International Core Journal of Engineering Volume 7 Issue 8, 2021
ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

Design of Fire Recognition System based on ZYNQ

Zhenhan Shen

College of information science and technology, Chengdu University of Technology (College of network security, Oxford Brooks College), Chengdu, 610059, China.

Abstract

In response to some difficulties in deploying video fire early warning systems in traditional old remote locations, by introducing the concept of edge computing, a ZYNQ-based edge computing-oriented fire early warning system is proposed. The main workflow of the system is as follows: First, obtain the location information and image data of the current area in real time through the positioning module and the high-definition camera module. Subsequently, the acquired images are directly transmitted to ZYNQ for processing, and the built-up convolutional neural network model is used for fire identification. After the recognition is completed, the system will transmit the recognition results and latitude and longitude data with a very small amount of data to the network node through the Wi-Fi communication module, and the upper computer on the PC side connects to the node to obtain the data, display fire alarm and precise positioning. Finally, it realizes the efficient identification of fire on the embedded platform.

Keywords

Fire Warning; Wireless Communication; Edge Computing; ZYNQ; Hardware Acceleration.

1. Introduction

With the increasing demand for fire protection in society and the promotion of technologies such as the Internet of Things, big data and artificial intelligence, the concept of smart fire protection has gradually entered people's field of vision. As a comprehensive fire fighting plan, smart fire fighting pays more attention to the information and data interaction of various fire fighting links than traditional fire fighting methods. Adhering to the fire protection ideas of earlier detection, faster transmission, and better handling, the risk and harm of fire can be controlled to a minimum [1]. Although my country's research on smart fire protection is relatively late, with the investment of many scientific research companies and scientific and technological workers, there are currently some complete smart fire protection constructions that have been put into the market and have achieved good benefits. Based on the need to improve the comprehensive fire protection capabilities of the society, under the current background of smart fire protection, it is of great significance to explore and research smart fire warning schemes suitable for these areas [2].

Convolution Neural Network (CNN), as a widely used artificial neural network, can achieve tasks such as target detection and target recognition with high precision [3]. However, CNN achieves high accuracy at the cost of high computational complexity. In order to speed up the calculation, developers have adopted dedicated hardware accelerators for CNN deployment, such as SoC, ASIC, GPU and FPGA [4]. Although the use of GPU to accelerate neural networks is the most commonly used acceleration method, GPU-based acceleration platforms often have larger volumes and higher power consumption, so hardware platforms with smaller volumes and lower power consumption have become A popular research direction for neural network acceleration. Among them, Field Programmable Gate Array (FPGA) has become an effective solution due to its flexibility, parallelism,

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

integration, short time to market and high energy efficiency. Traditional FPGA development mostly uses hardware languages such as VHDL and Verilog, which are difficult to develop and have a long development cycle, and require developers to have a certain hardware professional knowledge background, which is difficult for software engineers. However, the use of High-Level Synthesis (HLS) reduces the difficulty of programming FPGAs, allowing software engineers to develop FPGAs faster [5]. Therefore, the design of artificial neural network unit based on FPGA is widely studied and applied. Based on the heterogeneous characteristics of the ZYNQ platform, this paper studies the FPGA-based convolutional neural network hardware acceleration technology, and uses a variety of optimization strategies to enhance the real-time nature of the fire warning system.

2. System Scheme Design

After comprehensively considering the needs of the system, the overall design of the system is carried out. In order to avoid some drawbacks caused by long-distance video transmission, this system will adopt edge computing mode, and the processing process will be completed directly in the embedded processor. According to requirements, the basic framework of this system should include five parts as shown in Figure 1, namely the location information acquisition part, the image acquisition part, the embedded processor part, the wireless transmission part and the upper computer part.

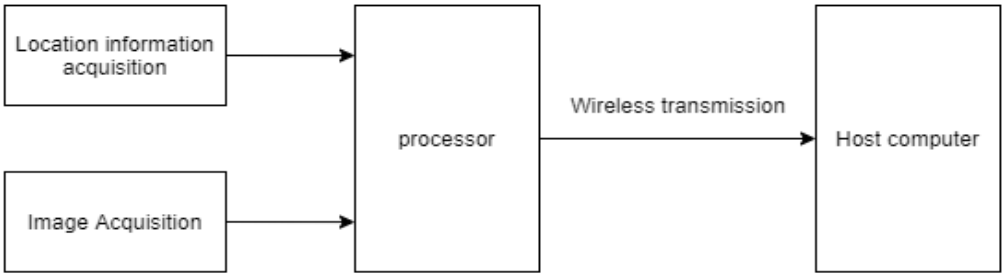

Figure 1. Basic system framework

The general flow of the system is as follows: First, the acquired position information and image data are transferred to the processor for processing. The positioning information is parsed, and the key latitude and longitude information is extracted. The image data will generate a corresponding recognition result after the recognition judgment is completed. Finally, the longitude and latitude data and the fire identification result are transmitted to the upper computer on the PC side through wireless transmission, and the upper computer displays the fire warning and fire location based on the received data.

In the fire early warning system, the important indicators for judging its performance are accuracy and real-time. Among them, the accuracy rate is greatly affected by the fire recognition algorithm. Traditional fire image recognition often requires artificial classification of fire features, so the effect of classification depends on people's experience and knowledge. When different people identify differences in fire characteristics, larger errors will be caused. The convolutional neural network algorithm relies on powerful machine learning capabilities. It only needs to input a large number of training samples into the network model to realize automatic feature extraction and learning, avoiding human interference, and thus has a higher accuracy rate.

2.1 Convolutional neural network structure

As a hierarchical model, the convolutional neural network is constructed by arranging different computing layers in a certain combination. The typical convolutional neural network structure is shown in Figure 2. After the data is input to the convolutional neural network model, it is superimposed through multi-layer convolution and pooling operations to extract the characteristics of the data, and finally realize the classification through the fully connected layer, and output the final result [6].

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

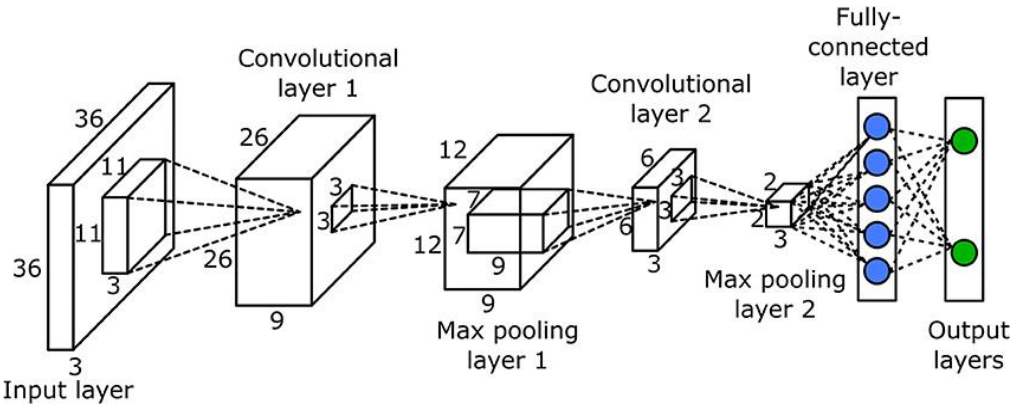

Figure 2. Convolutional neural network structure

As the basic unit of the entire convolutional neural network, the convolutional layer is also the core unit. Most of the calculations in the network are completed in the convolutional layer. In this layer, the data undergoes convolution operation to achieve feature extraction, which provides a key basis for subsequent classification. In addition, when the features of the object are to be extracted more perfectly, it is often necessary to use multi-layer convolution units to cooperate and cooperate. For example, in the first convolutional layer, low-level edge features are extracted, and subsequent higher layers will implement the extraction of more complex abstract features, such as contour and color extraction [7]. It can also be seen that when the network model is deeper, the convolution kernel's perception of the input data is larger, and the range of information that can be considered will be wider. When the input data is operated on a single convolutional layer, the convolution kernel and the corresponding input data are first multiplied. Then the calculation results are added to obtain the convolution result of each pixel position. Finally, the corresponding offset value is added, and the final output characteristic map is obtained by nonlinear activation through the activation function. The specific calculation process formula for a single convolutional layer is:

$$y = f(\sum_{j=0}^{J-1}\sum_{i=0}^{I-1} w_{ij}\, x_{m+i,n+j} + b), (0 \le m \le M, 0 \le n \le N)$$

2.2 FPGA technology and features of Zynq platform

FPGA does not depend on the instruction-level architecture, but is implemented based on a look-up table, which mainly includes look-up tables, flip-flops, programmable IO, programmable routing resources, embedded memory, and IP hard cores. FPGA has a unique parallel pipeline operation method, so that FPGA has a great operation bandwidth and data transmission rate. Due to the reconfigurability of FPGA, users can customize the algorithm architecture and interface, deploy the deep learning algorithm on the FPGA, and can be modified repeatedly without changing the FPGA chip, which is suitable for the deployment and deployment of continuously iteratively updated deep learning algorithms. Development.

The Zynq platform is a programmable SoC developed by Xilinx. Unlike traditional FPGA products, the Zynq platform not only includes the FPGA chip, but also the ARM processor, and provides multiple interfaces such as AXI bus between the two parts [8]. Zynq combines the advantages of an ARM-based embedded platform and FPGA. It can not only easily deploy an operating system and run software programs on ARM, but also use the parallel characteristics of FPGA to accelerate the parallel accelerated parts of the software and improve the overall The operating efficiency of the platform, while retaining the embedded platform's small size, low energy consumption, convenient deployment and migration characteristics. However, there are often more linear structures in neural networks, and the operating efficiency will be higher on ARM. This fits the advantages of the Zynq platform. Therefore, the Zynq platform is a hardware platform that is very suitable for building embedded neural network units [9].

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

3. System implementation

Since this system needs to implement fire recognition through the convolutional neural network algorithm on ZYNQ, and only needs to implement the inference process, the first step is to perform offline training of the convolutional neural network through the fire image data set on the PC side, and The model parameters after training, that is, the weight values and parameter values of each layer, are extracted for subsequent porting to the ZYNQ platform.

3.1 Hardware design

3.1.1 Image acquisition circuit design

When the fire image is recognized, the accuracy of the recognition result is often greatly affected by the image quality. In order to make the fire early warning system have a higher accuracy, this design selects the IMX222 camera as the system's image acquisition equipment. According to the data manual recommendation, the input clock is 37.125MHZ. This camera is a high-end brand of Sony, with a maximum resolution of 1920*1080 (1080P), and an equivalent pixel of about 2 million pixels, which has a good imaging effect.

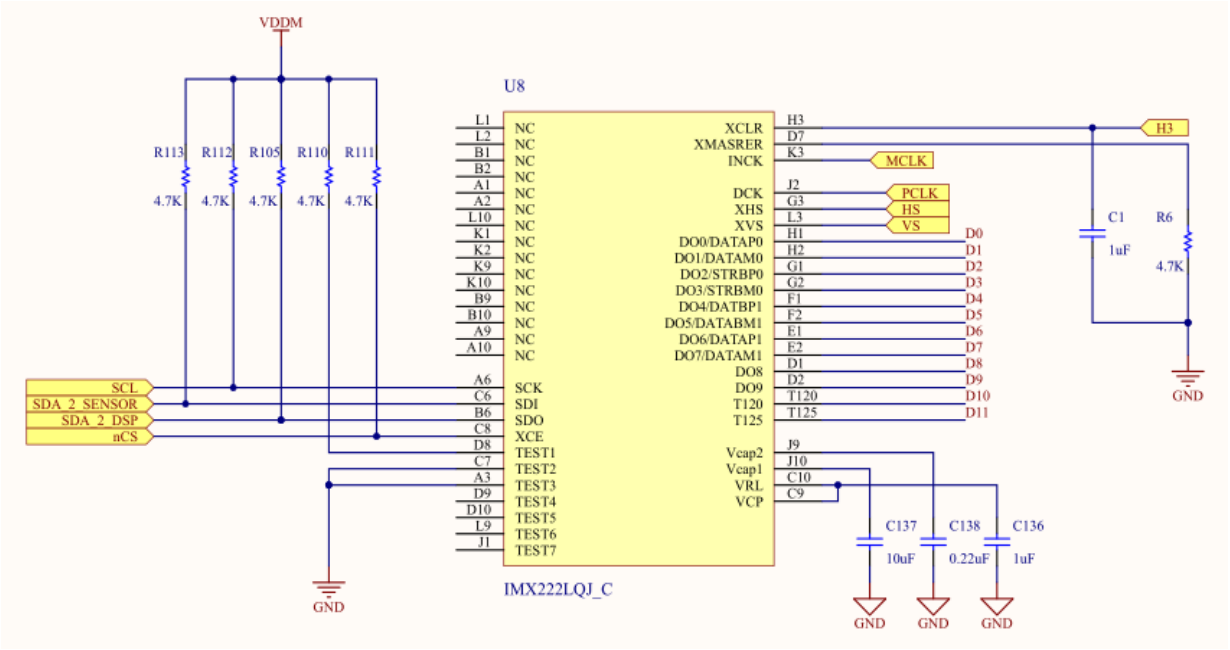

Figure 3. IMX222 module schematic diagram

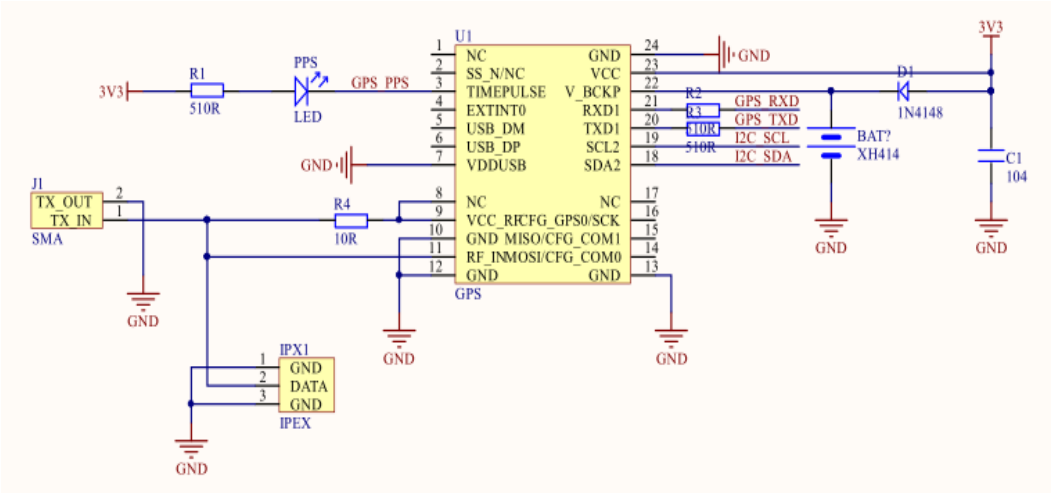

Figure 4. ATGM332D module schematic diagram

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

3.1.2 Positioning module circuit design

In order to better locate the fire location, the system will obtain the location information of the current monitoring area through the positioning module. This design uses the ATGM332D series positioning module, and the positioning message information can be directly output through the serial port. The working voltage of the module is 3.3V, and it can support Beidou and GPS satellite navigation systems. The cold start capture sensitivity is -148dBm, the tracking sensitivity is -162dBm, and the positioning accuracy can reach 2.5 meters. It also has a built-in antenna short-circuit protection function. The advantages of high sensitivity, low power consumption and high stability. The circuit schematic diagram of the ATGM332D module is shown in Figure 4

3.2 Convolutional neural network model training

In this design, a cross-validation method is used for the training of the convolutional neural network to obtain the optimal parameters of the model. The 20,000 fire sample data sets were randomly divided into two groups, of which 4,000 were used as the validation set and the remaining 16,000 formed the training set. The initial learning rate was set to 0.001 and the batch_ size was set to 128, while the maximum number of training sessions was set to 200. Figure 5 show the changes of the loss rate and accuracy of the network model as the number of training sessions increased, respectively. From the loss rate graph in Figure 5, we can see that the loss rate of the validation set only fluctuates slightly after 60 training sessions, which proves that the effective data set expansion can prevent overfitting well. The accuracy change on the right side of the graph shows that after 60 training sessions, the accuracy of the validation set has not changed significantly, and after 185 training sessions, when the accuracy of the training set tends to 1, the accuracy of the validation set remains at around 0.916. This shows that the model has been trained and the weight and bias values have been determined.

Figure 5. Change in accuracy

After the model has been trained, the final model parameters need to be exported because the algorithm has to be ported to a hardware platform for implementation. The parameters are saved in an h5 file via TensorFlow's own save function, and then the parameters of the specified layer are converted into an array form and finally printed out in a loop[10].

3.3 Convolutional neural network hardware acceleration design

After completing the above software design, the project file was compiled and the resulting BOOT. BIN file was downloaded to the ZYNQ board for testing, at which point the convolutional neural

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

network algorithm was run in pure software on the ARM only. According to the results of the actual test, the system takes about 27 seconds for a complete response, which is poor in real time. Based on the demand for real-time performance of the fire warning system, this design will optimise the convolutional neural network through the hardware acceleration technology of FPGA in order to improve the response speed of the system. By analysing the percentage of arithmetic power consumed by each module of the system, it can be found that the convolutional layer consumes the highest arithmetic power. Therefore, this design focuses on the optimisation of the convolutional layer. The pseudo-code of the convolutional layer is shown in Figure 6, which shows that this part of the program consists of multiple layers of for loops nested together. The innermost layer of the loop needs to be reused several times, which causes a bottleneck in the system's response time when it is calculated as a serial operation on the PS side. If this part of the loop could be hardware accelerated, the overall speed of the convolutional neural network could be improved and the system latency reduced.

```
void conv1(float im[conv1_C][conv1_imX][conv1_imY],float rst[conv1_K][conv1_I][conv1_J]){
  int k,i,j,c,x,y;
  for (k=0;k<conv1_K;k++){
    for(c=0;c<conv1_C;c++){
      for(i=0;i<conv1_I;I++){
        for(j=0;j<conv1_J;j++){
          for(x=0;x<conv1_X;x++){
            for(y=0;y<conv1_Y;y++){
              rst[k][i][j]+= ker_1[k][c][x][y]*im[c][i+x][j+y];
            }
          }
        }
      }
    }
  }
```

Figure 6. Convolutional layer pseudo-code

After the above analysis, this design chooses to modularise the innermost layer of the convolutional layer, the circular convolutional computation part, into a function (mmult.c) for optimisation. The modified convolution layer pseudo-code is shown in Figure 7. ker_1[k] [c] denotes the convolution kernel, im[c] denotes the input image, and resullt denotes the output result after the convolution operation.

```
for (k=0;k<conv1_K;k++){
  for(c=0;c<conv1_C;c++){
    float result[60*60];
    mmult(ker_1[k][c],im[c],result);
    for(i=0;i<conv1_I;i++){
      for(j=0;j<conv1_J;j++){
        rst[k][i][j]+=result[i*60+j];
      }
    }
  }
}
```

Figure 7. Pseudo-code for convolutional layers after modularisation

When implementing convolutional neural network hardware acceleration based on FPGA, it is usually chosen to change the area for speed to optimize from the direction of increasing data throughput and reducing cycle delay time. Common methods include array partitioning, loop unrolling, loop pipeline, etc. According to the above strategy, this design uses the HLS optimization instruction to optimize the pseudo code of the mmult () function as shown in Figure 8.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

```
#include "mmult.h"
void mmult_kernel(float in_A[A_NROWS][A_NCOLS],
      float in_B[B_NROWS][B_NCOLS],
      float out_C[60*60]){
#pragma HLS INLINE self

#pragma HLS array_partition variable=in_A complete dim=2
#pragma HLS array_partition variable=in_B complete dim=2
   int i,j,x,y;
   for(i=0;i<60;i++){
      for(j=0;j<60;j++){
#pragma HLS PIPELINE
      float result =0;
      for(x=0;x<A_NROWS;x++){
#pragma HLS UNROLL
      for(y=0;y<A_NCOLS;y++){
      float product_term = in_A[x][y]*in_B[x+i][y+j];
            result += product_term;
            }
         }
         out_C[i*60+j]=result;
      }
   }
}
```

```
void mmult(float in_A[A_NROWS][A_NCOLS],
      float in_B[B_NROWS][B_NCOLS],
      float out_C[60*60]){
   int i,j;
   float a_buf[A_NROWS][A_NCOLS];
   float b_buf[B_NROWS][B_NCOLS];

   for(i=0;i<A_NROWS;i++){
      for(j=0;j<A_NCOLS;j++){
#pragma HLS PIPELINE
         a_buf[i][j]=in_A[i][j];
      }
   }

   for(i=0;i<B_NROWS;i++){
         for(j=0;j<B_NCOLS;j++){
#pragma HLS PIPELINE
            b_buf[i][j]=in_B[i][j];
         }
   }

   mmult_kernel(a_buf,b_buf,out_C);
}
```

Figure 8. Optimised pseudo-code for the mmult function

After adding the optimization instructions for the acceleration function in the convolutional layer, the project was recompiled and the performance of the acceleration function was estimated by the Estimated evaluation tool, and the results are shown in Figure 9.

Details

Performance estimates for 'mmult in conv1.c:13' function

HW accelerated (Estimated cycles)	24424

Details

Performance estimates for 'mmult2 in conv2.c:14' function

HW accelerated (Estimated cycles)	19229

Resource utilization estimates for HW functions

Resource	Used	Total	% Utilization
DSP	134	220	60.91
BRAM	60	140	42.86
LUT	37469	53200	70.43
FF	27698	106400	26.03

Figure 9. Hardware acceleration evaluation results

It can be seen that the most time-consuming part of the system consumes very little time after hardware acceleration, effectively improving the real-time performance of the system.

4. Test and result

In order to verify the recognition rate of the system for fire, this paper conducted 20 experiments on three test objects of flame, smoke and incandescent lamp respectively, and recorded the recognition results of the upper computer interface. The final statistical test results are shown in Table 1.

Table 1. Fire test results statistics table

Test results	fire	smoke	light
Number of fires identified	16	13	17
Recognized as normal times	4	7	3

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

In the actual test, it is found that when the flame and smoke are closer to the camera, there are more fire features in the image, which has a better recognition result, while the distance is too far and it cannot be correctly recognized. And during the smoke test, the denser the smoke, the better the system can identify the fire. However, when the background is similar to smoke, or when the smoke is thinner, the system will fail to report. In the process of testing the incandescent lamp, when the position of the incandescent lamp is close or in a dark area of the background, the system will cause a false alarm. Generally, the more complex the convolutional neural network model, the more image features can be extracted. However, this design takes into account the hardware resource limitations of ZYNQ-7020, only a relatively simple model is implemented, and some subtle features are not well extracted. As a result, when there are fewer fire features or more similar features in the image, the result is not very high. Good recognition effect. The physical debugging experiment is shown in Figure 10.

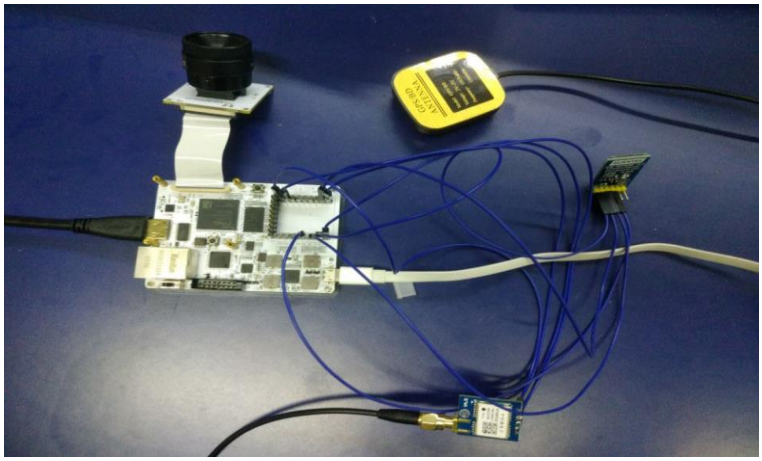

Figure 10. Physical picture of working status

5. Conclusion

This paper designs a fire early warning system based on ZYNQ for edge computing based on the current needs for fire prevention and control in old and remote places. This design uses edge computing technology to directly complete the fire identification and location functions at the monitoring terminal, avoiding the delay and distortion problems that may be caused by the long-distance transmission of video data in the traditional video fire warning scheme. In addition, the system results can be wirelessly transmitted through the Wi-Fi module, which has low requirements for basic network construction, and has a high degree of compatibility with the current scattered and old remote fields. It has the advantages of simple deployment, easy implementation, and convenient promotion. Finally, after testing, it is verified that the system can perform high-efficiency and high-quality work through wireless transmission. It has a high accuracy rate for fire identification, and the response time is only 431ms, which meets the accuracy and real-time requirements of the system, and better achieves the expected design goals. However, this system still has some shortcomings, and it needs to be further improved in future work.

References

[1] Lee Y, Kim H, Park E, et al. Optimization for object detector using deep residual network on embedded board [C]// 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), IEEE, 2016: 1-4.

[2] Yu Y, Zhao T, Wang M, et al. Uni-OPU: an FPGA-Based uniform accelerator for convolutional and transposed convolutional networks[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2020, 28(7):1545-1556.

[3] Sze V,Chen Y H,Emer J,et al.Hardware for machine learning: Challenges and opportunities[C]//2017 IEEE Custom Integrated Circuits Conference (CICC),IEEE,2017: 1-8.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0008

[4] Yan C, Gong N, Wen J. 2017. Real-time outdoor fire detection algorithm and its embedded application. Application of Electronic Technique, 43(4): 145-148.

[5] Meloni P, Deriu G, Conti F, et al. A high-efficiency runtime reconfigurable IP for CNN acceleration on a mid-range all-programmable SoC [C]// 2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig),IEEE, 2016: 1-8.

[6] Lecun Y, Bottou L. 1998. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[7] Gatys L A, Ecker A S, Bethge M. 2016. Image Style Transfer Using Convolutional Neural Networks[C]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2414-2423.

[8] Yu Zi-jian. FPGA-based accelerator for convolutional neural network [D]. Hangzhou: Zhejiang University, 2016.

[9] Chiuchisan I. 2013. A new FPGA-based real-time configurable system for medical image processing[C]. In 2013 E-Health and Bioengineering Conference (EHB). IEEE, 1-4.

[10] He K, Zhang X, Ren S, et al. 2016. Deep Residual Learning for Image Recognition[C]. IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 770-778.