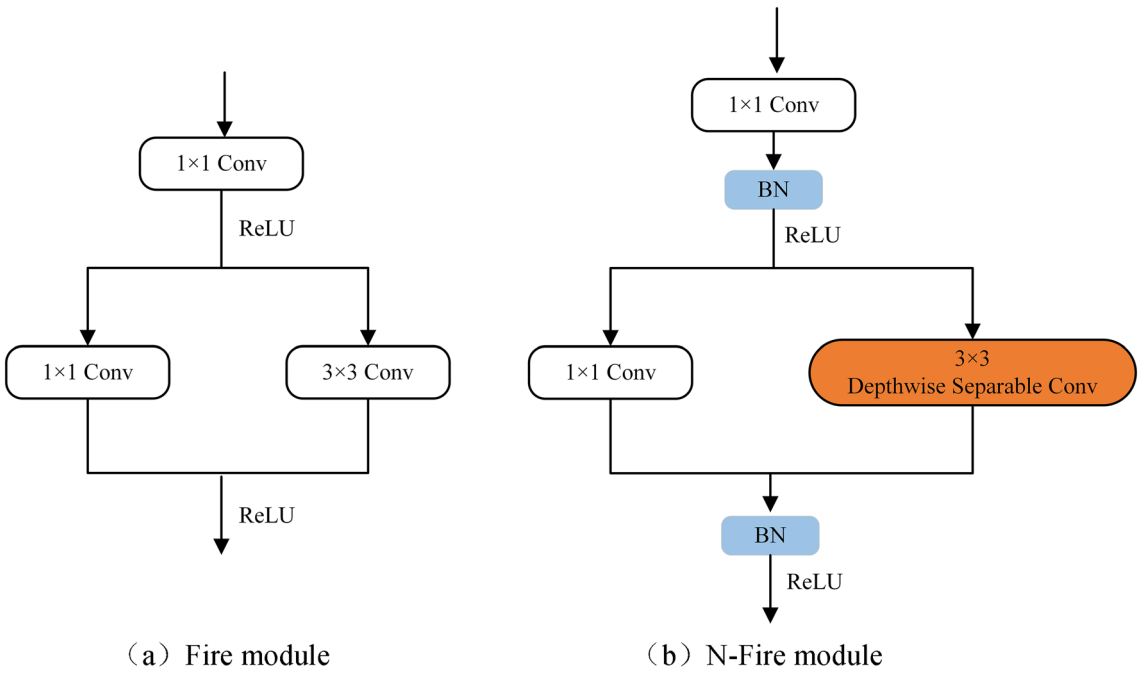

Fig. 4 Structural comparison of fire module and N-Fire module

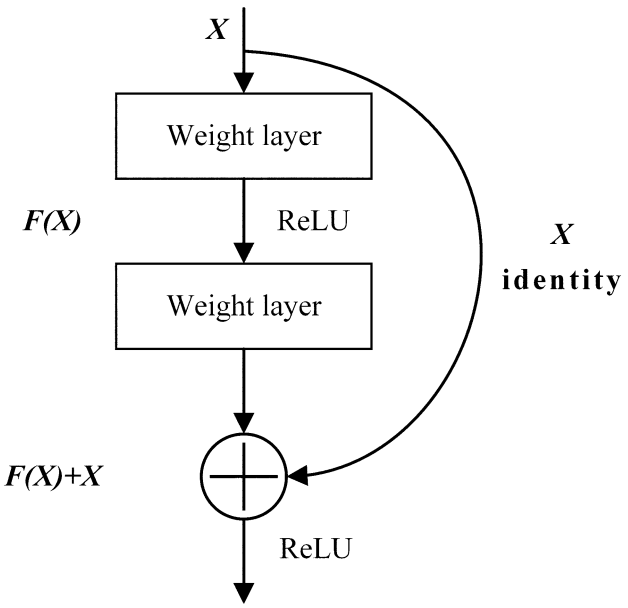

Fig. 5 Schematic diagram of the residual structure

The ECA module structure diagram is shown in Fig. 6, where the ECA module uses a 1×1 convolutional layer directly after the global average pooling layer, and the fully connected layer is removed. The module avoids dimensionality reduction and effectively captures cross-channel interactions. Although the ECA modules involve only a few parameters, they achieve good results. The ECA module realizes cross-channel information interaction through a one-dimensional convolution, and the convolution kernel size adapts to changes through a function. Given channel dimension C, the kernel size is adaptively determined by K:

$$K = \left| \frac{log_2(C)}{\gamma} + \frac{b}{\gamma} \right| \tag{1}$$

where γ and b are hyperparameters.

Figure 6 depicts the implementation process of the ECA module. First, the input feature map passes through the global average pooling layer (GAP), and the feature map changes from the matrix $[H,W,C]$ to the vector $[1,1,C]$. Then, the adaptive one-dimensional convolution kernel size K is calculated according to the number of feature map channels. After obtaining the K value, it is used in the one-dimensional convolution to obtain the weight of each feature map channel. Finally, the normalized weight and the original input feature map are multiplied channel by channel to generate a weighted feature map. In this paper, the γ and b are set to 2 and 1 respectively, so the convolution kernel K is equal to 3.

3.4 Improved SqueezeNet

Figure 7a illustrates the final improved SqueezeNet network model, which contains the N-Fire module, residual structure and ECA module. The input feature map initially

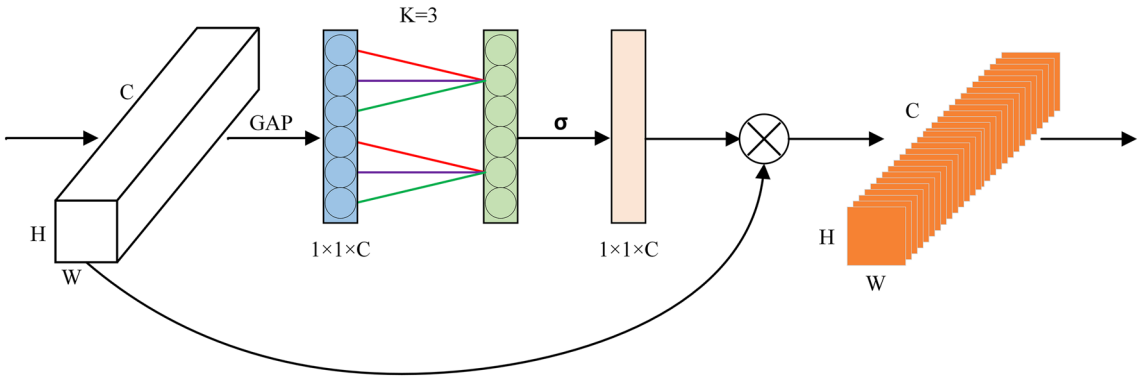

Fig. 6 ECA module structure diagram

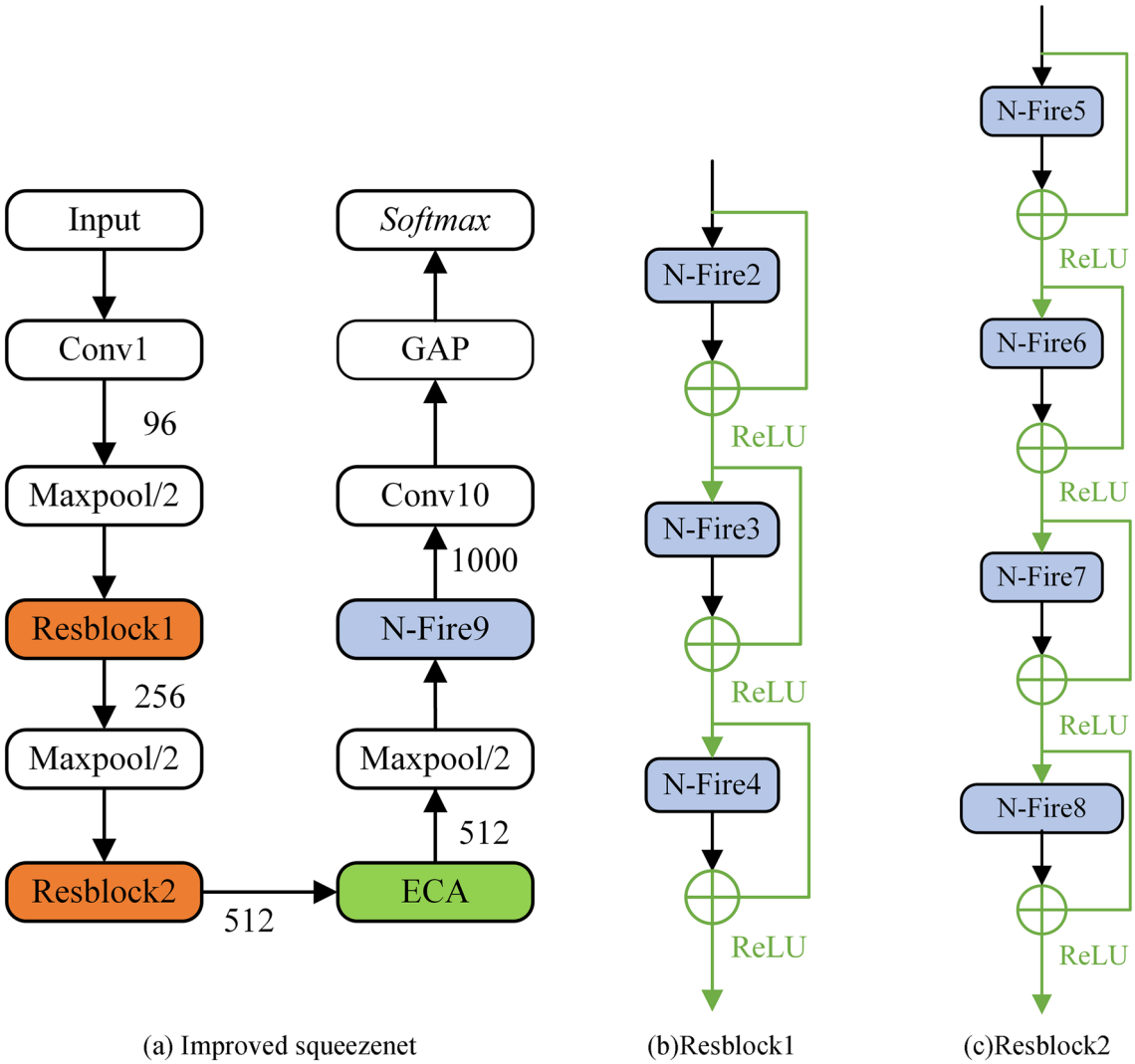

Fig. 7 Structure of the improved SqueezeNet

passes through a convolutional layer (Conv1), followed by a max pooling layer (Maxpool/2). It then passes through two blocks (Resblock1 and Resblock2), each followed by a max pooling layer to reduce the size of the feature map. The Resblock2 is followed by an ECA module to boost the model's ability to recognize important features by adjusting the weights of channel features, thereby enhancing overall performance. The subsequent process includes an N-fire module (N-Fire9), a convolutional layer (Conv10), and a global average pooling (GAP), and finally the classification result is obtained through a *Softmax* layer.

As shown in Fig. 7b, Resblock1 consists of three N-Fire modules (N-Fire2, N-Fire3 and N-Fire4), each of which is connected by a ReLU activation function. The introduction of the N-Fire module significantly reduces the model's parameter count. The green line in the figure represents the residual connection, which directly connects the input and output of the N-Fire module. This design helps alleviate the gradient vanishing problem in deep networks and ensures that the gradient can be effectively propagated in the network.

The structure of Resblock2 is shown in Fig. 7c. It is similar to Resblock1 and contains four modules: N-Fire5, N-Fire6, N-Fire7, and N-Fire8. Each module is followed by a ReLU activation function. The residual connection also directly connects the input of the N-Fire module to its output, improving the ability to learn complex functions while maintaining the depth and efficiency of the network.

4 Datasets and evaluation indicators

4.1 Experimental datasets

The Institute of Computer Application of Liaoning Normal University creates the weld defect data set used in the following experiments. The dataset is a linear structured light weld dataset made by a laser sensor and the triangular ranging principle [33]. The dataset images are divided into four categories: welding depressions, welding holes, welding burrs, and no defects. Figure 8 presents a structured light weld under the four defect types. Among them, welding depressions are caused by improper heat control; welding holes are caused by trapped gas or incomplete material evaporation; welding burrs are sharp, protruding metal pieces or spatters formed due to uneven metal melting and solidification.

Given that the original dataset is small and the number of categories is inconsistent, the balance of categories is low, and therefore the dataset must be enriched. To ensure the effectiveness of subsequent image classification and recognition training, we employ the same data augmentation method as described in reference [33]. For the image with no specific defect types, symmetry and rotation were used to enlarge the image data. For the image with the hole type, the image data were enlarged by rotating the origin of the image 90° to the right. Following augmentation, the dataset comprises a total of 2000 images, evenly distributed with 500 images in each of the four categories.

The image dataset is divided into a training, validation, and test set using an 3:1:1 ratio. The image data of the training set is used to train the model, and the image data of the validation set is used to predict the model. During training, images of the input model are pre-processed using Mosaic data enhancement and adaptive image scaling.

4.2 Evaluation indicators

This article employs a range of evaluation metrics to assess the model's performance, including *Precision*, *Accuracy*, *Recall*, and *F1-score*. Additionally, to evaluate the model's efficiency, *FLOPs* and *Parameters* are also used. The classification of real samples involves two types: positive and negative samples. *TP* (True Positive) indicates that positive samples are correctly identified as positive, while *TN* (True Negative) denotes that negative samples are correctly identified as negative. Conversely, *FP* (False Positive) occurs when negative samples are incorrectly classified as positive, and *FN* (False Negative) refers to instances where positive samples are mistakenly identified as negative.

Precision is the proportion of correctly identified positive samples among all samples that are predicted to be positive. The formula for calculating precision is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Accuracy, which is the percentage of all correctly predicted samples in the total sample, is calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

where $TP + TN$ is the sum of the correctly predicted positive and negative classes. Generally, the higher the accuracy, the better the model's classification effect.

Recall, which is the percentage of the positive samples that are correctly identified, is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

F1-score is a classification metric where the model's harmonic average of accuracy and recall is calculated as follows:

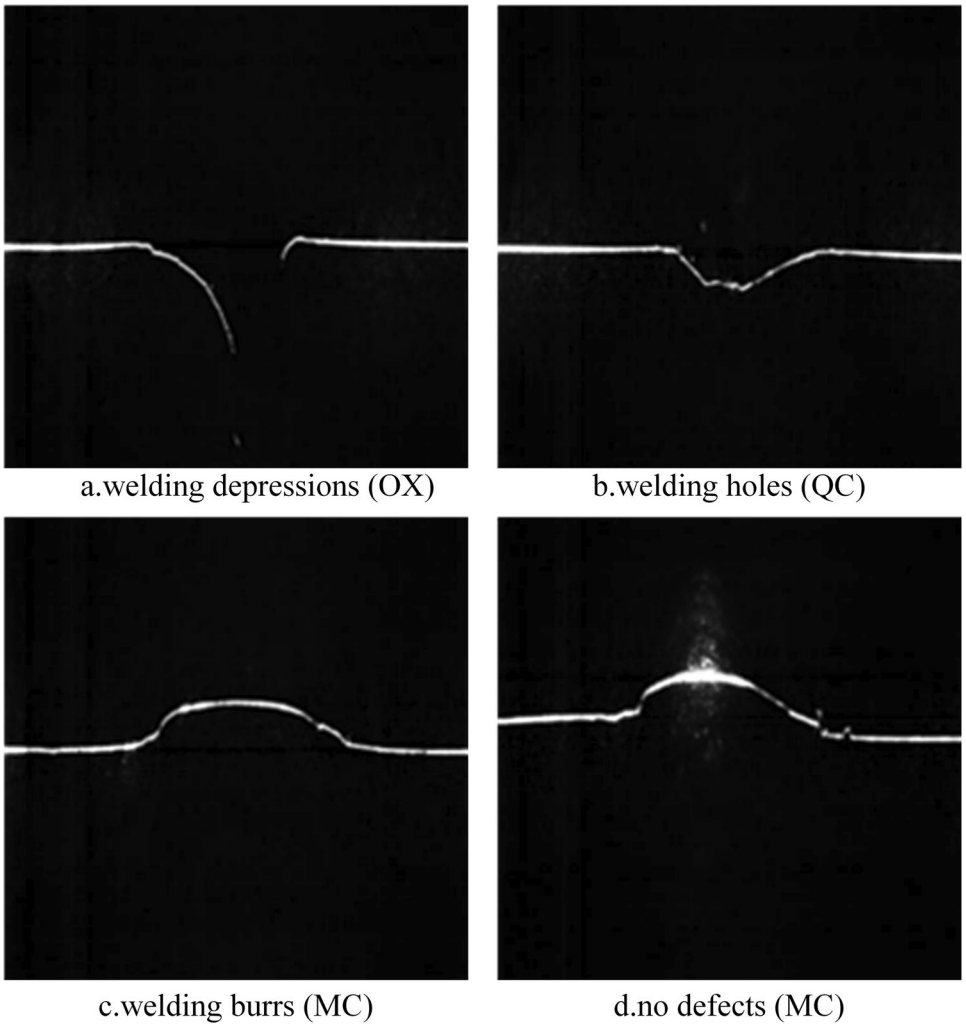

Fig. 8 Laser weld diagram of line structure

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5}$$

5 Results and analysis

This chapter uses the weld dataset that is enrichment. The training set, a subset of the entire dataset, is used to train the model, and the validation is used to make the predictions after the training is completed and verify the performance of the improved SqueezeNet network model. The obtained performance is compared using the metrics presented in Sect. 4.2.

5.1 Experimental environment and parameter settings

Training the CNN involves many calculations and configuring the corresponding deep learning environment. Thus, the following experiments are conducted on a cloud server, using an Intel Xeon Platinum 8225C CPU with 43G memory and an NVIDIA RTX2080Ti GPU with 11G video memory. In the CUDA environment, the PyTorch deep learning framework PyTorch 1.11.0 is used to build a weld defect identification network combined with Python3.8.

The input image size of the network is 224×224, the sample batch size is 8, the number of iterations is 200, and the learning rate is 0.002 to train the processed dataset. During training, the adaptive moment estimation (Adam) algorithm is used to dynamically adjust the learning rate of the parameters so that the parameter change range is not too large and the parameter training is more stable.

5.2 Ablation experiments

The effect of specific improvement methods on our method's defect identification performance is investigated through an ablation study that involves five experiments. The first experiment utilizes the original SqueezeNet model as the

baseline network [34]. The second experiment employs SqueezeNet by adding a residual module, and the third trial modifies the fire module to adopt the improved deep separability method. The fourth experiment incorporates the ECA channel attention mechanism in the model, and the fifth experiment combines the residual module, the improved fire module, and the ECA channel attention mechanism with the original SqueezeNet model. To enhance clarity, these five models are SqueezeNet0, SqueezeNet1, SqueezeNet2, SqueezeNet3, and SqueezeNet4.

This paper employs a series of ablation experiments to assess the impact of various structural enhancements on model performance. According to the data in Table 1, SqueezeNet0 has 738,502 parameters. By incorporating a residual module, SqueezeNet1 sees a slight increase to 744,262 parameters. SqueezeNet2, utilizing an improved depth-separable method, significantly reduces the parameter count to 449,862. SqueezeNet3, which introduces the ECA channel attention mechanism, maintains a parameter count comparable to SqueezeNet0, while SqueezeNet4, which integrates multiple improvements, has the lowest at 255,947. In terms of computational complexity, SqueezeNet1 and SqueezeNet3 are on the higher end, whereas SqueezeNet2 and SqueezeNet4 demonstrate lower FLOPs, particularly SqueezeNet4, where computational complexity is notably reduced to 356.06M, showcasing extremely high computational efficiency. Regarding accuracy, SqueezeNet4 achieves the highest performance with 98.00%, while SqueezeNet1 and SqueezeNet2 also achieve commendable accuracies of 97.00% and 97.75%, respectively. These findings illustrate that through structural improvements and technological integration, the SqueezeNet model's performance can be effectively enhanced, reducing computational complexity and increasing classification accuracy.

Table 1 Comparison of the improved model effects

Method	Params	Flops/M	Accuracy/%
SqueezeNet0	738,502	732.92	94.78
SqueezeNet1	744,262	745.06	97.00
SqueezeNet2	449,862	510.63	97.75
SqueezeNet3	738,507	733.20	95.50
SqueezeNet4	255,947	356.06	98.00

5.3 Model comparison experiments

We challenge our method against other common classification models for a more comprehensive comparison. Specifically, we compare the final improved SqueezeNet network, i.e., SqueezeNet4, against ResNet, Inceptionv4, and MobileNetv3. The prediction effect is compared after 200 training iterations under the same platform and hyperparameters.

As shown in Table 2, SqueezeNet4 outperforms SqueezeNet0 across all performance metrics due to structural improvements, notably reducing computational complexity to 356.06M Flops. In comparison to traditional and larger models such as ResNet and Inceptionv4, which provide stable performance, they also require substantial computing resources. Among them, Inceptionv4 has the highest computational demand, reaching 6153.58M Flops. Compared with MobileNetv3, SqueezeNet4 maintains high accuracy and is further optimized considering parameter quantity. Indeed, SqueezeNet4 has 0.26M parameters, proving it is lightweight.

Based on the above comparative data, the improved SqueezeNet significantly reduces the number of parameters and floating-point operations while ensuring high recognition accuracy and precision. Overall, the improved SqueezeNet is very convenient for lightweight deployment.

5.4 Visual analysis of species identification results

After training the improved SqueezeNet, it is tested on the weld defect dataset. The confusion matrix in Fig. 9 shows the relationship between the true labels and predicted labels for the four welding defect categories (MC, OX, QC, and WD). The rows of the matrix represent the true labels, and the columns represent the predicted labels of the model. The values on the diagonal (47, 49, 50, 50) represent the number of correct predictions for each category. For example, 47 samples in the MC category are correctly predicted, 49 in the OX category, and 50 samples in the QC and WD categories are completely correctly predicted. The off-diagonal elements in the matrix represent the number of prediction errors. For example, 3 samples in the MC category are incorrectly predicted as WD, and 1 sample in the OX category is incorrectly predicted as MC. The results infer that the

Table 2 Comparison of different models

Model	Precision/%	Accuracy/%	Recall	F1-score	Params/M	Flops/M
SqueezeNet0	95.00	94.78	95.50	95.24	0.74	732.92
SqueezeNet4	98.06	98.00	98.00	97.99	0.26	356.06
ResNet	96.51	96.48	96.50	96.50	25.31	4131.69
Inceptionv4	97.50	97.95	98.32	97.91	41.14	6153.58
MobileNetv3	96.01	97.65	97.00	96.50	4.20	232.96

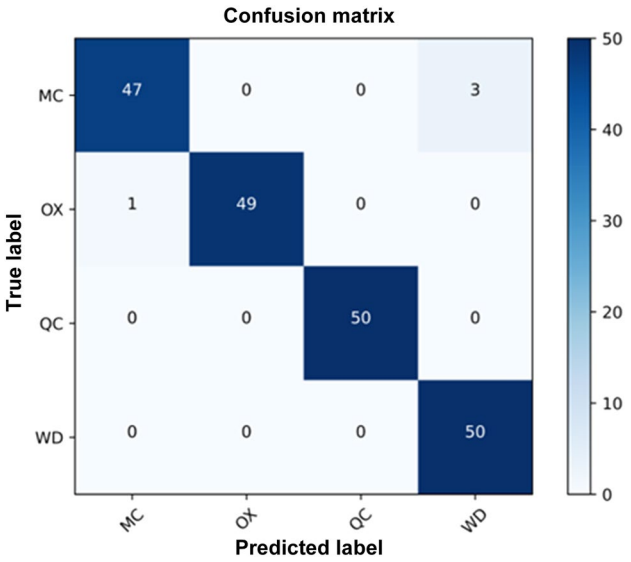

Fig. 9 Confusion matrix

maximum prediction classification is on the diagonal, indicating that most defect features are successfully predicted, verifying the effectiveness of the improved model for weld defect identification.

According to the confusion matrix, the accuracy and recall of the improved SqueezeNet model in various welds can be calculated for specific quantitative analysis (Table 3).

Table 3 reveals that the overall prediction results of the improved model on the weld defect test set are better. The accuracy and recall rate is relatively high, among which the recognition accuracy of welding burrs (MC) is the lowest (94%), and the recognition accuracy of welding holes (QC), and no defects(WD) is relatively high (100%).

5.5 K-fold cross validation

Cross-validation is a statistical analysis technique employed to assess the generalization capability of a model, specifically its proficiency in predicting unseen data. This method facilitates a more comprehensive evaluation of the model's performance across various subsets, thus enhancing its reliability and accuracy.

Table 3 Weld defect classification test results

Category	Number	Precision/%	Recall/%
MC	50	94.00	97.92
OX	50	98.00	100.00
QC	50	100.00	100.00
WD	50	100.00	94.34

K-fold cross-validation is a widely utilized method for validating models, particularly for assessing the generalization capability of statistical models. In this approach, the dataset is divided into K equal-sized subsets. The process involves several key steps: firstly, the entire dataset is randomly split into K non-overlapping subsets, each approximately of the same size. Subsequently, one of these subsets is selected as the test set, while the remaining K-1 subsets serve as the training set. The model is then trained on the training set and evaluated on the test set. This procedure is repeated K times, each time with a different subset acting as the test set and the others as the training set. Ultimately, the model's performance is determined by the average of the results from the K separate tests. This method ensures a comprehensive and fair evaluation of the model by allowing each data point an equal chance to be tested.

In this article, the K value for the K-fold cross-validation method is set to 5. *Accuracy* and *macro-F1 scores* are utilized to evaluate the outcomes of the cross-validation process.

Table 4 presents the results of a K-fold cross-validation test conducted using the SqueezeNet4 model, with K values ranging from 1 to 5. As K increases, both the accuracy and F1-score metrics show a trend of improvement, suggesting enhanced model performance with larger K values. Specifically, when K is set to 1, the model achieves an accuracy of 94.75% and an F1-score of 95.00%. These metrics improve progressively with each increase in K; by the time K reaches 5, accuracy has risen significantly to 97.75%, and the F1-score to 97.00%. This improvement can be attributed to the model having more comprehensive training and validation across the different subsets of data, thereby reducing variance and bias in the evaluation process.

5.6 Specific application

The lightweight welding defect identification algorithm based on a convolutional neural network, proposed in this article, will be implemented in our welding robot to inspect weld quality both during and after the welding process. The welding robot, depicted in Fig. 10, includes components such as a mechanical arm, welding gun, camera, weld sensor, and a four-wheel differential chassis. An Advantech 276

Table 4 K-fold cross validation test results

Model	K	Accuracy/%	F1-score/%
SqueezeNet4	1	94.75	95.00
	2	95.50	95.00
	3	95.50	95.00
	4	96.50	96.00
	5	97.75	97.00

Fig. 10 Schematic diagram of welding robot

industrial computer serves as the host to control the robot's welding operations. Since the welding control program, various dependency packages, and video monitoring resources are all stored on this industrial computer, the memory available to support the welding defect identification model is limited. Despite these limitations, the proposed welding defect model can meet the real-time requirements for monitoring weld quality throughout the welding process.

6 Conclusion

This paper proposes a weld defect recognition algorithm based on a CNN for the problem of weld defect recognition in the welding process. Based on the SqueezeNet network, the algorithm improves the fire module and adds the residual structure and ECA attention channel mechanism to the network, demonstrating that the improved model can effectively identify four types of weld defects.

Data availability The datasets generated during and/or analysed during the current study are available in Datasets-weld at https://github.com/daureliano/Datasets-weld.

Declaration

Conflict of interest All authors disclosed no relevant relationships.

References

1. Zhang R, Wu S, Tu Q et al (2015) Design and application of high spatial resolution distributed temperature sensing system. Optical Instrum 37(1):79–82
2. Zhang X, Meng Z, Hu Z (2011) Sensing system with Michelson-type fiber optical interferometer based on single FBG reflector. Chin Opt Lett 9(11):73–75
3. Qiu G, Wang L, Bai L (2021) GANs-based synthetic data augmentation for defects recognition. J Electron Meas Instrum 35(2):212–220
4. Li S, Yuan W, Yang J et al (2019) Wood defect classification based on local binary difference excitation pattern. Chin J Sci Instrum 40(6):68–77
5. Yu Y, Yin G, Yin Y et al (2014) Defect recognition for radiographic image based on deep learning network. Chin J Sci Instrum 35(9):2012–2019
6. Liu H, Guo R (2018) Detection and identification of SAWH pipe weld defects based on X-ray image and CNN. Chin J Sci Instrum 39(4):247–256
7. Malarvel M, Sethumadhavan G, Bhagi RCP et al (2017) An improved version of Otsu's method for segmentation of weld defects on X-radiography images. Optik Int J Light Electron Opt. 142:109–118
8. Jiang H, Zhao Y, Gao J, Wang Z (2016) Weld defect classification based on texture features and principal component analysis. Insight 58(4):194–200
9. Jiang H, Wang R, Gao Z et al (2019) Classification of weld defects based on the analytical hierarchy process and Dempster–Shafer evidence theory. J Intell Manuf 30(4):2013–2024
10. Goumeidane AB, Bouzaieni A, Nacereddine N et al (2015) Bayesian networks-based defects classes discrimination in weld radiographic images. In: Tsapatsoulis N et al (eds) Computer analysis of images and patterns: part II. Springer, Cham, pp 554–565
11. Zhou H, Lu J, Lu J et al (2018) Research on weld defect type identification method based on multi-scale texture features. Electromech Technol 3:14–16
12. Xu H (2017) Weld identification and trajectory planning based on machine vision. Guangxi University, Nanning
13. Lu H, Zhang M, Liu Y et al (2017) Convolution neural network feature importance analysis and feature selection enhanced model. J Softw 28(11):2879–2890
14. Chen J, Li Z, Wang H et al (2018) Automatic defect detection of fasteners on the catenary support device using deep convolutional neural network. IEEE Trans Instrum Meas 67(2):257–269
15. Luo H, Yang Y, Tong B et al (2018) Traffic sign recognition using a multi-task convolutional neural network. IEEE Trans Intell Transp Syst 19(4):1100–1111
16. Gu J, Xie Z, Zhang X (2020) Weld defect detection based on improved deep learning. J Astronaut Metrol Meas 40(3):75–79
17. Shao W, Liu X, Wu Z (2019) A robust weld seam detection method based on particle filter for laser welding by using a passive vision sensor. Int J Adv Manuf Technol 104(5–8):2971–2980

18. Alwzwazy HA, Albehadili HM, Alwan YS et al (2016) Handwritten digit recognition using convolutional neural networks. Int J Innov Res Comput Commun Eng 4(2):1101–1106
19. Khumaidi A, Yuniarno EM, Purnomo MH (2017) Welding defect classification based on convolution neural network (CNN) and Gaussian kernel. In: International seminar on intelligent technology and its applications (ISITIA), IEEE, pp 261–265
20. Jiao J, Li S, Chang Y et al (2017) Defect classification of weld surface in header pipe joint. Chin J Sci Instrum 38(12):3044–3052
21. Zhou S, Liu S, Yang H et al (2023) Improved U-Net robust weld seam recognition algorithm based on integrating attention mechanism. Comput Integr Manuf Syst 1–18. https://github.com/zhao-1231/References.git
22. Zhang J (2017) Welding defects recognition algorithm researching based on principal component analysis. Xi'an Shiyou University. https://github.com/zhao-1231/References.git
23. Orhan AE, Pitkow X (2017) Skip connections eliminate singularities. arXiv preprint arXiv:1701.09175
24. Liang S, Khoo Y, Yang H (2020) Drop-activation: implicit parameter reduction and harmonious regularization. Commun Appl Math Comput 3(2):1–19
25. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
26. He K, Zhang X, Ren S et al (2015) Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
27. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
28. Han S, Mao H, Dally WJ (2015) Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149
29. Liu N, Chen X (2016) Infrared image detail enhancement approach based on improved joint bilateral filter. Infrared Phys Technol 77:405–413
30. Wu J, Yang Q, Zhongliang L (2019) Face recognition based on improved SqueezeNet. Sci Technol Eng 19(11):218–223
31. Wang J, Bai Y (2022) Vehicle detection based on improved YOLOv5s. Inf Comput 34(10):80–83
32. Iandola FN, Han S, Moskewicz MW et al (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360
33. Liu Y, Yuan K, Li T et al (2022) NDT method for line laser welding based on deep learning and one-dimensional time-series data. Appl Sci 12(15):7837
34. Li D, Wang M, Liu J et al (2022) Steel surface defect recognition based on a lightweight convolutional neural network. Chin J Sci Instrum 43(3):240–248