Target Detection System Design and FPGA Implementation Based on YOLO v2 Algorithm

Fanghong Bi
Information Science and Engineering
Yunnan University
Yunnan, Kunming
e-mail: 2993785907@qq.com

Jun Yang
Information Science and Engineering
Yunnan University
Yunnan, Kunming
e-mail: Junyang@yun.edu.com

Abstract—**This paper proposes Target detection system design and FPGA implementation based on YOLO v2 algorithm, in order to realize offline real-time image detection in a platform with limited resources and power consumption. First, this paper studied the algorithm of YOLO v2 convolutional neural network, designed and trained the neural network. Secondly, a special floating-point number matrix multiplication unit and a double-cache data processing circuit are designed to improve the calculation speed and cell utilization efficiency of floating-point number matrix multiplication in convolutional neural network, and further accelerate the target detection speed on the hardware level. In this paper, the above scheme is implemented at the board level. The test results show that the average recognition speed is 50frame/s under the offline state, which basically achieves the design goal of real-time detection.**

Keywords-YOLO v2 algorithm; FPGA; Convolution Neural Network; object detection; Inference Accelerator

I. INTRODUCTION

In recent years, the popularity of artificial intelligence continues to rise, and the convolutional neural network model for target detection keeps emerging. In 2016, Redmon and his team proposed YOLO convolutional neural network model that can complete end-to-end training [1].With a certain accuracy, the speed (45frame/s) that can detect video is basically achieved [2].In the same year, an upgraded version of YOLO, YOLOv2, was proposed, which can maintain a high detection accuracy at a high speed and has a high advantage in real-time image processing [3].Compared with the R-CNN convolutional neural network algorithm that requires multiple CNN operations, although the accuracy is worse than the R-CNN algorithm, the YOLO algorithm is faster [4].

Most convolutional neural network model using the cloud server operations of the computing power of the powerful [5], but for specific such as underground parking, vehicle active safety configuration, etc. the user needs to terminal equipment with real-time, security, from the linear, small features such as energy consumption [6], therefore, in view of the specific application scenario for the convolutional neural network miniaturization offline dedicated hardware is at present academia and industry research hot spot. Traditional platforms (such as CPU and GPU) realize convolutional neural network reasoning with low energy efficiency and large volume [7], which is not applicable to the above terminal equipment. For these devices, researchers are turning to FPGA and ASIC. The neural network realized on ASIC achieves greater computing performance and lower power consumption, but it has high research and development cost, long design cycle, and cannot be reprogrammed to adjust the hardware structure, which increases the risk and cost of design [8].In recent years, field programma-ble gate array (FPGA) devices have been equipped with powerful computing performance and logic implementation ability. The neural network implemented on FPGA can be reprogrammed to adjust the hardware structure, which can better solve the shortcomings of ASIC-based neural network [9].

For the real-time target detection system, the inference prediction process consumes most of the computing time, hardware resources and power consumption [10]. Therefore, this paper focuses on the structural implementation of the dedicated inference accelerator based on YOLO v2 algorithm. In this paper, the YOLO v2 convolutional neural network is designed, the training model is used to obtain the weight parameter data, and Verilog HDL language is used to design the YOLO v2 convolutional neural network accelerator as the core of the real-time target detection system. Among them, a floating-point matrix multiplier with parallel architecture is designed to accelerate the detection algorithm at the hardware level. Secondly, the "double-BRAM" cache structure is proposed to maximize the use of computing modules. Finally, the FPGA of Xilinx was used to design and simulate the above mentioned YOLO v2 convolutional neural network inference accelerator. Simulation results show that the convolutional neural network reasoning accelerator in this paper has a peak speed of 3.188GMAC/s and a power consumption of 2.519W at 100MHz clock, which is 8.46 times faster than the general CPU. Compared with the traditional cloud server convolutional neural network target detection system, this paper has better processing speed, high system integration, less volume and energy consumption, strong reconfiguration, and convenient application in various scenarios, which has certain practical value and practical significance.

II. YOLO V2 CONVOLUTIONAL NEURAL NETWORK ALGORITHM

A. The Principle of YOLO v2

The YOLO algorithm was first proposed by Redmon in 2016. It is a brand-new end-to-end detection algorithm. In the same year, YOLO v2 was proposed. Among them,

978-1-7281-3663-9/19/$31.00 ©2019 IEEE

darknet-19 includes 19 convolutional layers and 5 maximum pooling layers [12]. The network USES a large number of 3*3 convolution kernels, and after each pooling operation, the number of channels is doubled. Put 1*1 convolution kernel between 3*3 convolution kernel to compress features and increase network depth. Batch normalization operation and dropout operation are added after each convolution layer. The YOLO v2 detection network takes darknet-19 as the basic model for feature extraction, and modifies its network structure accordingly. Remove the last convolutional layer of darknet-19 network, add three convolution layers of size 3*3 and channel number 1024, and add a convolution layer of size 1*1 after each convolutional layer, and the output dimension is the quantity required for detection.

Compared with the traditional convolutional neural network model, YOLO v2 only contains the convolutional layer and pooling layer, and the convolutional layer consists of convolution operation, batch standardization and activation operation [13]. After the convolution operation of the input image, the corresponding output feature graph is obtained, and then the input value distribution of neurons in each layer of the convolutional neural network is pulled back to the standard normal distribution by means of batch normalization to accelerate the convergence speed. Leaky ReLU function was adopted for the forward reasoning. By increasing the constant and multiplying the negative axis data, the data distribution was corrected so that all negative axis information would not be lost, thus solving the problem of "dead point" of the ReLU function [14]. The pooling layer reduces the parameters and computation on the premise of retaining the main features, so as to realize the dimensionality reduction operation of the input feature graph, prevent overfitting during training, and improve the generalization ability of the model.

B. The Model Structure of YOLO v2 Algorithm

The convolutional neural network architecture of YOLOv2 designed in this paper is shown in figure 1. There are a total of 22 layers of convolutional layer, 5 layers of maximum pooling layer, 2 layers of connection layer and 1 layer of recombination layer. The size of convolution kernel is 3*3 and 1*1, and the step size of pooling layer is 2*2.After pre-processing, the size of the input image is changed to 416*416, and the final output is 13*13 feature graph for inference and prediction. The value of k is 5, and the output result is a fixed tensor with dimensions of 7*7*30.

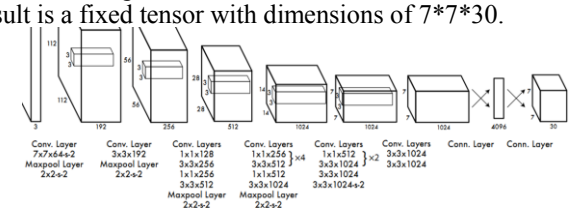

Figure 1. Convolutional neural network architecture diagram

III. YOLO ALGORITHM REASONING ACCELERATOR IMPLEMENTATION SCHEME

A. Overall Hardware Scheme

First, confirm that you have the correct template for your paper size. This template has been tailored for output on the US-letter paper size. If you are using A4-sized paper, please close this template and download the file for A4 paper format called "CPS_A4_format".

This design the overall structure diagram as shown in figure 2, hardware terminal based on the real-time target detection system, the input image data transferred by the real-time image acquisition unit to the target detection arithmetic unit of image preprocessing module, input characteristics of the processed figure is controlled by a control module, outside the piece of accelerator storage module and a reasoning and data calculation, data interaction between ultimate accelerator operation is completed through reasoning characteristic vector detection module is given by the sort of test results and send data receiving unit.

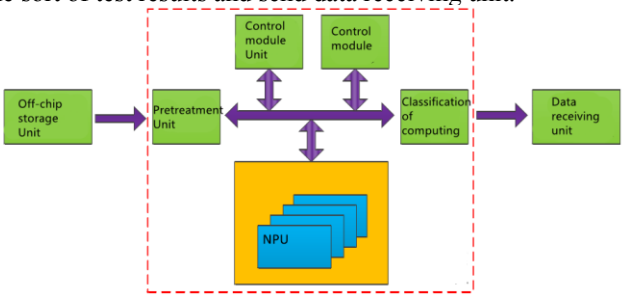

Figure 2. Schematic diagram of overall structure

The top design diagram of the reasoning accelerator based on floating-point Numbers is shown in figure 3. It is mainly composed of three modules, namely, the matrix multiplication unit, the distributed on-chip storage unit and the arithmetic logic unit, which are connected with each other through data lines.

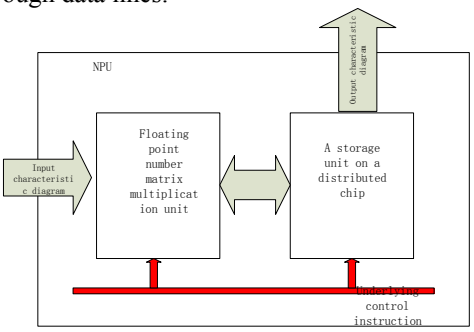

Figure 3. The design Top plan

B. Floating Point Matrix Multiplication Cell Design

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

The essence of convolution operation is matrix multiplication, which has a high computational complexity and its computational performance directly affects the

overall performance of the system [15].This paper involves the system to calculate high real-time demand, in order to improve the identification precision of the system, this paper designed floating-point unit matrix multiplication is input and output are taken up to the standard of the IEEE 754-1985 64 - bit double precision floating point number, the high computing performance requirements are put forward for the hardware system, floating point matrix multiplication becomes the bottleneck of improving overall system real-time performance. However, matrix multiplication requires a lot of multiplication and accumulation operations. In this paper, a parallel scheme between convolution Windows of the same input feature graph is selected, and a parallel structured double-precision floating point matrix multiplier is designed to improve the real-time performance.

The floating-point matrix multiplication unit designed in this paper contains P * P processing elements (PE) to reduce the calculation complexity. PE is the basic unit of the floating-point matrix multiplier. Each PE contains a floating-point multiplication and addition unit and a storage unit for storing the calculation results.

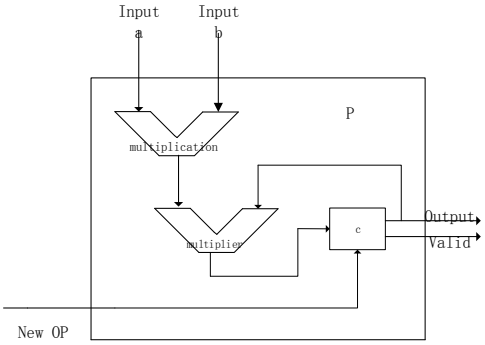

Figure 4. PE structure

C. Floating Point Matrix Multiplication Cell Design

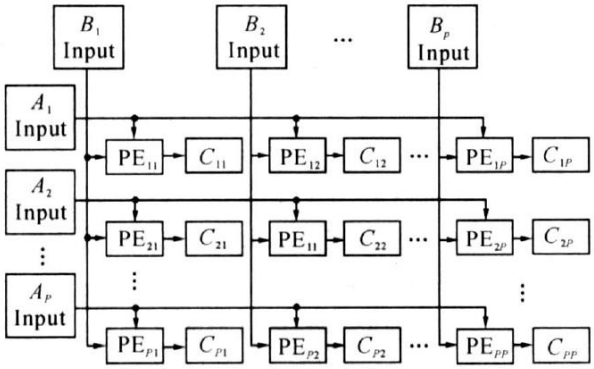

Figure 5. Structure of matrix multiplier

In order to improve the parallelism of system matrix multiplication, the above PE units are arranged into a P by P array as shown in figure 5. When performing matrix multiplication, all PE units in the same row share the input data of A matrix. All PE units in the same column share the input data of matrix B. In addition to containing a storage unit for intermediate results, each PE unit is configured with a storage unit for final results in the matrix multiplier. In this way, the main processor can read out the result of this multiplication and the matrix multiplier can carry out the next multiplication, which improves the efficiency of the multiplier.

D. Storage unit Design on Distributed Chip

The convolutional neural network itself has a large number of parameters, so on-chip storage in FPGA devices is difficult to store all parameters. The accelerator needs on-chip storage to realize on-chip caching of feature graph and coefficient data. Data cache operations usually include data loading and data processing. In order to avoid packet loss in out-of-chip data transmission, data processing operations can only be performed step by step after the RAM data loading operation on a single chip. When the cache is in the loading state, PE must wait, which will reduce the computational efficiency. To this end, this paper proposes a "double-bram" approach as shown in figure 5 to cache the weight parameter data and input characteristic graph data of the current layer. When the data of the weight parameter cache unit or the input feature image storage unit participates in the calculation, the DMA module loads the data to the other cache area. Data loading and data processing are completely parallel, which reduces the waiting time of PE unit when loading data. PE utilization is mentioned.

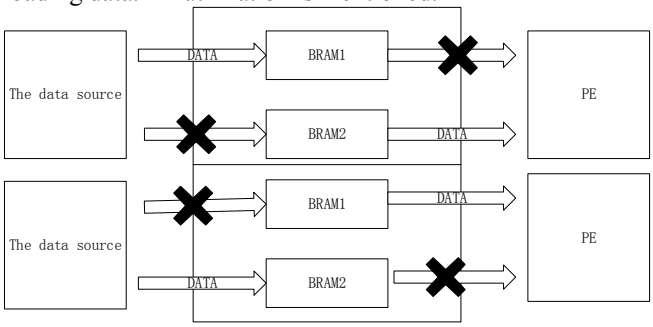

Figure 6. Data cache structure diagram

IV. Experimental Results

Before you begin to format your paper, first write and save the content as a separate text file. Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar:

In the experiment, the ZC702 development board provided by Xilinx was used. The development board contained 1GB of off-chip DRAM and was equipped with XC7Z020SOC. The chip contained an ARM processor and FPGA resource, which could simultaneously complete the software and hardware design of the architecture proposed in this paper. The hardware development environment adopted in the experiment was Vivado2014.4, and the software

development environment was Xilinx SDK 2014.4.The platform for YOLOv2 algorithm training and reasoning is PyCharm IDE launched by JetBrains company, which adjusts the structural parameters of the neural network. The target test set is the same as the hardware platform and is GTSRB. The obtained weight parameters are used for inference and prediction.

A. Recognition Accuracy

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as ASME, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

In this paper, the parameters quantized by data were substituted for the original parameters and tested on the software platform. It was found that the recognition accuracy of convolutional neural network decreased from 96.6% to 96.0%, that is, the accuracy introduced by data quantization decreased to 0.6%.After the board-level implementation, the recognition accuracy is still 96.0%, indicating that the hardware implementation does not introduce a new reduction in the recognition accuracy.

B. Comparison with General CPU Performance

In this paper, the performance of the floating-point number based inference accelerator is compared with that of the general CPU neural network. As shown in Table 1, in the same convolutional neural network model, the design can process 3.188g single-precision floating-point multiplication and accumulation operations per second, and the power consumption is 2.519w, and the acceleration ratio is 8.46 compared with the general CPU, but the power consumption is 3.88% of the general CPU. Through comparison, it can be known that inference accelerator has a huge computing advantage in processing neural networks with a large number of parallel computing, and it has a lower power consumption and is more suitable for terminal devices.

TABLE I. COMPARISON WITH GENERAL CPU

CPU			**This design(FPGA)**		
Intel Core i7-6700k			Xilinx XC7k325T		
GMAC/s	**Power consumption (W)**	**Frequency (MHz)**	**GMAC/s**	**Power consumption (W)**	**Frequency (MHz)**
0.377	65	4000	3.111	2.555	100

V. SUMMARY AND PROSPECT

A. Summary

The design has a recognition accuracy rate of 96.0%, a working frequency of 100MHz, a static power consumption of 0.184w, a dynamic power consumption of 2.335w, a single-precision floating-point multiplication and accumulation operation of 3.188g per second, a power consumption of 2.519w, and an acceleration ratio of 8.46 to the general CPU, but the power consumption is 3.88% of the general CPU.

Based on the high-speed parallel processing capability of FPGA chip, a dedicated hardware system for real-time target detection of YOLOv2 convolutional neural network algorithm is designed in this paper. Firstly, the YOLOv2 model is trained in the server segment design. Secondly, the training model is used to design the relevant circuit, and the high-speed floating point matrix multiplier and efficient calculation and caching circuit for YOLOv2 are designed. The system architecture and circuit implementation ensure the performance and efficiency of the design. Finally, Verilog HDL language is used to complete the design of the algorithm, and finally integrated into the IP core. In this paper, the target detection and recognition speed is faster, can be offline and real-time processing of large amount of data operation, through the actual test, achieved the expected design goal. The dedicated hardware system design of convolution neural network algorithm compared to the convolutional neural network system based on the cloud server has lower power consumption, higher level of integration, smaller equipment volume, and the system can be applied to real-time high offline scenarios, the cost is low, refactoring sex is strong, has a certain application value and practical significance.

B. Prospect

The prospect of since the embedded artificial intelligence of this terminal is still a relatively new field and there is no perfect theoretical guidance, there are still many areas for improvement in this paper. This paper can be further studied in the following aspects: first, although the inference accelerator is simulated and verified in this paper, the decoding part from the top instruction set to the bottom control instruction, the image preprocessing module, the classification calculation module and so on have not been completed, which needs to be further implemented on the FPGA board after improvement. Secondly, the binary YOLO algorithm model is not designed in this paper, so that it can replace the original intermediate floating point number calculation with the fixed point number of 16, 8 or even fewer digits, further reducing the use of hardware resources and power consumption of reasoning accelerator.

REFERENCES

[1] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J]. 2018

[2] CHANG, F. H, BROADBENT, et al. INFLUENCE OF TRACE METALS ON CARBON DIOXIDE EVOLUTION FROM A YOLO SOIL.[J]. Soil Science, 1981, 132(6):416-421.

[3] Wang L, Yang S, Yang S, et al. Automatic thyroid nodule recognition and diagnosis in ultrasound imaging with the YOLOv2 neural network[J]. World Journal of Surgical Oncology, 2019, 17(1):12.

[4] Wang H, Zhang Z. A vehicle real-time detection algorithm based on YOLOv2 framework[C]// Real-time Image & Video Processing. 2018.

[5] Li H, Zhe L, Shen X, et al. A convolutional neural network cascade for face detection[C]// Computer Vision & Pattern Recognition. 2015.

[6] Shafiq F, Yamada T, Vilchez A T, et al. Automated flow for compressing convolution neural networks for efficient edge-computation with FPGA[J]. 2017.

[7] Qiu J, Wang J, Yao S, et al. Going Deeper with Embedded FPGA Platform for Convolutional Neural Network[C]// Acm/sigda International Symposium on Field-programmable Gate Arrays. 2016.

[8] Zhang J, Li J. Improving the Performance of OpenCL-based FPGA Accelerator for Convolutional Neural Network[C]// Acm/sigda International Symposium on Field-programmable Gate Arrays. 2017.

[9] Zhang J, Li J. Improving the Performance of OpenCL-based FPGA Accelerator for Convolutional Neural Network[C]// Acm/sigda International Symposium on Field-programmable Gate Arrays. 2017.

[10] Qiao Y, Shen J, Huang D, et al. Optimizing OpenCL Implementation of Deep Convolutional Neural Network on FPGA[J]. 2018.

[11] Yang W, Zhang J, Wang H, et al. A vehicle real-time detection algorithm based on YOLOv2 framework[C]// Real-time Image & Video Processing. 2018.

[12] Jianwei L, Kai G. [IEEE 2018 37th Chinese Control Conference (CCC) - Wuhan, China (2018.7.25-2018.7.27)] 2018 37th Chinese Control Conference (CCC) - Research on Monocular Visual Gesture Positioning based on YOLO v2[C]// 2018:9101-9106.

[13] Jo K U, Im J H, Kim J, et al. [IEEE 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA) - Kuching (2017.9.12-2017.9.14)] 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA) - A real-time multi-class multi-object tracker using YOLOv2[J]. 2017:507-511.

[14] Bachand P , Bachand S M , Fleck J , et al. Transpiration Driven Hydrologic Transport in vegetated shallow water environments: Implications on Diel and Seasonal Soil Biogeochemical Processes and System Management[C]// AGU Fall Meeting Abstracts, 2011.

[15] Yonekawa H, Nakahara H. On-Chip Memory Based Binarized Convolutional Deep Neural Network Applying Batch Normalization Free Technique on an FPGA[C]// 2017.