Hardware Acceleration of Face Detection Using a Deep Convolutional Neural Network – a demo

Dominika Przewłocka
AGH University of Science
and Technology Krakow, Poland
E-mail: dprze@agh.edu.pl

Tomasz Kryjak, *Member IEEE*
AGH University of Science
and Technology Krakow, Poland
E-mail: tomasz.kryjak@agh.edu.pl

***Abstract*—Face detection is one of the most popular algorithms used in vision systems, especially for access control, video surveillance and image tagging in social media. In order to achieve high accuracy, a two-stage detector was proposed. Firstly, a colour based skin region segmentation followed by connected component labelling and object parameter analysis was applied. Then a Deep Convolutional Neural Network (DCNN) was used to classify pre-selected areas as face or non-face. A DCNN has a high computational and memory complexity, which significantly affects the frame processing time. Therefore in this case hardware acceleration of the detector is essential. The algorithm has been implemented on the Xilinx ZCU102 heterogeneous platform using a hardware-software approach. The system is able to work with video stream up to 4K.**

***Index Terms*—DCNN, face detection, Zynq UltraScale, FPGA, 4K**

I. Introduction

Face detection is an important issue in the context of vision systems. However it is also quite challenging due to several factors: different face appearance (rotations, hair cut, glasses), different sizes and uneven lightning, Therefore many approaches have been proposed by the computer vision community. A large group of existing solutions are algorithms based on skin colour segmentation combined with geometric facial filters – [1], [2]. Other are algorithms based on object features – e.g. HOG (Histogram of Oriented Gradients) [3] or Haar features [4]. Deep convolutional neural networks are another, relatively new, group of algorithms used for face detection. DCNN are quite resistant to various interferences. However, these algorithms have high demand for both computing power and memory, which can be a drawback in case of systems with real-time (or close to real-time) operating requirements and low energy budget (so-call edge devices). In this work, the implementation of a two-step face detection (including pre-processing and classification) on a heterogeneous Zynq UltraScale+ hardware platform is proposed.

II. Related work

In recent years, the issue of hardware acceleration of deep convolutional neural networks with the use of FPGA platforms has gained popularity. The algorithm proposed in the work [5] is based on the possibility of parallelizing calculations in different directions in convolutional layers, loop optimization and special memory organization with the use of transitional buffers. In the paper [6] authors proposed the use of Winograd transform for operations in convolutional layers, which led to their simplification and, as a result, resource saving. Another solution are networks with reduced precision of calculations, such as, for example, binarized convolutional neural networks [7].

The work presented in this paper was supported by the National Science Centre project no. 2016/23/D/ST6/01389

III. The used algorithms

Image pre-processing consists of two stages: segmentation and connected component labelling. To designate skin colour areas, the image is converted from RGB to YCbCr and HSV, and then the image is binarized using thresholds in all three spaces. The resulting binary map is subjected to median filtration and morphological operations to remove minor disturbances. The last step is labelling, which provides the coordinates of the bounding box of objects with skin colour. Each candidate is checked for size - recognitions with too small area are rejected, then the rest is classified.

A deep convolutional neural network is used as a classifier. The input is a selected and scaled area from the input image. The output of network is a two-element vector of probabilities each representing the face or non-face class. The implemented network was trained by fine-tuning the AlexNet network.

IV. Hardware-software implementation

The scheme of the proposed system is presented in Figure 1. The video stream from the camera is fed to the programmable logic (PL) of the Zynq device, where the pre-processing takes place. The areas determined by labelling are transferred to the processing system (PS), from where they are respectively sent to input of the DCNN implemented in PL. The results obtained after the last convolutional layer are transferred back to the PS in order to calculate fully connected layers and the network output. In the case of positive classification, face coordinates are sent to the visualization module.

A. Hardware implementation of DCNN

The DCNN module consists of the following elements: BRAM memory bank for kernels used in a given layer (or layer group), two BRAM memory banks for intermediate results of the convolutional and pooling layers, module of

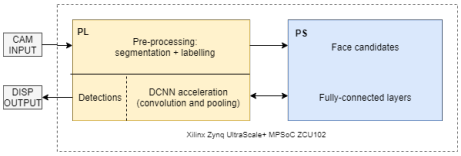

Fig. 1. The proposed hardware-software system

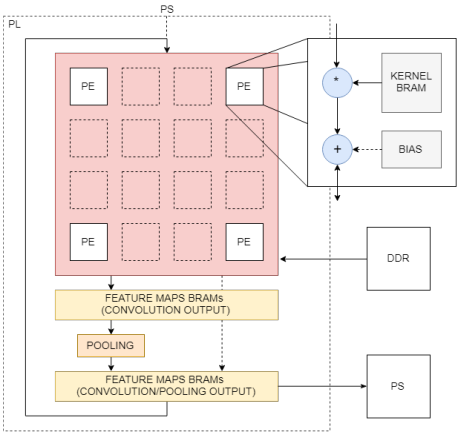

Fig. 2. Hardware architecture of the DCNN

Processing Elements Matrix (PE) performing multiplication and accumulation operations, a module for reading data from DDR memory, a module for writing and reading data from the processing system, and a state machine module managing the resources and controlling the algorithm (cf. Figure 2).

B. Memory organization

Three different types of data storage are used: internal registers, BRAM memories and external DDR memory. The kernels of all layers of the DCNN are stored in the DDR memory. Prior performing calculations in a given layer, the weights are read and buffered into the BRAM bank, from which each block is corresponding to one filter (depending on the number of filters, not all blocks are active). Filter weights are stored as fixed point, 9-bit numbers (sign, 1 bit integer, 7 bits fractional). Intermediate results of successive layers (convolutional and pooling) are written in one of two other available BRAM banks as 18-bit fixed-point numbers (sign, 10 integer, 7 fractional). Each BRAM corresponds to one output feature map. The bias of each layer is stored on-board in registers.

C. Memory and resources limitations

Beside precision reduction from floating to fixed point and the use of external DDR memory, two additional solutions were used to reduce memory and resources consumption. First, filter grouping - in the case when volume of filters exceeds the BRAM capacity in the kernels' bank, the calculations are divided into two groups, dividing the input and filter channels in half and performing convolution respectively on first and second group. Secondly, fully connected layer calculations are moved to the processing system.

TABLE I
THE USE OF FPGA RESOURCES.

LUT	FF	BRAM	DSP
51450 (18.77 %)	65622 (11.97 %)	777.5 (85.25 %)	384 (15.36 %)

D. Organization of calculations

The Processing Elements Matrix consists of PE modules. Each performs a MACC (Multiply ACCumulate) operation for a specific filter in a layer using DSP block. The input data is sent in series, the element receives subsequent pixels from a context with relevant weights – after calculating the weighted sum, the data is sent in parallel (from all PE) to the BRAM bank with intermediate results. The pooling layer is implemented in a simple way – in parallel for all the output channels of the previous layer, the context is read and the sub-sampling operation is performed. The results are saved to the second (free) BRAM bank. After performing calculations in all of the convolutional and pooling layers, the feature maps are sent to the processing system, where the DCNN classification vector is calculated.

V. CONCLUSION

In the paper a hardware-software implementation of a DCNN based face detection system was presented. The biggest challenge was the selection of an appropriate computing architecture, which would allow to meet the memory and resources limitations of the hardware platform. The used resources are presented in Table I. Due to the fact that the DCNN module is a part of a larger system and its hardware implementation could not use all available resources, external DDR memory and transfer of some calculations to the processing system was used. The proposed solution fully exploits the capabilities of the heterogeneous Zynq UltraScale device as allows real-time processing of up to 4K video stream.

REFERENCES

[1] Y. Yang, C. Xie, L. Du and Q. Lu, "A new face detection algorithm based on skin color segmentation," 2015 Chinese Automation Congress (CAC), Wuhan, 2015, pp. 523-526.
[2] S. Yadav and N. Nain, "Fast Face Detection Based on Skin Segmentation and Facial Features," 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Bangkok, 2015, pp. 663-668.
[3] H. c. Yang and X. A. Wang, "Cascade Face Detection Based on Histograms of Oriented Gradients and Support Vector Machine," 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Krakow, 2015, pp. 766-770.
[4] S. Wang, G. Wen and H. Cai, "Research on face detection based on fast Haar feature," 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Shanghai, 2017, pp. 1-6.
[5] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao and J. Cong, "Optimizing fpga-based accelerator design for deep convolutional neural networks," FPGA, 2015.
[6] U. Aydonat, S. O'Connell, D. Capalija, A. C. Ling, G. R. Chiu, "An OpenCL(TM) Deep Learning Accelerator on Arria 10," FPGA 2017.
[7] H. Yonekawa and H. Nakahara, "A batch normalization free binarized convolutional deep neural network on an fpga," 2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Lake Buena Vista, FL, 2017, pp. 98-105.