FPGA Implementation of Neural Network Accelerator for PCB Defect Detection

Jinzhou Zhang[a], Hui Zhang*[b], Bingrui Zhao[c], Jiaxuan Liu [a], Xidong Zhou [a]
[a]College of Electrical and Information Engineering, Changsha University of Science & Technology, Changsha, China; [b] School of Robotics, Hunan University, Changsha, China; [c] Department of control science and engineering, Hunan University, Changsha, China
* Corresponding author: zhanghuiby@126.com

ABSTRACT

With the rapid development of artificial intelligence, deep neural network (DNN) has been widely used in industrial defect detection, intelligent driving, medical research, etc. However, DNN is still limited in the implementation of edge computing and mobile devices due to its characteristics of high model complexity and high computing resource consumption. Therefore, we designed a neural network hardware accelerator based on Field Programmable Gate Array (FPGA) for printed circuit board (PCB) defect detection. In this paper, firstly, since structure re-parameterization can improve the network's accuracy without increasing the inference model's complexity, we introduce structure re-parameterization to improve the YOLOv2 model and propose RepYOLOv2. Secondly, a low-bit quantization method based on integer type is adopted to quantify the model data to 6-bit. Then a specific convolutional computing module and neural network hardware accelerator are designed according to the characteristics of the model. Experimental results on Xilinx ZCU102 FPGA show that the real-time processing speed of the system reaches 2.12 FPS, the throughput is 68.53 GOP/s, and the power consumption is only 1.12 W. Compared with similar work, better performance is obtained.

Keywords: defect detection, PCB, quantization, hardware accelerator, FPGA

1. INTRODUCTION

A PCB is the core component of modern electronic products. PCB defects can cause connected electronic components to fail, degrade their performance, or even cause the entire production system to fail. Its quality inspection has become a pivotal link to meeting the long-term regular operation of electronic products. PCB surface defect detection mainly relies on artificial vision detection in the initial stage. However, artificial vision detection has disadvantages such as low detection efficiency, high labor cost and significant subjective error. In addition, PCB surface defects are minor, which makes it difficult for inspectors to observe and form effective and efficient detection.

In recent years, with the rise of machine vision and the breakthrough progress of deep learning technology, defect detection algorithm based on the deep neural network has been applied in various fields, especially industrial defect detection. For example, Wen et al.[1] aiming at the problem that it is difficult to detect the surface defects of semiconductor drawings by artificial eyes, they proposed a method for detecting the surface defects of semiconductor products based on deep convolution neural network. Du et al.[2] proposed using a deep learning method to detect the defects in the production process of silicon photovoltaic cells, given the problems of low defect detection efficiency, fewer detection data, and high false detection rate in the existing industrial production lines. However, the mainstream defect detection methods based on deep learning often rely on the graphics processing unit (GPU) mighty computing power, low detection efficiency, high cost and insufficient real-time. It is challenging to meet the flexible deployment requirements at the mobile and edge end.

This paper uses a neural network accelerator based on FPGA for PCB defect detection. Its main contributions are as follows:

- In order to enhance the feature extraction capability of YOLOv2, the original model was improved by introducing the idea of structure re-parameterization. In the inference stage of the improved RepYOLOv2 network, the model complexity did not increase, but the mean average accuracy (mAP) increased by 1.79%.

- For the sake of being able to deploy deep neural networks on low-cost FPGAs with limited resources, the low-bit quantization method based on integer was adopted. On this basis, the quantization performance of 16-bit, 8-bit and 6-bit is tested to reduce the number of model parameters further.
- On behalf of improving the computing efficiency of DNN, a neural network hardware accelerator based on various hardware optimization structures was designed for the low-bit quantization scheme and the improved RepYOLOv2. The inference speed of the whole system reached 2.12 FPS, and the dynamic power consumption was only 1.12 W.

2. SCHEME SELECTION

2.1 Algorithm selection

Before we can start designing a hardware accelerator for PCB defect detection, we must select a suitable target detection algorithm for the system. Target detection algorithms based on DNN can be divided into two types: single-stage and two-stage. The two-stage algorithm is based on selecting the target's possible region and then further classification and regression of the selected region. It has the characteristics of high detection accuracy and slow detection speed. Representative algorithms include R-FCN[3], FRCN[4], etc. The single-stage algorithm treats the object detection problem as regression and directly generates the object category probability and position coordinate value. The detection speed of a single-stage algorithm is faster than that of a two-stage algorithm, but the detection accuracy is lower. Representative algorithms include YOLOv2[5], YOLOv3[6], etc.

Table 1 compares several algorithms' speed and average accuracy (AP) on the COCO test-dev. The AP value indicates that the detection effect is better, and a smaller time value indicates that the detection speed is faster. In order to better complete the real-time detection of PCB defects, it is necessary to choose a fast detection algorithm. It can be seen from Table 1 that YOLOv2 and YOLOv3 are faster than the other two-stage algorithms with similar accuracy. In addition, given the limited hardware logic resources of low-end FPGA chips, the efficiency of implementing a single branch network such as YOLOv2 is generally higher than that of YOLOv3, which introduces a multi-branch network structure with the residual connection. Therefore, YOLOv3 AP is 11.4% higher than YOLOv2 AP, but YOLOv2 is faster. Besides, to achieve a better balance between high performance and easy implementation, we choose the YOLOv2 algorithm to design and apply the FPGA hardware accelerator.

Table 1. Performance comparison of different quantization bit widths.

Algorithm abbreviation	Test AP(%)	inference speed(ms)
R-FCN[3]	29.9	85
FRCN[4]	36.2	172
YOLOv2[5]	21.6	25
YOLOv3[6]	33.0	51

2.2 YOLOv2 algorithm model

YOLOv2 is improved based on YOLOv1 and adopts the darknet-19 backbone network, which includes nineteen convolutional layers and five max pooling layers. 3×3 convolution and 1×1 convolution are mainly adopted, in which 1×1 convolution can compress the number of channels of the feature graph to reduce the computational amount and parameters of the model. Batch Normalization (BN) layer is used after each convolution layer to accelerate model convergence and prevent over-fitting. In addition, the prior box, feature fusion and a variety of training techniques are used to make the model maintain a breakneck speed and significantly improve the problems of inaccurate positioning of YOLOv1 and poor detection ability of overlaps.

However, YOLOv2 does not introduce the multi-branch network structure of residual connection. Compared with the latest multi-branch algorithm, its precision performance is not superior. In order to obtain the most effective application in PCB-oriented defect detection, it needs to be further improved and optimized to improve the detection accuracy without increasing the model complexity.

Besides, the high parameter scale and high computational complexity of the YOLOv2 model still pose some challenges to FPGA implementation with limited hardware resources. In order to alleviate the memory-intensive and computation-intensive problems of the DNN model deployed at the edge, model compression technology should be adopted to further

optimize the improved YOLOv2 model. Model compression and optimization techniques are generally not targeted at specific hardware platforms. However, some methods can combine the flexible, reconfigurable characteristics of FPGA to achieve better compression effects, such as low-bit quantization.

Low-bit quantization can be further divided into floating point quantization, fixed-point quantization and integer quantization according to data types. Because of the high overhead of floating-point calculation on FPGA and the need for more clock cycles, fixed-point quantization and integer quantization are generally adopted. For example, Wang et al. [7] quantized the weight into 2-bit by the fixed-point quantization method. This method reduces the multiplications of AlexNet and ResNet-50 to one thousandth. The most representative of integer quantization is Binarized Neural Networks (BNN), proposed by Courbariaux et al. [8]. The weight and activation value are quantized to 1-bit, and the effectiveness of binary quantization of neural network is proved experimentally. However, the exponential decline in bit width inevitably leads to a considerable loss of information in the original data, resulting in a massive loss of accuracy on a more extensive data set. Therefore, we should not blindly pursue the reduction of model computation caused by extremely low-bit quantization. We should also consider the accuracy loss when choosing the optimal quantization strategy and quantization bit width according to the application scenario.

In summary, we introduce the idea of structural re-parameterization to improve the accuracy of the original network model and use the low-bit quantization scheme to compress the model size further. The specific method will be introduced in Section 3.

3. MODEL OPTIMIZATION

3.1 Proposed RepYOLOv2

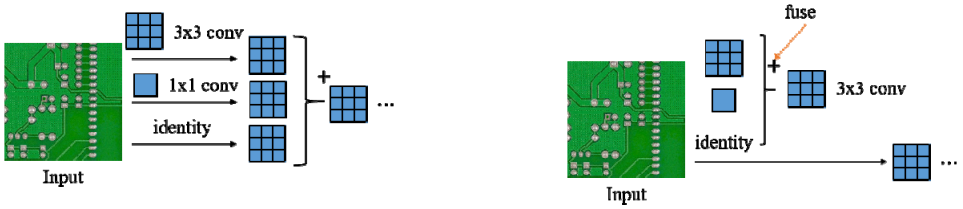
The model training uses the PCB defect dataset[9] from the Intelligent Robot Open Laboratory of Peking University, which includes 10668 defect images taken. The defect types were divided into six types: missing hole, mouse bite, open circuit, short, spur and spurious copper.

In the model training stage, we are considering that YOLOv2 is a straight bucket single-channel network, and the relatively new multi-branch design concept is not used, which makes it unable to benefit from the performance improvement brought by the multi-branch structure. Therefore, repYOLOv2, decoupled from training and reasoning, is proposed to improve the original model by introducing the idea of structure reparameterization in PCB defect detection.

Structural re-parameterization refers to the first construction of a series of structures (generally used for training) and the equivalent transformation of their parameters into another group of parameters (generally used for inference or deployment) to use parameter transformation to decouple training and inference structures[10].

Ding et al. [11] took the lead in using the idea of structural re-parameterization in the ACNet network to decouple training and reasoning. In the training stage, the conventional K×K convolution was split into K×K, 1×K and K×1 three-way convolution. In the reasoning stage, the additive property of convolution is used to replace the three-way convolution with K×K convolution through BN fusion and branch fusion. This method does not introduce any hyperparameters in the training process nor increases any parameters or calculation cost in the reasoning stage.

Based on the above ideas, as shown in Fig.1, we introduced the structure re-parameterization scheme in paper[12] to split 3×3 convolution in YOLOv2 model into 3×3, 1×1 and identity shortcut connection three-way convolution. that is, in order to improve the accuracy performance of the model. Multi-branch convolution component is added in the training stage, and branches in the FPGA deployment stage fuse the equivalent 3×3 convolution.



(a) Training-time (b) Deployed model

Figure 1. Structural re-parameterization of RepYOLOv2.

Experimental results show that, as shown in Fig.2, the detection accuracy of the improved RepYOLOv2 network model is improved compared with the original YOLOv2 model, and 1.79 % increases the mAP. In addition, the repYOLOv2 network model improves the detection accuracy of minor target defects and the model's reliability.

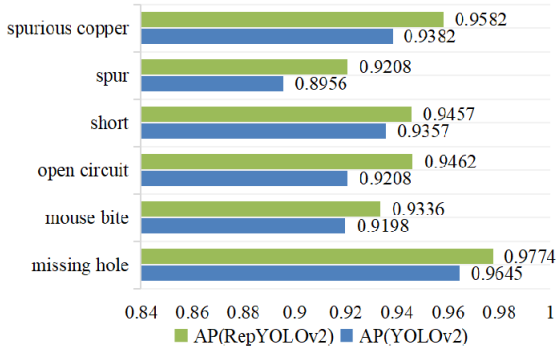


Figure 2. Comparison of detection before and after improvement.

3.2 Low bit quantization

As we all know, once the model is trained, the weight value of each layer will not change in the inference phase, and the weight value of each layer after training presents a Gaussian-like distribution. The weight distribution of each convolutional layer after training is quite different. In addition, a low-bit quantization method based on fixed point number is adopted in model optimization since the cost of floating-point number operation is much higher than fixed-point number operation in FPGA implementation.

The quantization scheme starts with a pre-trained CNN model. First, we collected each channel's weight distribution in the pre-training model and determined the search range of the optimal scale factor of each layer through (1) to calculate the clipping threshold $S_{\min}$ of each layer. Furthermore, using (2), the search completion point S is defined according to the maximum weight.

$$S_{\min} = \frac{2^n - 1}{2\delta} \tag{1}$$

$$S = \frac{2^n - 1}{\max(abs(x_{\max}), abs(x_{\min}))} \tag{2}$$

where the n represents the quantization bit width, the δ represents the standard deviation.

Then define the number of searching nodes as N. The scaling factors to be searched are evenly divided according to the search range and searching nodes.

In this study, N was set to 1000. Next, quantize the model with each searching node, and use MSE to measure the distance between the quantized channel and the original Non-quantized channel. Then find the scaling factor with the smallest MSE, as shown in (3) and (4):

$$S_{\min} S_{opt} = \arg\min_s err(s_k \mid w_k) \tag{3}$$

$$err(s_k \mid w_k) = \frac{1}{s_k} \times \left\| s_k \times w_k - clip(round(s_k \times w_k)) \right\|_2 \tag{4}$$

Where w_k is the k-th channel weights in the original float-point model, and s_k is the k-th channel weights after quantization by one of the scaling factors in searching nodes. Repeat the above steps in each model channel to get the optimal scaling factor table S and corresponding optimal clipping threshold table T as shown in (5).

$$T_k = \frac{2^b - 1}{s_k} \tag{5}$$

Finally, based on the clipping threshold of each channel, we fine-tune the network and impose a maximum weight constraint on each channel to decrease the clipping error. While the fine-tuned model converges, we use the optimal scaling factors table to re-quantize the model.

We performed quantization optimization on the trained RepYOLOv2 model based on the above quantization scheme. The optimization work mainly focused on several specific bit widths, including 16-bit, 8-bit and 6-bit, to match the best implementation of PCB detection application through different bit widths.

Table 2 compares the quantization results of different bit widths using the proposed method and those before quantization. Through comparison, it can be found that 16-bit quantization has the best effect, the accuracy rate reaches 94.58%, and the error loss is only about 0.11%. With the decrease of quantization bit width, the error loss increases, and the loss of 6-bit quantization reaches about 1.21%, but this result is still within the acceptable range of practical application. The quantization method can meet the accuracy requirements of PCB defect detection in this paper.

Table 2. Performance comparison of different quantization bit width

Method	Data-width(bit)	Accuracy(%)	Degradation(%)
original	FP32	94.69	-
quantization	INT16	94.58	0.11 ↓
	INT8	94.17	0.52 ↓
	INT6	93.48	1.21 ↓

4. FPGA ACCELERATOR IMPLEMENTATION

4.1 Overview of Accelerator Architecture

Fig.3 shows the overall architecture of the RepYOLOv2 accelerator for PCB Defect Detection. We use two bus interfaces, AXILite Slave and AXI Master. First, read and write the pixels and weights of DDR4 through the AXI Master and load them into the input buffer inside the accelerator. Secondly, because of the small amount of control signal data, AXILite Slave is used to transmitting the control signal. Then the processing module of the accelerator corresponding to the loaded data will be processed. Then the output result of the output buffer module will be transmitted to the external DDR4. The flow direction of volume data can be referred to in Fig.3 's arrow.

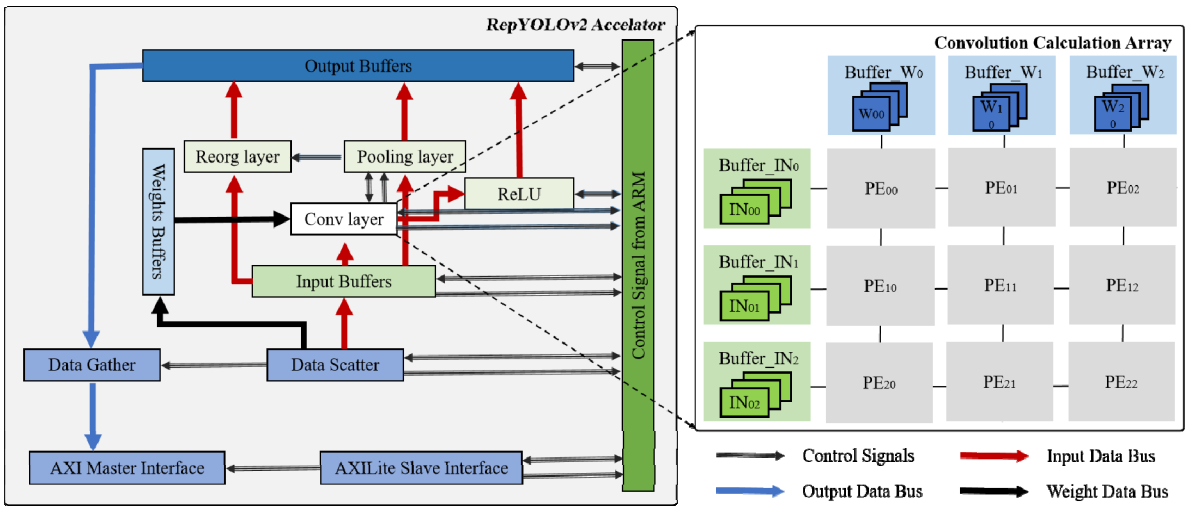


Figure 3. Overview of neural network hardware accelerator for PCB defect detect.

4.2 Convolution calculation Array

As mentioned above, RepYOLOv2 is composed of many convolutional layers, of which there are 23 in the whole network. Optimization of convolutional computation will bring significant performance improvement. Therefore, in order to save the resource consumption of the PCB defect detection system and improve the efficiency of convolutional computation, for the 6-bit quantization scheme, The traditional multiplication operation based on a convolution processing unit array is designed to replace conventional convolution, and its structure is shown in the right part of Fig.3.

As shown in Fig.3, based on cyclic block, the quantized 6-bit weight value and activation value are loaded into the processing unit (PE), each PE is responsible for the calculation of one output matrix element, and one call of PE can realize one multiplication accumulation calculation (MAC). In order to improve the parallelism, we design 9 PEs to form a convolution computing array for processing and map the row, the output Channel and the input channel in the convolution operation into three parallel dimensions row parallelism, column parallelism and PE parallelism, respectively.

In addition, the neural network hardware accelerator is designed by various optimization methods such as ping-pong cache and loop unrolling, and the inference speed of the whole system reaches 2.12 FPS.

5. EXPERIMENTAL RESULTS

5.1 Performance Test Settings

Xilinx ZCU102 is the development board used in the design. The specific model of FPGA is XCZU9EG-FFVB1156-2-I, which contains 274080 LUTs, 1824 BRAM_18K, 548160 FFs, and 2520 DSPs. In addition, the SoC also includes an ARM Cortex-A53 microprocessor, which is used for system control scheduling.

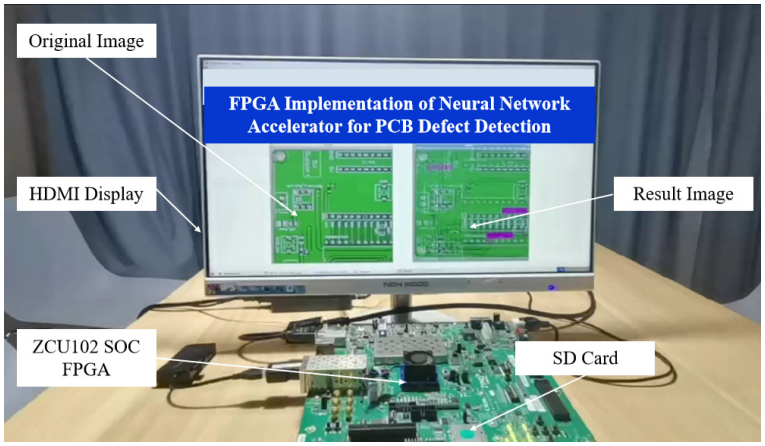


Figure 4. System physical photograph for PCB defect detection.

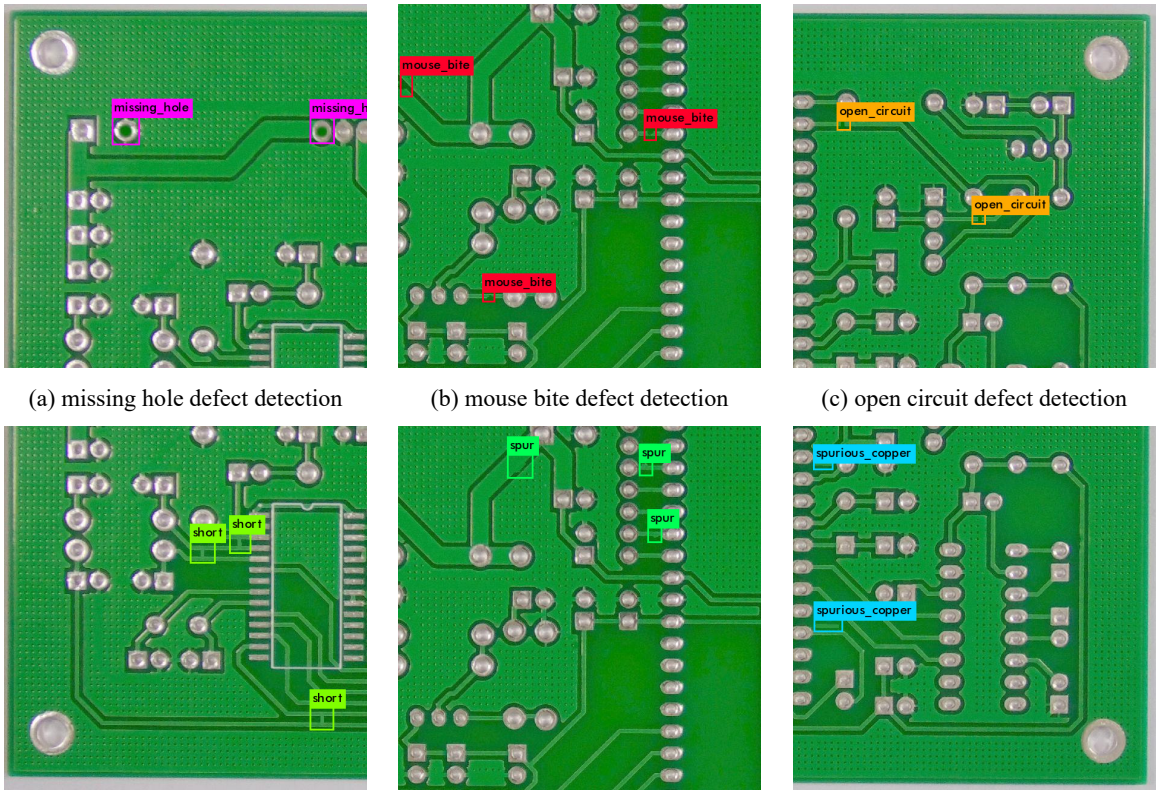
The purpose of this experiment is to test the actual performance of the PCB defect detection system, that is, the utilization of FPGA hardware resources and the runtime performance of the system. As shown in Fig.4, to better display the test results, we used the official Xilinx Vivado system integrated development environment and developed a desktop application with Petalinux and XSDK for the interaction between PS and PL at the system level. In advance, we stored the quantized parameters and test picture sets in an SD card, cache data using DDR4, and output the results to the HDMI display in real-time after the accelerator detected the tested pictures.

5.2 Performance test results and analysis

FPGA hardware resource consumption is related to the actual performance of the system. Therefore, we measure the resource utilization of the system. As shown in Table 3, the resource utilization of the whole hardware system is 20.8% LUT, 4% FFs, 9.8% BRAM and 6% DSP. Compared with previous work [13] and [14], higher throughput and energy efficiency are achieved, and dynamic power consumption is only 1.12W, which is very advantageous for low-cost FPGA hardware to implement deep neural networks. In addition, the detection effect is shown in Fig.5. It can be seen from Fig.5 that the neural network hardware accelerator in this paper can accurately realize PCB defect detection.

Table 3. Implementation hardware architecture performance comparison

FPGA Component	**Resource Utilization**		
	[13]	[14]	Ours
Device	Cyclone V	ZedBoard	ZCU102
Model	Tiny-YOLOv2	YOLOv2	RepYOLOv2
Image Size	416×416	416×416	416×416
Frequency(MHz)	117	150	250
Precision(bit)	16	16	6
LUTs	72320(64%)	37342(70%)	56927(20.8%
BRAM	-(40%)	88(63%)	178(9.8%)
DSP	122(41%)	153(70%)	152(6%)
FPS	-	1.02	2.12
Operations(GOPS)	21.96	34.90	29.47
Throughput (GOPS/s	21.60	35.60	68.53
Dynamic Power(W)	-	1.20	1.12
Power Efficient(GOPS/s/W)	-	29.67	61.20



(a) missing hole defect detection (b) mouse bite defect detection (c) open circuit defect detection

(d) short defect detection (e) spur defect detection (f) spurious copper defect detection

Figure 5. Effect of applying neural network hardware accelerator for PCB defect detection.

6. CONCLUSION

In this paper, to implement PCB defect detection algorithm on edge device, we designed a neural network hardware accelerator based on FPGA. Firstly, we introduced the idea of structural re-parameterization to optimize the network model. Based on this, RepYOLOv2 is proposed, and the mAP of this network is improved by about 1.79% compared with the original model. Secondly, a low-bit quantization method based on integer type is adopted, and a particular convolution array is designed for the 6-bit quantization scheme to improve the computational efficiency. Finally, we design a PCB defect detection system based on ARM-FPGA heterogeneous system. The system achieves a throughput of 68.53 GOPS, real-time inference speeds up to 2.12 FPS, and the power consumption is only 1.12 W.

REFERENCES

[1] Wen G, Gao Z, Cai Q, et al. A novel method based on deep convolutional neural networks for wafer semiconductor surface defect inspection[J]. IEEE Transactions on Instrumentation and Measurement, 2020, 69(12): 9668-9680.

[2] Du B, He Y, He Y, et al. Intelligent classification of silicon photovoltaic cell defects based on eddy current thermography and convolution neural network[J]. IEEE Transactions on Industrial Informatics, 2019, 16(10): 6242-6251.

[3] Dai J, Li Y, He K, et al. R-fcn: Object detection via region-based fully convolutional networks[J]. Advances in neural information processing systems, 2016, 29.

[4] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.

[5] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.

[6] Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[7] Wang P, Cheng J. Fixed-point factorized networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 4012-4020.

[8] Courbariaux M, Hubara I, Soudry D, et al. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1[J]. arXiv preprint arXiv:1602.02830, 2016.

[9] Ding R, Dai L, Li G, et al. TDD‐net: a tiny defect detection network for printed circuit boards[J]. CAAI Transactions on Intelligence Technology, 2019, 4(2): 110-116.

[10] Ding X, Xia C, Zhang X, et al. Repmlp: Re-parameterizing convolutions into fully-connected layers for image recognition[J]. arXiv preprint arXiv:2105.01883, 2021.

[11] Ding X, Guo Y, Ding G, et al. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 1911-1920.

[12] Ding X, Zhang X, Ma N, et al. Repvgg: Making vgg-style convnets great again[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 13733-13742.

[13] Yap J W, bin Mohd Yussof Z, bin Salim S I, et al. Fixed point implementation of tiny-yolo-v2 using opencl on fpga[J]. International Journal of Advanced Computer Science and Applications, 2018, 9(10).

[14] Liu C. YOLOv2 acceleration using embedded GPU and FPGAs: pros, cons, and a hybrid method[J]. Evolutionary Intelligence, 2021: 1-7.