Research Article

Deep Convolutional Generative Adversarial Network and Convolutional Neural Network for Smoke Detection

Hang Yin ^{1,2,3,4,5} **Yurong Wei** ² **Hedan Liu** ² **Shuangyin Liu** ^{1,3,4,5} **Chuanyun Liu** ²
and **Yacui Gao** ²

¹Academy of Smart Agricultural Engineering Innovations, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China

²School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China

³Smart Agriculture Engineering Research Center of Guangdong Higher Education Institutes, Guangzhou 510225, China

⁴Guangdong Provincial Agricultural Products Safety Big Data Engineering Technology Research Center, Guangzhou 510225, China

⁵Guangzhou Key Laboratory of Agricultural Products Quality & Safety Traceability Information Technology, Guangzhou 510225, China

Correspondence should be addressed to Shuangyin Liu; hdlxylq@126.com

Received 24 June 2020; Revised 24 September 2020; Accepted 26 October 2020; Published 16 November 2020

Academic Editor: Zhile Yang

Copyright © 2020 Hang Yin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Real-time smoke detection is of great significance for early warning of fire, which can avoid the serious loss caused by fire. Detecting smoke in actual scenes is still a challenging task due to large variance of smoke color, texture, and shapes. Moreover, the smoke detection in the actual scene is faced with the difficulties in data collection and insufficient smoke datasets, and the smoke morphology is susceptible to environmental influences. To improve the performance of smoke detection and solve the problem of too few datasets in real scenes, this paper proposes a model that combines a deep convolutional generative adversarial network and a convolutional neural network (DCG-CNN) to extract smoke features and detection. The vibe algorithm was used to collect smoke and nonsmoke images in the dynamic scene and deep convolutional generative adversarial network (DCGAN) used these images to generate images that are as realistic as possible. Besides, we designed an improved convolutional neural network (CNN) model for extracting smoke features and smoke detection. The experimental results show that the method has a good detection performance on the smoke generated in the actual scenes and effectively reduces the false alarm rate.

1. Introduction

In recent years, frequent fires have caused tremendous losses to human life and society. Smoke is a characteristic of the initial stage of fire. If the smoke can be detected and alarmed as soon as possible, it can effectively reduce the incidence of fire. Therefore, it is of considerable significance to detect whether smoke is generated accurately.

Traditional smoke detection methods are mostly based on sensors, and there is a strict requirement on the number of sensors and the location of installation. Sensors need to be installed near fire-prone places. Alarms will alert by detecting smoke particles in the air. The disadvantage is that

it would take a long time to detect smoke. It will result in too long response time and the purpose of real-time monitoring will not be achieved.

As a result, many algorithms based on image smoke detection have been proposed. Most algorithms are based on manually extracting smoke features. For example, they used smoke motion, color, and edge blurry features for smoke detection. Gubbi et al. [1] generated features from each subband of three-level wavelet transformed images for smoke recognition, including arithmetic mean, geometric mean, standard deviation, skewness, kurtosis, and entropy. Ye et al. [2] used adaptive background subtraction to extract motion and color features to detect smoke from image

sequences. Yuan [3] proposed several smoke detection methods, which are based on motion accumulation. Local Binary Pattern Variance (LBPV) is based on pyramids [4] and high-order local ternary patterns with locality preserving projection [5]. However, in actual scenes, the characteristics of smoke vary significantly due to environmental influences. The instability of smoke makes manual extraction of smoke characteristics less efficient. Besides, some nonsmoke objects such as pedestrians and vehicles whose colors may be similar to the smoke will affect smoke detection and result in false detection.

With the development of deep learning, convolutional neural networks, as an example of the most widely used and researched models, have been used in various fields, such as object detection [6–10], face recognition [11], and image classification [12, 13]. More and more scholars apply convolutional neural networks to smoke detection and have proposed models for smoke detection. Yin et al. [14] proposed an image smoke detection algorithm based on a deep normalized convolutional neural network. A batch normalization layer was added after the convolutional layer, which accelerated the network convergence speed and improved the accuracy. However, the recognition rate of smoke with large changes in shape is not high. Yin and Wei [15] used an improved algorithm based on cascading classification and deep convolutional neural network for smoke detection. Kaabi et al. [16] used deep belief networks to classify smoke and nonsmoke images, but the size of each frame of the smoke image will affect the effect of smoke recognition. Xu et al. [17] trained a convolutional neural network model with synthetic and real smoke images based on domain adaptation methods, which reduced the false detection rate. Nevertheless, the synthetic smoke images will affect the smoke detection performance of the training model in real scenes. Yuan et al. [18] designed a basic block of convolutional neural networks and stack basic blocks to propose a novel deep multiscale CNN for smoke recognition. Gu et al. [19] devised a new deep dual-channel neural network for smoke detection. In addition, Yin et al. [20] used WGAN for data expansion in the case of limited datasets, which can partially resolve the problem of insufficient datasets in real scenes to some extent.

The model we proposed uses the vibe algorithm to obtain part of the smoke images in the actual scenes. At the same time, it can obtain images of objects that will cause false alarms in the actual scenes. These images can be organized and classified into the different datasets including smoke images and nonsmoke images. Then the DCGAN uses them to expand the datasets. The smoke in different environment has variety of shapes due to the different wind directions, temperature, and background environment. DCG-CNN has fully considered the various smoke forms in the actual scene rather than the smoke in the perfect ideal state, so it can get different smoke characteristics, extract the in-depth features of smoke, and then use them for further smoke detection.

This paper mainly includes the following contributions:

- (1) The vibe algorithm is used to obtain the original datasets including smoke images and nonsmoke

images from the actual scene. The multiscale, multiform smoke images make the performance of the smoke detection better.

- (2) This paper proposes the improved DCGAN to extend the smoke images and nonsmoke images and adjusts the network to generate images that are more similar to the actual scene according to different features of smoke.
- (3) We optimized the relevant quantities of the CNN (convolution kernel settings, activation function, learning rate, and batch size) in order to improve the detection rate of the model compared with other traditional models.

The remainder of the paper is organized as follows. Section 2 introduces the related work based on vibe algorithm, DCGAN, and CNN. Section 3 presents network architecture and model training of DCG-CNN in detail. Experiments and results are demonstrated in Section 4. Finally, Section 5 gives the conclusions.

2. Related Work

2.1. Vibe Algorithm. It is well known that the vibe algorithm has good applications in moving object detection. After the smoke is generated, the form concentration of the smoke will change with environmental factors such as wind direction temperature. The threshold can be set to control the conditions for generating the alarm. Due to the fixed background of the actual scene, we can use the vibe algorithm [21] to monitor and extract the motion area from the background. Each background point has a value of $1/\varphi$ probability to update model sample and neighbor point model sample. When the model of pixel change cannot be determined, the random updating strategy can simulate the uncertainty of pixel change to some extent. Assume that the time is continuous, and the probability that the sample value remains after dt time is expressed as follows:

$$P(t, t + dt) = e^{-\ln(N/N-1)dt}. \quad (1)$$

Different from the traditional vibe algorithm, we can make use of the background for static feature extraction and classification because the smoke has a practice process from generation to the disappearance. Vibe algorithm can not only ensure the continuity of the pixel model in time but also ensure the principle of spatial consistency. Because the smoke has the characteristics of continuous diffusion, by comparing two adjacent frames of images, the original background model obtained by vibe and frame difference is used to update the background frame by frame.

The foreground extraction of traditional vibe algorithm may have missed detection or false detection due to the changes of the environmental factors. As a result, the improved vibe algorithm was proposed to obtain the real background by setting adjusting parameters in mean background modelling and setting an adaptive threshold according to the actual scene changes to detect adaptively the foreground during the foreground detection process. The

improved vibe algorithm has been proved to detect the dynamic area in the actual scene better and meet the real-time requirements.

Figure 1 shows the flow chart of the vibe algorithm. In the actual scene, a wide-angle camera is used for real-time monitoring, and the vibe algorithm will alert when a dynamic target is detected in the scene. We set the threshold to alarm when there is a suspected smoke object in the scene. If there is suspected smoke in the dynamic area of the scene, a prompt box will appear, as shown in Figure 2. We cannot get the detailed characteristics of the smoke because the camera is far away from the dynamic area and the pixels of the extracted dynamic are small. As a consequence, the vibe algorithm performs coordinate conversion after detecting the dynamic target, and a zoom high-speed-dome camera was used to capture images located in the target moving area. These images are images of smoke or other dynamic objects that cause an alarm, such as a moving vehicle or pedestrian.

Wind direction, wind speed, light, and other factors in the actual environment will have a great impact on the smoke and change its density and texture, thus increasing the difficulty of smoke detection. Figure 3 shows the smoke or other dynamic objects' images affected by different environmental factors. The smoke detection trained by the ideal smoke images could not have an excellent detection effect in actual scene. Besides, the number of actual smoke datasets is not enough to train a stable and robust smoke detection model. The generative adversarial network (GAN) [22] has been experimentally proven to have a good effect in solving the problem of small sample datasets and datasets' imbalances. So we consider using GAN to generate part of the smoke images similar to the smoke pattern in the actual scene. Then we mix the generated images and original images as training datasets and train the CNN model to extract smoke features and detect smoke appearing in the actual scene. In this way, we can solve the problems of few samples and unbalanced samples of the smoke datasets.

FIGURE 1: The flow chart of the vibe algorithm. The camera monitors the actual scene and the vibe algorithm detects the moving object in actual scene. It will alert if detecting the moving objects suspected of smoke.

2.2. Results and Discussion of GAN and DCGAN Models. The generative adversarial network (GAN) consists of two modules, the generator model (G) and the discriminator model (D). The best network performance is achieved by the two models fighting against each other. Figure 4 shows GAN model composition. The purpose of the generator network is to confuse the antagonist, and the purpose of the discriminator is to distinguish between the generated datasets and the original datasets. A generation network composed of multilayer perceptrons is followed by a discriminant network composed of multilayer perceptrons. The input of the discriminator selects real samples or the output of the generator network. The output of the discriminator is the probability that the input picture is real. When the discriminator network distinguishes whether the output of the generator is a real sample or not, it can indicate what kind of sample is closer to the real sample through the gradient and then adjust the generation network through this information. The function of the GAN is expressed as follows:

$$\min \max V(D, G) = E_{x \sim P_{\text{data}}(x)} \left[\log D\left(\frac{x}{y}\right) \right] + E_{z \sim P_z(z)} \left[\log \left(1 - D\left(G\left(\frac{z}{y}\right)\right) \right) \right]. \quad (2)$$

However, GAN will have problems such as instability during training process. Compared with the original GAN, DCGAN uses convolution layer with stride instead of upsampling layer and convolution layer instead of a fully connected layer, which can play a better performance in extracting smoke image features. Almost every layer in the generator and discriminator uses the batch norm layer to normalize the output layers of the features, which speeds up the training and improves the stability of the training process. Moreover, the leaky ReLU activation function is used in the discriminator to prevent gradient sparseness. Figure 5 shows the network structure diagram of the generator.

It would be difficult to detect smoke as the shape of smoke is changeable, and the smoke concentration will drop

due to the diffusion of smoke. In addition, good smoke detection results require high quality of smoke datasets. Thus, a new loss function is added to the middle layer feature map to accelerate the convergence of the balance and the DCGAN will make use of the virtual batch normalization. It can reduce the dependence of samples in a mini batch on other samples by using fixed reference mini batch samples determined at the beginning of training to calculate normalized batch statistics. These can ensure the diversity and clarity of the generated smoke images in order to achieve better experimental results.

Therefore, we select the improved DCGAN to generate images. Only the generated images are closer to the images obtained in the actual environment, and they could be used as the training datasets of the model training process. The

FIGURE 2: Actual scene image. The vibe algorithm monitors the actual scene in real time; once the dynamic area is detected, a prompt box will appear as shown in the figure.

FIGURE 3: Collected images by the vibe algorithm. (a), (b), and (c) are the smoke images affected by the different environmental factors and (d), (e), and (f) are the nonsmoke images collected in the actual scene.

FIGURE 4: GAN model composition. The generator model generates the images and the discriminator distinguishes whether the output of the generator is a real sample or not.

FIGURE 5: Network structure of the generator of DCGAN. The generator uses convolution layer with stride instead of upsampling layer and convolution layer instead of a fully connected layer.

experimental results prove that the detection rate is greatly improved by the generated smoke images with high quality.

2.3. Convolutional Neural Network. The convolutional neural network is a deep feedforward neural network that extracts features by learning the input picture layer by layer. CNN uses a convolution kernel to extract features, and it contains a three-layer structure, convolutional layers, pooling layers, and fully connected layers. Different layers have different functions. These functional layers are composed of many neurons, and each neuron connects only a part of the neurons in the adjacent layer. It reduces the complexity of the network and improves the calculation efficiency.

The convolutional layer consists of some convolutional neurons and is considered as a feature extraction layer. The size of the convolution kernel will determine the size of the output feature map. After the kernel function is convolved and transported, the size of the feature map is calculated by the following equation:

$$\begin{cases} N_x^l = \frac{N_x^{l-1} - K_x^l + 2P_x^l}{S_x^l}, \\ N_y^l = \frac{N_y^{l-1} - K_y^l + 2P_y^l}{S_y^l}, \end{cases} \quad (3)$$

where l is the current number of layers, K is the size of the convolution kernel, P is the fill pixel value, and S is the step size. After the convolution operation, it is a nonlinear transformation of the activation function. The back-propagation algorithm obtains the weights and parameters of each neuron, and the expression of the convolutional layer neuron is as follows:

$$x_j^l = \text{Relu} \left(\sum_{i \in M_j} x_i^{l-1} w_{ij}^l + b_j^l \right), \quad (4)$$

where M is the filter size and w and b represent the connection weight and offset, respectively.

The pooling layer performs feature mapping, usually between two convolutional layers. Pooling operations include maximum pooling and average pooling. In [16], overlapping application pools are also proposed. By reducing the variance of the converted data through the subsampling layer, the values of specific features in the input layer can be calculated and combined, which can reduce the number of neurons while keeping the features unchanged. The equation of the pooling layer is as follows:

$$y = \max(x_i), \quad x_i \in x, \quad (5)$$

where x is a region of the feature map and x_i is the output of the neuron in this region.

The fully connected layer connects all the neurons of the previous convolutional layer to the current layer and converts all local features to global features. Our network contains three fully connected layers at the end of the neural network. Fully connected layers are prone to overfitting problems. To overcome this problem, we use the dropout function to reduce the overfitting of the first two layers. The last fully connected layer is the output layer, and we replace it with an output layer containing two neurons, indicating the probability that the output result is smoke or nonsmoke. The probability is defined by the softmax function as follows:

$$y_j = \frac{\exp(f_j)}{\sum_{i=1}^2 \exp(f_i)}, \quad j = 1, 2, \quad (6)$$

where y_j is the output probability of the j th neuron.

3. Network Architecture and Model Training

3.1. Network Architecture. The DCG-CNN network architecture has been proposed, combining improved DCGAN and CNN to expand smoke images and extract smoke features more comprehensively. Figure 6 shows the DCG-CNN framework. The smoke images from the datasets were put into the DCGAN in order to generate smoke images with various shapes and textures. Besides, the other images that could cause false detection were also generated.

The CNN structure consists of convolutional layers, pooling layers, and three convolutional layers. The convolutional layers extract smoke features by layers of learning

FIGURE 6: DCG-CNN framework. The images were input to the DCGAN. Then, CNN trains and tests the network by the labelled datasets and classifies the smoke and nonsmoke images.

smoke images. The pooling layers perform feature dimensionality reduction, compressing the amount of data and number of parameters without affecting the feature data. The fully connected layer integrates local feature map to realize classification function. The last layer of the fully connected layer contains two neurons, indicating whether it is a smoke image. We adjust the network structure parameters so that the model has a reasonable detection rate and a low false alarm rate for the smoke that appears in the actual scenes.

3.2. Network Training. DCGAN uses the Adam optimizer. In the training process, the discriminator is usually trained first, and then the entire DCGAN is trained. Besides, the discriminator cannot update the parameters during training process, so all layers must be frozen. In order to speed up network training, we train multiple discriminators and train generators again, which can improve the entire network training rate.

In order to ensure the balance of the generated datasets and the quality of the smoke images, we adjust the proportion of the smoke images that have a variety of shapes and textures during the DCGAN training process. In addition, we set different epochs and observe the generated smoke images after several epochs for the most suitable smoke images.

CNN uses stochastic gradient descent (SGD) as the optimizer of the CNN model. The model is updated based on a mini batch instead of a single sample, so we can choose to update the variance to make the convergence more stable. The learning rate can control the rate of convergence, and we select a model learning rate of 0.01 in the experiment. SGD combines the gradient and the updated weight of the previous iteration to update the network model weights; the whole process can be expressed in the two following equations:

$$V_{t+1} = \mu V_t + \alpha \nabla L(W_t), \quad (7)$$

$$W_{t+1} = W_t - V_{t+1}, \quad (8)$$

where W_{t+1} represents the network weight after $t + 1^{\text{th}}$ times and V_{t+1} represents the network weight in $t + 1^{\text{th}}$ iterative updates.

In the process of training the network model, a suitable loss function plays a crucial role in the performance of the network. Typical loss functions currently include log loss, contrastive loss and hinge loss. We choose the cross-entropy loss function because it is more suitable for binary classification problems. Detecting smoke or nonsmoke images is a typical binary classification problem. We use it to measure the similarity of the two probability distributions of the original label and the predicted label. The cross-entropy loss function can be described as follows:

$$H(p, q) = - \sum_x p(x) \log q(x), \quad (9)$$

where $p(x)$ and $q(x)$ represent input and output labels, respectively.

We conducted experiments to demonstrate the importance of hyperparameters for network design. We used training-related hyperparameters as shown in Table 1 and set the mini batch to 32. The final optimized network hyperparameters are shown in Table 2. The settings of different hyperparameters play a significant influence on our network. With the hyperparameters described in Tables 1 and 2, the CNN achieves good detection rate. In summary, overlapped max pooling achieves better performance than non-overlapped max pooling layers. Besides, appropriately reducing the number of neurons in fully connected layers not only shortens the convergence time but also improves the recognition ability for detection.

4. Experiments and Results

4.1. Experimental Datasets. Too few training images and unbalanced distribution of training datasets will affect the training performance of deep CNNs. Therefore, we generate more data by DCGAN and add them to the training datasets to ensure sample balance. We use GAN, SS-GAN, and DCGAN to generate new training datasets based on the same original datasets. There are multiple evaluation indicators for the GANs models. We choose Frechet Inception Distances (FID) [23, 24] which are widely used as the evaluation indicators for our comparative experiments. FID represents the distance between the feature vector of the generated images and the feature vector of the real images.

TABLE 1: Training-related hyperparameters.

Hyper parameters	Initial learning rate	Mini batch	Learning rate decay coefficient	Momentum coefficient
Value	0.01	32	0.01	0.9

TABLE 2: Model-relevant hyperparameters.

No.	Network layer	Kernel number	Kernel_size	Strides
1	Convolution1	16	3	1
2	Convolution1	16	3	1
3	Convolution1	16	3	1
4	Maxpooling1	16	2	2
5	Convolution2	32	3	1
6	Convolution2	32	3	1
7	Convolution2	32	3	1
8	Maxpooling2	32	2	2
9	Convolution3	64	3	1
10	Convolution3	64	3	1
11	Convolution3	64	3	1
12	Maxpooling3	64	2	2
13	Fully connected	Neurons number		1024
14	Fully connected	Neurons number		1024
15	Output	Neurons number		2

When the features of the generated images and the real images are more similar, the square of the mean difference is smaller, the covariance is also smaller, and the sum (FID) is also smaller. The FID is calculated by the following equation:

$$FID(g, r) = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\sum \mathbf{r} + \sum \mathbf{g} - 2(\sum \mathbf{r} \sum \mathbf{g})^{1/2}\right), \quad (10)$$

where g and r represent the generated image and the real image, respectively, μ_g and μ_r represent the mean values of the respective eigenvectors, $\sum \mathbf{g}$ and $\sum \mathbf{r}$ represent the covariance matrices of the respective eigenvectors, and Tr represents the trace of the matrix. Besides, we use the similarity (S) between the original data and the generated data as another evaluation indicator. The definition of similarity is $S = 1/FID$ [20]. It can intuitively show the similarity between the generated images and the original images. The larger the value, the better the generation effect of the GANs.

The experimental data is shown in Table 3. The classification accuracy of DCGAN is higher than those of GAN and SS-DAN. It means that the datasets generated by DCGAN achieve a better classification performance than others. The images produced by DCGAN have a higher quality. Consequently, the improved DCGAN was used for generating images as training datasets. Figure 7 shows the loss value of the generator and discriminator of the two networks during training process. DCGAN has a better experimental performance.

We use TensorFlow and Keras [25] to build and train our framework. The hardware conditions used in the experiment are NVIDIA GTX2060 GPU and an Intel i7 quad-core CPU with 32G RAM.

In comparing the performances of the different network structures, we used the same datasets for experiments.

Table 4 shows seven datasets. Set1 has 1383 images: 552 smoke images and 831 nonsmoke images. Set2 has 1505 images: 688 smoke images and 817 nonsmoke images. Set3 and Set4 have a large number of datasets. Set3 has 2201 smoke images and 8511 nonsmoke images. Besides, Set4 has 2254 smoke images and 8364 nonsmoke images. According to the number of datasets, we used Set3 as the dataset for training the model and Set4 as the verification set. Set1 and Set2 are used as the test datasets to test the performance of the algorithms. In addition, we also selected 547 images in the actual scene named SetA containing 332 smoke images and 415 nonsmoke images. DCGAN uses these images to generate 2988 images named SetB including 1328 smoke images and 1660 nonsmoke images. We selected 1518 smoke images and 1780 nonsmoke images from SetA and SetB to construct a new balanced training set named SetC. Figure 8 shows the smoke images of the datasets. (a), (b), and (c) are the smoke images of the actual scene and the others are the generated smoke images.

4.2. Evaluation Methods. In order to compare the performance of our algorithm with the performances of the existing algorithms, we used the three indicators of detection rate (DR), false alarm rate (FAR), and accuracy rate (AR) as evaluation criteria [5]. DR represents the probability that a positive sample is correctly detected, FAR represents the probability that a negative sample is detected as a positive sample, and AR represents the probability that all samples are correctly classified, defined in the three following equations:

$$DR = \frac{TP}{P}, \quad (11)$$

$$FAR = \frac{NN}{N}, \quad (12)$$

$$AR = \frac{TP + TN}{P + N}. \quad (13)$$

Among them, TP and TN represent the number of positive samples correctly detected and the number of negative samples correctly detected, NN, P , and N represent the number of negative samples wrongly detected, the number of negative samples, and the number of positive samples, respectively. A model is considered as a good model if it can achieve a high detection rate, high accuracy, and low false alarm rate. Our goal is to train a network structure that has good performance in actual scenarios.

4.3. Experiments on Smoke Images. At the same time, in order to compare the performance of the algorithm, we compared our proposed model with some representative classical convolutional neural network models, for example,

TABLE 3: Comparative experiments between GAN, SS-GAN, and DCGAN.

Models	Training set classification accuracy (%)	Testing set classification accuracy (%)	FID	S
GAN	89.76	85.43	1.68	0.59
SS-GAN	92.36	89.24	1.34	0.74
DCGAN	95.68	93.24	0.83	1.20

FIGURE 7: The loss value of the discriminator and the generator. (a) is the loss value of the discriminator and (b) is the loss of the generator. Results show that DCGAN has a better performance compared with GAN.

TABLE 4: Datasets for smoke detection.

Datasets	Number of smoke images	Number of nonsmoke images	Total number of images	Purpose for
Set1	552	831	1383	Testing
Set2	688	817	1505	Testing
Set3	2201	8511	10712	Training
Set4	2254	8364	10617	Validation
SetA	332	415	747	—
SetB	1328	1660	2988	—
SetC	1518	1780	3298	Training

AlexNet [26], VGG16 [27], and ZF-Net. Moreover, we compared our algorithm with DNCNN [10], which has been experimentally verified to have good experimental results. Because each network architecture has different requirements for the input images, we adjusted the size of the input images to meet the input requirements of each network model.

The experimental results are shown in Table 5. We can see that, on the Set1 and Set2 datasets, our AR is higher than those in the other classical convolutional neural networks. Although our detection rate is lower than that of VGG16 on Set1, our false detection rate is better than it. The false detection rate of AlexNet is very close to that of our model on Set1 and Set2, but our AR and FAR are better than those of AlexNet. The AR of ZF-Net on Set1 and Set2 is well, but the FAR is higher than others. The DR of DNCNN performs better than our model on Set1 and Set2; however, our AR and FAR have a little advantage over it. Our network model performs well compared with it.

The training process of different networks is shown in Figure 9. Figure 9 shows that our network converges faster and achieves an accuracy rate close to 1.0 at 70 epochs. At the same time, the accuracy of other networks is not higher than 0.95 and is stabilized at 200 epochs except DNCNN. We can see the same trend on the validation dataset. Our network reached 0.95 at 10 epochs and stabilized, while the other networks reached an accuracy rate higher than 0.95 after 100 epochs. This implies that our network consumes less computing resources and has faster convergence to extract smoke features with better performance for smoke detection.

In short, our DCG-CNN performs better than VGG16, AlexNet, ZF-Net, and DNCNN based on the same datasets. It is indicated that our network structure performs better and reduces false alarms caused by moving objects effectively. Moreover, our learnable parameters are less than other networks, which means that our network model is more efficient. It is meaningful for the real-time performance of smoke detection.

FIGURE 8: Smoke images. (a), (b), and (c) are the smoke images of the actual scene and (d), (e), and (f) are the generated smoke images. (a) is the smoke in the morning breeze, (b) is the smoke under strong light and breeze at noon, and (c) is the smoke in which wind speed is fast and wind direction is uncertain.

TABLE 5: Experimental comparison results.

Deep CNNs	Set1			Set2		
	DR (%)	AR (%)	FAR (%)	DR (%)	AR (%)	FAR (%)
VGG16	96.19	97.48	0.60	95.05	97.48	0.48
AlexNet	93.28	97.18	0.26	96.03	97.01	0.32
ZF-Net	93.66	96.82	1.08	93.60	96.68	0.73
DNCNN [14]	97.83	95.28	0.48	96.36	96.08	0.48
DCG-CNN	95.72	97.65	0.32	96.65	98.02	0.24

In addition, we used the new training dataset SetC to train network models for smoke detection. Set1 and Set2 were used as the testing datasets. The experimental results are shown in Table 6. We compared our model with some CNNs, which have been proven to have a good smoke detection performance such as DNCNN [14], DMCNN [18], and DCNN [19]. The experimental results show that our network performance is slightly better based on the same datasets. Our algorithm achieves a lower FAR compared to others. It means that the balanced dataset processed by DCGAN is more conducive to training a network model with a low false alarm rate.

In order to prove the advantages of the proposed algorithm, we compared the proposed algorithm with the traditional smoke detection method consisting of manual feature extraction and classification. Smoke can be considered as a special texture and texture features have been

shown to be distinguishable in the representation of smoke. Support vector machine (SVM) is one of the most widely used classifiers. Therefore, our comparison experiment involves two methods based on texture descriptors, HLTPMC [5] and MCLBP [28]. The testing results for Set1 and Set2 are shown in Table 7.

Experimental data show that the proposed algorithm obtained slightly lower DR but higher AR on Set1 compared with HLTPMC and MCLBP. However, our method obtained lower FAR than HLTPMC and MCLBP. At the same time, our algorithm achieved good experimental performance on Set2. It is of great importance to achieve a very low FAR with acceptable DR in the actual scene. To sum up, our method outperformed HLTPMC and TPMC in terms of DR, FAR, and AR. It means that our deep learning-based algorithm clearly achieves higher detection rate and lower false alarm rate compared with traditional algorithms based on manual features.

FIGURE 9: Training and validation accuracy curves. (a) The training accuracy curves. (b) The validation accuracy curves.

TABLE 6: Comparative experiments with traditional algorithms.

Algorithms	Set1			Set2		
	DR (%)	AR (%)	FAR (%)	DR (%)	AR (%)	FAR (%)
DNCNN [14]	93.23	95.18	0.58	94.08	96.48	0.86
DMCNN [18]	94.03	94.17	0.63	94.32	95.72	1.13
DCNN [19]	94.37	95.74	0.47	95.24	96.65	0.65
DCG-CNN	95.59	95.62	0.32	95.39	97.02	0.44

TABLE 7: Comparative experiments with traditional algorithms.

Algorithms	Set1			Set2		
	DR (%)	AR (%)	FAR (%)	DR (%)	AR (%)	FAR (%)
HLTPMC [5]	96.19	97.48	4.57	96.05	97.48	2.48
MCLBP [28]	97.47	96.23	2.10	97.41	96.65	1.96
DCG-CNN	95.72	97.65	0.32	96.65	98.02	0.24

5. Conclusions

This paper proposes a deep convolutional generative adversarial network and convolutional neural network (DCG-CNN) for smoke detection. The model we proposed uses DCGAN to expand the datasets based on various shapes of smoke in the actual scenes and false-positive object images that easily affect the smoke detection rate. Then the convolutional neural network would make full use of a large number of labelled datasets for extracting smoke characteristics and smoke detection. Experimental results show that our model has better accuracy

and reduces the false alarm rate for various forms of smoke appearing in actual scenes. In addition, it can reduce false alarms caused by the movement of pedestrians, vehicles, and other objects in the environment.

Our future work will continue to focus on the feature learning and detection of smoke based on deep learning methods such as FlowNet [29]. FlowNet can reduce the impact of environmental factors on smoke detection. Next, we will consider the characteristics of specific smoke in different fixed backgrounds and explore a network model that can adaptively adjust the threshold according to the smoke datasets in different actual scenarios. Good optimization algorithm plays an important role in the improvement of network model performance. For example, monarch butterfly optimization (MBO) [30], earthworm optimization algorithm (EWA) [31], elephant herding optimization (EHO) [32], and moth search (MS) algorithm [33] have good applications in many fields. Therefore, we consider combining the new metaheuristic optimization algorithms with our network model for better smoke detection performance. We should focus on applying smoke detection to real-life or industrial environments, extending to real-time smoke detection. It is of great significance for the prevention of fire in production and life.

Data Availability

The simulation data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61871475, 61471133, and 61571444), Aeronautical Science Foundation of China (2015ZB54007), Guangzhou Innovation Platform Construction Project (201905010006), Guangdong Province Science and Technology Plan Project (2019b020215003, 2017b0101260016, 2017a070712019, and 2016a070712020), Guangdong University Engineering Technology Research Center (2017gczx0014), Guangdong Provincial Major Scientific Research Project (2016kzdxm0013), and the Scientific Research Foundation of Liaoning Education Department (L201627, L201750, and L201704).

References

- [1] J. Gubbi, S. Marusic, and M. Palaniswami, "Smoke detection in video using wavelets and support vector machines," *Fire Safety Journal*, vol. 44, no. 8, pp. 1110–1115, 2009.
- [2] S. Ye, Z. Bai, H. Chen, R. Bohush, and S. Ablameyko, "An effective algorithm to detect both smoke and flame using color and wavelet analysis," *Pattern Recognition and Image Analysis*, vol. 27, no. 1, pp. 131–138, 2017.
- [3] F. Yuan, "A fast accumulative motion orientation model based on integral image for video smoke detection," *Pattern Recognition Letters*, vol. 29, no. 7, pp. 925–932, 2008.
- [4] F. Yuan, "Video-based smoke detection with histogram sequence of LBP and LBPV pyramids," *Fire Safety Journal*, vol. 46, no. 3, pp. 132–139, 2011.
- [5] F. Yuan, J. Shi, X. Xia, Y. Fang, Z. Fang, and T. Mei, "High-order local ternary patterns with locality preserving projection for smoke detection and image classification," *Information Sciences*, vol. 372, pp. 225–240, 2016.
- [6] G. Cheng, P. Zhou, and J. Han, "Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 12, pp. 7405–7415, 2016.
- [7] T. Chen, S. Lu, and J. Fan, "S-CNN: subcategory-aware convolutional networks for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2522–2528, 2018.
- [8] X. Li and S. Wang, "Object detection using convolutional neural networks in a coarse-to-fine manner," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 11, pp. 2037–2041, 2017.
- [9] G. Cheng, J. Han, P. Zhou, and D. Xu, "Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 265–278, 2019.
- [10] G. Cheng, J. Yang, D. Gao, L. Guo, and J. Han, "High-quality proposals for weakly supervised object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 5794–5804, 2020.
- [11] K. Yan, S. Huang, Y. Song et al., "Face recognition based on convolution neural network," in *Proceedings of the 2017 36th Chinese Control Conference (CCC)*, pp. 4077–4081, Dalian, China, July 2017.
- [12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.
- [13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, Honolulu, HI, USA, July 2017.
- [14] Z. Yin, B. Wan, F. Yuan, X. Xia, and J. Shi, "A deep normalization and convolutional neural network for image smoke detection," *IEEE Access*, vol. 5, pp. 18429–18438, 2017.
- [15] H. Yin and Y. Wei, "An improved algorithm based on convolutional neural network for smoke detection," in *Proceedings of the 2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS)*, pp. 207–211, Shenyang, China, October 2019.
- [16] R. Kaabi, M. Sayadi, M. Bouchouicha, F. Fnaiech, E. Moreau, and J. M. Ginoux, "Early smoke detection of forest wildfire video using deep belief network," in *Proceedings of the 4th International Conference On Advanced Technologies For Signal And Image Processing (ATSIP)*, pp. 1–6, Sousse, Tunisia, March 2018.
- [17] G. Xu, Y. Zhang, Q. Zhang, G. Lin, and J. Wang, "Deep domain adaptation based video smoke detection using synthetic smoke images," *Fire Safety Journal*, vol. 93, pp. 53–59, 2017.
- [18] F. Yuan, L. Zhang, B. Wan, X. Xia, and J. Shi, "Convolutional neural networks based on multi-scale additive merging layers for visual smoke recognition," *Machine Vision and Applications*, vol. 30, no. 2, pp. 345–358, 2019.
- [19] K. Gu, Z. Xia, J. Qiao, and W. Lin, "Deep dual-channel neural network for image-based smoke detection," *IEEE Transactions on Multimedia*, vol. 22, no. 2, pp. 311–323, 2020.
- [20] H. Yin, Z. Li, J. Zuo et al., "Wasserstein generative adversarial network and convolutional neural network (WG-CNN) for bearing fault diagnosis," *Mathematical Problems in Engineering*, vol. 2020, Article ID 2604191, 16 pages, 2020.
- [21] O. Barnich and M. Van Droogenbroeck, "ViBe: a universal background subtraction algorithm for video sequences," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709–1724, 2011.
- [22] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 12, pp. 2672–2680, 2014.
- [23] D. C. Dowson and B. V. Landau, "The Fréchet distance between multivariate normal distributions," *Journal of Multivariate Analysis*, vol. 12, pp. 405–455, 1982.
- [24] M. Heusel, H. Ramsauer, T. Unterthiner et al., "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," *Advances in Neural Information Processing Systems*, vol. 12, 2017.
- [25] F. Chollet, Keras, 2017.
- [26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, no. 2, pp. 1097–1105, 2012.
- [27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computer Vision and Pattern Recognition*, vol. 25, 2014.
- [28] S. R. Dubey, S. K. Singh, and R. K. Singh, "Multichannel decoded local binary patterns for content-based image retrieval," *IEEE Transactions on Image Processing*, vol. 25, no. 9, pp. 4018–4032, 2016.

- [29] Z. Wang, M. Huang, Q. Zhu et al., "The optical flow detection method of moving target using deep convolution neural network," *Opto-Electronic Engineering*, vol. 48, no. 8, 2018.
- [30] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Computing and Applications*, vol. 31, no. 7, pp. 1995–2014, 2019.
- [31] G. G. Wang, S. Deb, and L. D. S. Coelho, "Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems," *International Journal of Bio-Inspired Computation*, vol. 12, no. 1, pp. 1–22, 2018.
- [32] J. Li, H. Lei, A. H. Alavi et al., "Elephant herding optimization: variants, hybrids, and applications," *Mathematics*, vol. 8, no. 9, p. 1415, 2020.
- [33] G.-G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, no. 2, pp. 151–164, 2018.