DT-CNN: Dilated and Transposed Convolution Neural Network Accelerator for Real-time Image Segmentation on Mobile Devices

Dongseok Im, Donghyeon Han, Sungpill Choi, Sanghoon Kang, and Hoi-Jun Yoo
School of Electrical Engineering
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea
dsim@kaist.ac.kr

***Abstract*—A convolution neural network (CNN) accelerator is proposed for real-time image segmentation on mobile devices. The proposed CNN processor cuts down the redundant zero computations in dilated and transposed convolution for higher throughput. As a result, the overall computations of the image segmentation are reduced by 86.6% and the proposed CNN processor boosts up the throughput 6.7×. Moreover, the proposed processor utilizes RoI (Region of Interest) based image segmentation algorithm to reduce the overall computational requirement significantly. Although RoI based image segmentation degrades the image segmentation accuracy, the proposed dilation rate adjustment compensates for the accuracy degradation and maintains the accuracy of the full-size image segmentation. Finally, the proposed CNN processor is simulated in 65 nm CMOS technology, and it occupies 6.8 mm^2. The proposed processor consumes 196 mW and shows 211 frames-per-second (fps) at the image segmentation for human body parts.**

Keywords—Deep neural network, convolution neural network, dilated convolution, transposed convolution, image segmentation, action recognition, and human body segmentation

I. INTRODUCTION

Recently, image segmentation is widely utilized for various vision tasks such as action recognition [1]. Semantic information of the body (head, torso, arms, legs, etc.) enables accurate action recognition by understanding the specific parts of the body in action. Coinciding with the dramatic improvement of vision algorithms based on deep learning, convolution neural network (CNN) is widely used for the image segmentation.

CNN based image segmentation requires much more computations compared with the CNN based image classification. Whereas CNN for classification gradually shrinks the feature size through pooling layers, CNN for segmentation maintains the feature size in order to preserve positional information. For instance, one of the light-weight image segmentation network, ENet [4], requires >16.4× more computations than the image classification network, MobileNets [2]. Therefore, real-time image segmentation is difficult to be implemented on conventional CNN inference accelerators [3, 6, 7] which focus on the image classification task. Although there exists an FPGA implementation of the image segmentation [8], it shows approximately 1W power consumption which is not applicable to mobile devices with limited battery capacity.

Fig. 1 illustrates the overall architecture of the image segmentation CNN [4]. The segmentation CNN consists of 3 different types of convolution layer, 2D convolutions, dilated convolutions (D-CONV), and transposed convolutions (T-CONV). D-CONV performs 2D convolution with the enlarged kernels created by inserting virtual zeros between the kernel elements. The number of inserted zeros is denoted as dilation rate, which represents the distance from the adjacent kernel elements due to the virtual zeros. Symmetrically, T-CONV is a 2D convolution with the enlarged input activation maps constructed by inserting zeros between the adjacent input neurons. Both D-CONV and T-CONV contain a large number of zeros so that more than 90% of the overall computations can be considered redundant because the zeros do not affect the final convolution result.

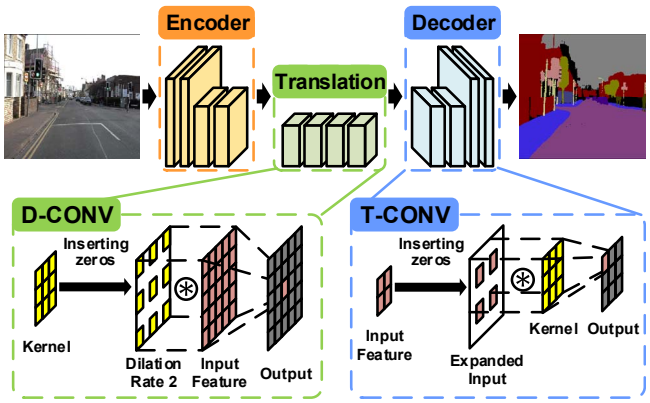


Fig. 1. The Overall Network Architecture for Image Segmentation and the Detail Operations of D-CONV and T-CONV.

Since both D-CONV and T-CONV can be replaced by the normal 2D convolution by inserting the virtual zeros in either kernel map or input feature map, previous CNN accelerators [3, 6, 7] can compute both D-CONV and T-CONV by feeding the kernels and input feature map with the virtual zeros. However, the redundant zero multiplications of D-CONV and T-CONV cause the processing elements (PEs) to be occupied with the redundant multiply-and-accumulate (MAC) operations. Finally, the effective PEs utilization decreases dramatically down to <10%, causing throughput degradation and power efficiency decrease.

For the purpose of the throughput enhancement, the region of interest (RoI) can be selected through light-weight algorithms. Then, the CNN operations are required only for the selected RoI, and a large portion of the computations can be eluded. For example, selecting 128×192 sized RoI out of 288×288 sized input image can reduce 70.3% of the CNN operations. However, since the segmentation network is trained with full-size images, severe accuracy degradation occurs as the RoI size decreases, showing 16.4 percentage point degradation with 128×192 RoI for human body parts

978-1-7281-0397-6/19/$31.00 ©2019 IEEE

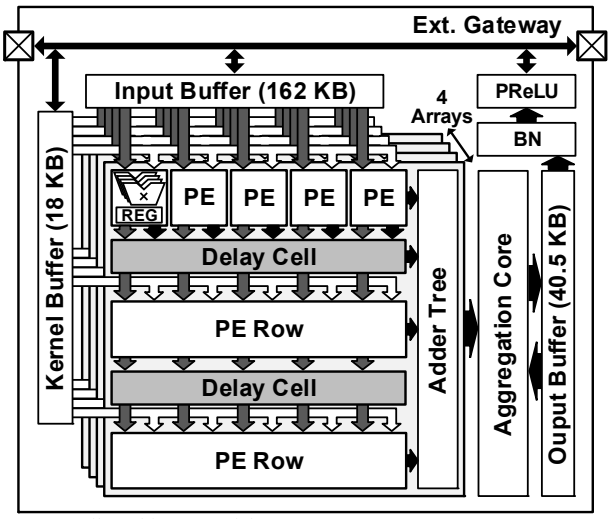


Fig. 2. Overall Architecture of the DT-CNN Processor.

segmentation. Therefore, a compensation method is required to restore the accuracy and exploit throughput enhancement with RoI based segmentation.

In this paper, the CNN accelerator (DT-CNN) for real-time image segmentation is proposed with the two key features: 1) the delay cell to reduce 86.6% of the overall MAC operations by skipping redundant zero operations in D-CONV and T-CONV, 2) dynamic dilation rate adjustment to reduce additional 29.6% computations without any accuracy degradation caused by RoI based segmentation. As a result, the proposed real-time image segmentation processor is implemented in 65nm CMOS process, achieving 211 frames-per-second (fps) segmentation throughput with only 196 mW power consumption.

II. Proposed Image Segmentation Processor

A. Overall Architecture

Fig. 2 describes the overall architecture of the proposed DT-CNN processor. It consists of the 4 PE arrays, an aggregation core, a PReLU unit, and a batch normalization (BN) unit.

Each PE array computes with different input channels of the input feature maps, then its outputs are summed up at the aggregation core. The PE array consists of the 3×5 PEs, where 5 different input features in the same channel are fed to the 5 columns of the array, and propagates down across the rows in systolic manner. The kernels are broadcasted in each row, for convolutional data reusability. Inside the PE, the 4 different kernels corresponding to different output channels are simultaneously multiplied with the same input feature to obtain the 4 different output features after convolution.

The outputs of the PEs are the partial sums during the CNN operation, and the partial sums in each row are summed up by the adder tree. The results of the adder tree are gathered in the aggregation core. The aggregation core collects the partial sums from different PE arrays, and then finally acquires the output features from 4 different channels. The final convolution outputs go through the PReLU and batch normalization unit, and they are stored in the external memory.

The on-chip memories consist of the input buffer (162 KB), the kernel buffer (18 KB), and the output buffer (40.5 KB). The kernel buffer stores all weight kernels required for a single layer, which is double buffered to fetch what's need for the next layer without any latency. The output buffer plays a role of temporal storage of the partial sums until a final convolution results are acquired.

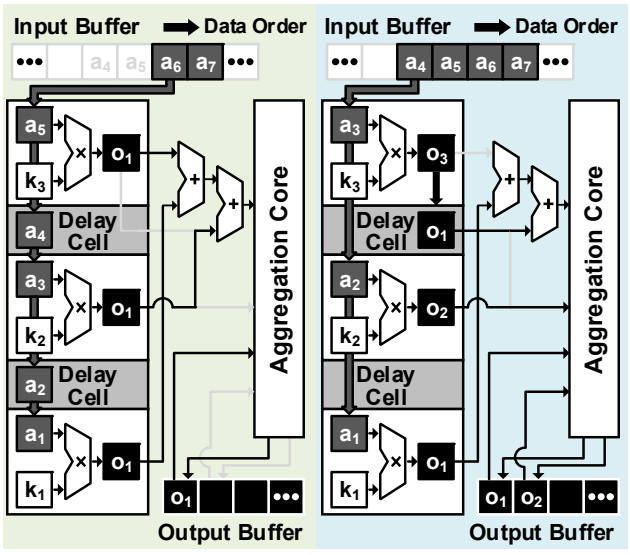
B. Dense CNN Operation with D-CONV and T-CONV

As denoted in the Section I, D-CONV inserts the virtual zeros between the kernel elements and these virtual zeros causes redundant computations. The DT-CNN processor needs to exclude these virtual zeros during the calculations to avoid the redundant computations. However, skipping the virtual zeros is challenging on the convolution data path, because it requires complex routing of the partial sums during the aggregation.

To solve the problem with small hardware overhead, the PE array with delay cells is proposed. The insertion of simple delay cells, which are based on registers, can solve the problem with the minor hardware costs. The elements of the input feature maps are latched temporarily in the delay cell before propagating to the next row of the PE array, so that the partial sums which should be accumulated can be generated at the same cycle.

Fig. 3(a) depicts the detailed data path for D-CONV. At the first step of D-CONV, an input feature element a_1 is fetched from the input buffer and fed into the first row of the PE array. Then, a_1 is multiplied with the kernel element, k_3. As the next input feature a_2 is fed at the next cycle, the PE pushes the a_1 into the delay cell, in column-wise direction. If the dilation rate is n, the a_1 stays at the delay cell for $(n-1)$ cycles before propagating to the next PE row. Fig. 3(a) shows the propagation scenario when the dilation rate is 2. The input feature a_1 rests in the delay cell for a single cycle, before being passed into the second row of the PE array. As a_1 propagates all the way down to the last row, the outputs from each PE in different rows correspond to a same partial sum, which will be accumulated by the adder tree to generate the first partial sum, o_1. In the aggregation core, the partial sums from different arrays and the pre-calculated partial sum from the output buffer are summed up to obtain the final convolution output.

The proposed data path can also be applied to the



(a) (b)

Fig. 3. The Data Path of (a) D-CONV and (b) T-CONV

conventional 2D convolution without any dilation, because it can be considered as D-CONV with the dilation rate of 1. Therefore, the input elements do not rest in the delay cell but directly passed onto the PEs in the next row.

On the other hand, T-CONV performs 2D convolution with input feature maps which are expanded by the regular zero patterns. The input features do not rest in delay cells (dilation rate of 1) because there is no virtual zeros in the kernels in the case of T-CONV. However, multiplying the kernels with the input features without the virtual zeros generates different partial sums each row, which will not be accumulated each other. In other words, the partial sums, which should be accumulated, are generated at different clock cycles from different rows. The proposed delay cell can also resolve the issue by latching the multiplication results, not the input feature. Temporarily storing the multiplication results in the delay cells helps synchronize the T-CONV process between different rows of the PE array with minimal hardware complexity.

Fig. 3(b) illustrates the operation of 3×3 T-CONV. The input elements for T-CONV are fetched and passed into the next row without resting at the delay cells. The output of the first row is o_3, and it is latched by the delay cell. The previous latched partial sum in the first row's delay cell is o_1, which was the multiplication result between k_3 and a_2, is added with the current output of the third row.

The proposed DT-CNN processor prevents the dramatic utilization drop of D-CONV and T-CONV with the proposed delay cell architecture. As a result, a D-CONV layer with dilation rate of 16 and a T-CONV layer improves the throughput 109× and 3.84× higher respectively. Therefore, register based simple delay cell alleviates the need for complex routing, enabling the DT-CNN processor to accelerate the high throughput.

C. *Variable Size of RoI based Image Segmentation with Configurable Dilation Rates*

The RoI based image segmentation can greatly reduce the required computations, and enhances the performance of DT-CNN processor. Cropping a 288×288 image into a 128×192 RoI can reduce the overall CNN operations by 70.3%. However, the RoI based image segmentation degrades the image segmentation accuracy by up to 16.4 percentage points because the pre-trained dilation rates are optimized for the original 288×288 images. As shown in Fig. 4(a), D-CONV with big dilation rate causes overflows in receptive fields if the target RoI is small. To resolve this problem, dynamic dilation rate adjustment is proposed to regulate the dilation rates dynamically depending on the RoI sizes. By adjusting the dilation rate to change the kernel to fit the RoI size, the overflow problem is resolved, showing much higher accuracy compared with RoI based segmentation with static dilation rate.

Fig. 4(b) shows the mIoU (mean Intersection over Union) of the image segmentation CNN over different RoI sizes. With the baseline indicating mIoU without RoI cropping, the mIoU decreases as the size of RoI gets smaller in static dilation rate. However, even with small sized RoIs, the proposed dynamic dilation rate adjustment compensates for the accuracy loss up to 15.8 percentage points. As a result, the throughput of the target segmentation CNN is increased by 3.08× without any accuracy drop, thanks to the RoI based

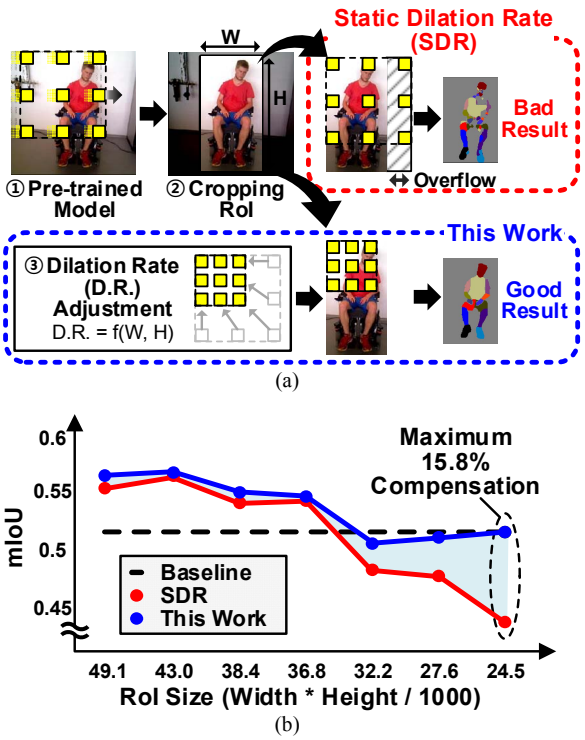


Fig. 4. (a) RoI based Image Segmentation with the Proposed Dilation Rate Adjustment, and (b) the Relation between mIoU and RoI sizes

segmentation with the proposed dynamic dilation rate adjustment. Furthermore, dynamic dilation rate is fully supported in the proposed DT-CNN processor. The dilation rate can be adjusted in the convolution data path by simply changing the number of cycles the input feature rests in the delay cell between the PEs.

III. Implementation Result

The layout photograph of the proposed DT-CNN processor is shown in Fig. 5. The processor is implemented in 65nm CMOS technology, and it occupies 6.8 mm^2. The processor operates in 200 MHz clock frequency with 1.2 V supply voltage, consuming 196 mW of power.

Table I shows that the throughput (8b) of the proposed DT-CNN processor is 96 GOPS with computing all the values including the virtual zeros. However, the proposed DT-CNN processor can perform at much higher throughput during the image segmentation, because it can skip the virtual zeros in D-CONV and T-CONV. Considering the factor, the DT-CNN processor shows 639.7 GOPS, which is higher than DNPU [3]. Moreover, the area efficiency and power efficiency of the processor are 2.18× and 1.40×

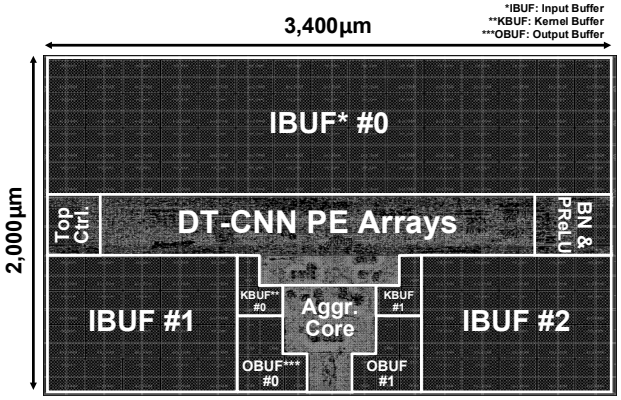


Fig. 5. The Layout Photograph

TABLE I. PERFORMANCE COMPARISON TABLE

	ISSCC2017 [3]	ISSCC2018 [6]	This Work
Technology	65 nm	65 nm	65 nm
Operating Condition	1.1 V, 200 MHz	1.1 V, 200 MHz	1.2 V, 200MHz
Throughput [GOPS]	600 (8b)	691.2 (8b)	96 (8b) / 639.7 (8b)*
Area [mm^2]	16	16	6.8
Power [mW]	279	297	196
Area Efficiency [GOPS/mm^2]	37.5	43.2	94.1*
Power Efficiency [TOPS/W]	2.15	2.33	3.26*
On-chip SRAM [KB]	256	256	220.5

**Based on Logical Throughput on ENet [4]*

higher than the state-of-the-art CNN accelerator.

Fig. 6 shows the summary of the performance enhancement by the 2 key features of the DT-CNN processor. As shown in Fig. 6(a), the previous image segmentation network [4] requires the 280 GOPS for the real-time operation which consists of 84% of D-CONV and 6% of T-CONV. However, the DT-CNN processor cuts down the 86.6% of the calculations by skipping the redundant computations with the proposed convolution array architecture with the delay cells.

The speed of the image segmentation can be represented in framerates, and the improvement of framerate of the proposed DT-CNN processor is shown in Fig. 6(b). The baseline is defined as the image segmentation of 288×288 image, including calculations of all the virtual zeros appeared in D-CONV and T-CONV in the DT-CNN processor. The baseline only obtains 9.3 fps and fails to meet the real-time requirement. However, skipping the redundant computations by the delay cell attains 68.4 fps in full-size image segmentation. Moreover, the image segmentation on small RoI sizes can additionally boost the performance. In the 128×192 sized RoI, the framerates increases up to 211 fps. Finally, the proposed DT-CNN processor implements the high throughput image segmentation by the delay cell and small-sized RoI based computations.

Fig. 7 describes the image segmentation results by the proposed DT-CNN processor. The tested network was the light-weight image segmentation network [4] and is newly trained in the 288×288 sized Freiburg sitting people dataset [5]. Trained D-CONV in the model consists of four types of dilation rates, 2, 4, 8, and 16. Based on this configuration, the segmentation accuracy in the 144×192 and 128×192 sized RoI are tested. In addition, it includes the result of image segmentation with the proposed dynamic dilation rate adjustment. As a result, The full-size image segmentation

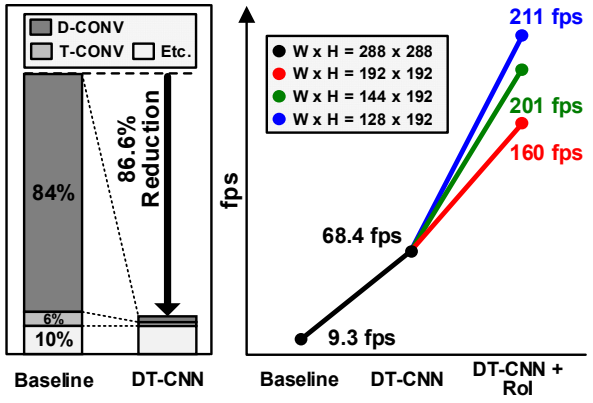


Fig. 6. The Performance Summary: (a) Reduced Convolution Operation Ratio for DT-CNN and (b) Framerate Enhancement in Different RoI Sizes.

obtained the 0.52 points in mIoU. However, RoI based image segmentation shows large mIoU drop, especially around the torso. Moreover, the result of RoI in 128×192 size is too messy to segment the body parts separately. On the other hand, DT-CNN processor with the dynamic dilation rate adjustment can recognize the body parts clearly and maintain the high segmentation accuracy. In specific, the 144×192 sized RoI regulates the dilation rate to 2, 4, 6, and 12, and the 128×192 sized RoI adjusts the rate to 2, 4, 6, 10. Changing dilation rate can change the size of the receptive field, therefore can overcome the accuracy degradation in different RoI sizes. Since the delay cell of the DT-CNN processor is able to compute the convolution with different dilation sizes, there is no computational overhead due to the dynamic dilation rate adjustment.

IV. CONCLUSION

The proposed DT-CNN processor accelerates both D-CONV and T-CONV for RoI based image segmentation. The DT-CNN processor with the delay cell and the dilation rate adjustments achieved 211 fps segmentation throughput with the 196 mW power consumption. By skipping the redundant calculations in D-CONV and T-CONV, it can drastically improve the throughput 6.7× higher than before. In addition, the RoI based image segmentation reduces 29.6% of the overall required computations. To minimize the accuracy degradation due to the small sized RoI, the proposed DT-CNN processor compensates for the RoI size by dynamically adjusting the dilation rates. Finally, the proposed DT-CNN processor successfully performs image segmentation with high accuracy, satisfying real-time constraint (211 fps) and low power consumption (196 mW).

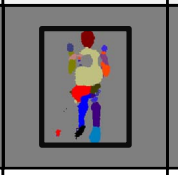
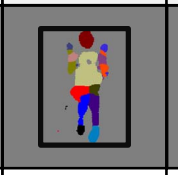
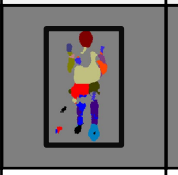
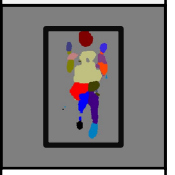
	Test Image	Ground Truth	Full Size	RoI Size	This Work	RoI Size	This Work
Image							
Image Size (W x H)	288 x 288	288 x 288	288 x 288	144 x 192	144 x 192	128 x 192	128 x 192
Max. Dilation Rate	-	-	16	16	12	16	10
mIoU	-	-	0.5200	0.4659	0.5105	0.4345	0.5161
ΔmIoU	-	-	0%	- 10.4%	- 1.8%	- 16.4%	-0.0%

Fig. 7. The Image Segmentation Results of the Different RoI sizes

V. REFERENCE

[1] C. Xu and J. J. Corso, "Actor-Action Semantic Segmentation with Grouping Process Models," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 3083-3092.

[2] A. Howard, et al., “MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[3] D. Shin, J. Lee, J. Lee and H. Yoo, "14.2 DNPU: An 8.1TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks," 2017 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, 2017, pp. 240-241.

[4] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, 2016.

[5] Gabriel L. Oliveira, A. Valada, C. Bollen, W. Burgard, Thomas Brox, “Deep Learning for Human Part Discovery in Images,” IEEE International Conference on Robotics and Automation (ICRA), 2016.

[6] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim and H. Yoo, "UNPU: An Energy-Efficient Deep Neural Network Accelerator With Fully Variable Weight Bit Precision," in IEEE Journal of Solid-State Circuits..

[7] Y. Chen, T. Krishna, J. S. Emer and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," in IEEE Journal of Solid-State Circuits, vol. 52, no. 1, pp. 127-138, Jan. 2017.

[8] R. Nahar, A. Baranwal and K. M. Krishna, "FPGA based parallelized architecture of efficient graph based image segmentation algorithm," 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, 2017, pp. 98-103.