JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 16, No. 2, June 2022, pp. 113-119

Review of Optimal Convolutional Neural Network Accelerator Platforms for Mobile Devices

Hyun Kim*

Department of Electrical and Information Engineering, Research Center for Electrical and Information Technology, Seoul National University of Science and Technology, Seoul, Korea

hyunkim@seoultech.ac.kr

Abstract

In recent years, convolutional neural networks (CNNs) have achieved remarkable performance enhancement, and researchers have endeavored to use CNN applications on power-constrained mobile devices. Accordingly, low-power and high-performance CNN accelerators for mobile devices are receiving significant attention. This paper presents the overall process of designing optimal CNN accelerator platforms for mobile devices based on algorithm, architecture, and memory system co-design while introducing various existing studies related to specific research fields.

Category: Embedded Systems

Keywords: Convolutional neural networks; Mobile device; Network compression; Hardware accelerator; Low-power memory

I. INTRODUCTION

With the recent intensive development of hardware accelerators, including graphics processing units (GPUs) and field-programmable gate arrays (FPGAs), various convolutional neural networks (CNNs) have achieved significant performance improvement in computer vision tasks, such as classification [1-9], object detection [10-13], and segmentation [14-19]. In particular, object detection and segmentation based on CNNs are utilized in various practical applications, and the demand for operating CNN applications in mobile devices such as smartphones and autonomous driving is increasing to reduce the tremendous amount of computation and solve memory resource problems imposed on the cloud server [5-9, 20].

However, CNNs require high computational costs because they involve using deep hidden layers to achieve high accuracy, which consequently results in high power consumption [21-23]. Thus, cost-effective and power-efficient hardware accelerators for mobile devices are required to operate CNN applications in real-time on power-constrained mobile devices [21, 24]. In general, most researchers use GPUs to operate CNNs; however, GPUs exhibit various problems such as large size, high cost, and power issues. Recently, well-designed FPGA and application-specific integrated circuit (ASIC) platforms have been reported to be more cost-effective and power-efficient than GPU platforms [21, 24-30]. Hence, the overall process for implementing the optimal CNN accelerator for mobile devices on FPGA and ASIC platforms is introduced in this study. Herein, we present the co-design technique of three levels (i.e., algorithm, architecture, and memory) for optimal CNN accelerator

Open Access http://dx.doi.org/10.5626/JCSE.2022.16.2.113 http://jcse.kiise.org

Received 20 May 2022; **Accepted** 08 June 2022

*Corresponding Author

pISSN: 1976-4677 eISSN: 2093-8020

platforms and introduce various studies on each level. Please note that this study is extended from [31]. Herein, we present the co-design technique of three levels (i.e., algorithm, architecture, and memory) for optimal CNN accelerator platforms and introduce various studies on each level.

II. OVERALL PROCESS OF IMPLEMENTING OPTIMAL CNN ACCELERATOR PLATFORMS FOR MOBILE DEVICES

A. Overall Process for Optimal CNN Accelerator Platforms

CNN accelerator platforms for mobile devices are implemented to achieve the following three goals: (1) high accuracy, (2) fast processing speed, and (3) low power consumption. As shown in Fig. 1, optimization must be performed at the algorithm, architecture, and memory levels to achieve these goals.

First, the CNN must be optimized by applying performance enhancement and software-based low complexity schemes while considering the trade-off between accuracy and power consumption (i.e., the algorithm-level approach). The algorithm-level approach is required because network optimization is the only method available to improve accuracy and reduce the network size. By conducting network optimization, a network that can achieve high accuracy with less computation can be developed. Second, based on this optimized network, application-specific hardware accelerators are designed on FPGA or ASIC platforms by applying hardware-based low complexity schemes (i.e., the architecture-level approach). The architecture-level approach is used to ensure that the optimal CNN can be accelerated through dedicated hardware support to achieve fast processing with low power consumption. As mentioned above, because the well-designed FPGA and ASIC platforms can outperform the GPU platforms in terms of cost-effectiveness and power efficiency, many application-specific hardware accelerators dedicated to CNNs have been proposed recently, focusing on FPGA and ASIC platforms. Finally, low-power memory systems for the CNN also should be developed (i.e., the memory-level approach). The memory-level approach is required because CNNs require significant memory access; hence, the memory power constitutes a significant portion of the total power in artificial intelligence (AI) systems. Consequently, by integrating these multilevel approaches, an optimal CNN accelerator platform for mobile devices can be realized.

B. Algorithm Level

At the algorithm level, the performance enhancement and network compression schemes should be performed

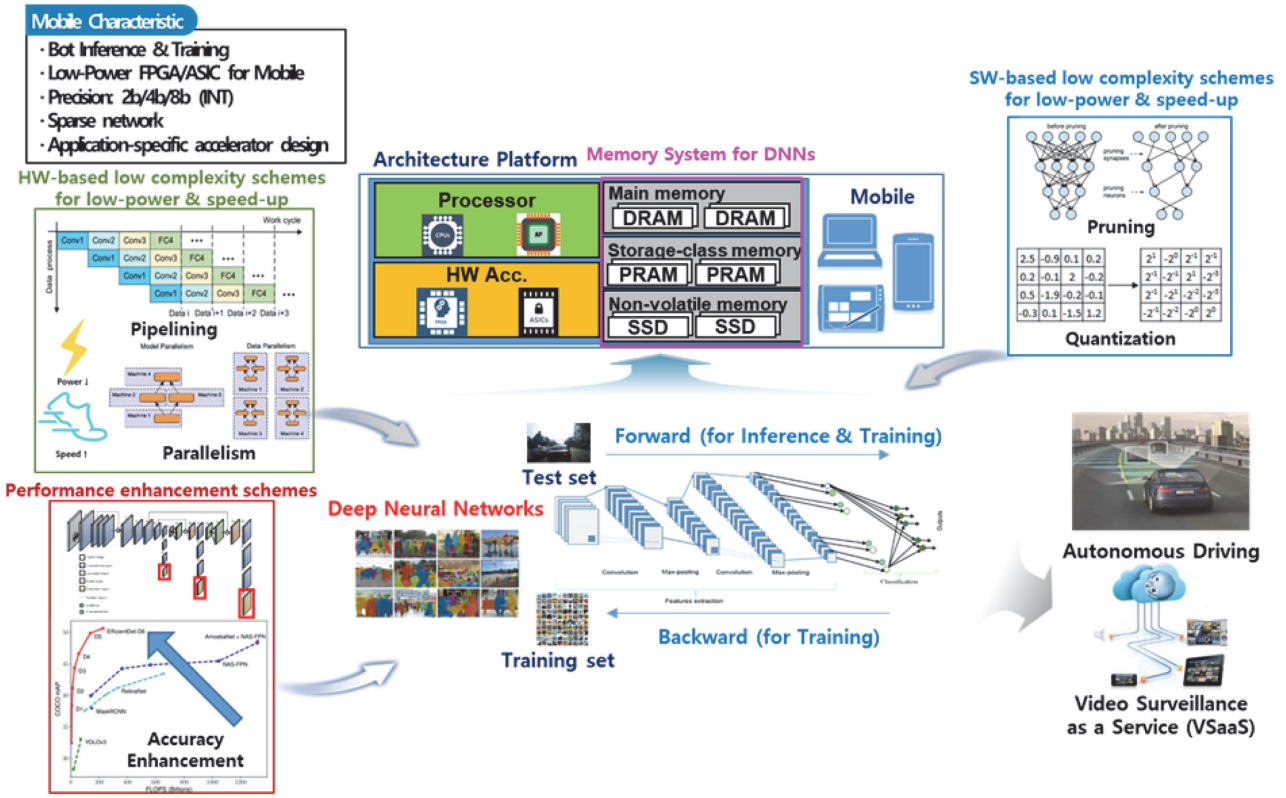

Fig. 1. Overall process of designing optimal hardware accelerator platforms for convolutional neural networks.

first. During this process, the trade-off between accuracy and computation amount must be considered. We introduce the representative studies for each of the two approaches in the following subsections.

1) Performance Enhancement Scheme

Performance enhancement is the only method that can be used to achieve high accuracy [2-4, 20, 23, 32]. In particular, to achieve real-time and low-power operations in mobile devices, accuracy must be improved, and any increase in the computation amount should be minimized.

For image classification tasks, Xie et al. [2] proposed a technique for adversarial training that involves different distributions of a clean image and an adversarial image. In the proposed technique, a separate auxiliary batch norm was introduced for adversarial examples, and different distributions of inputs were controlled. Consequently, significant improvements were achieved on various ImageNet datasets. Zhang et al. [3] proposed a self-distillation method that distills a model's internal knowledge to enhance the performance of CNNs. Self-distillation is desirable for mobile devices as it allows a trade-off between accuracy and resources using classifiers at different depths. He et al. [4] investigated the performances of various existing CNN training techniques by adjusting the batch size, learning rate, and batch norm. By applying these refinements, they improved the accuracy of various CNN models significantly.

For object detection tasks, Choi et al. [20] proposed a simple and efficient method to increase network performance. The parameters of the bounding box were modeled as Gaussian parameters, and the reliability of the network was improved using variance as the localization uncertainty. Bochkovskiy et al. [12] attempted to optimize learning and testing by combining typical and practical features in object detection models, including weighted-residual-connections, cross-stage-partial-connections, cross mini-batch normalization, self-adversarial training, and Mish activation. Liu et al. [11] proposed a method that involved the application of a small convolutional filter to a feature map to predict the category scores and box offsets for a fixed set of default bounding boxes.

In segmentation tasks, Bolya et al. [17] proposed a simple fully convolutional model for real-time instance segmentation, named YOLACT, by segregating instance segmentation into two parallel subtasks: generating a set of prototype masks and predicting per-instance mask coefficients. Wang et al. [18] introduced a novel notion of instance categories (i.e., quantized center locations and object sizes), which enables objects to be identified as segmented locations to distinguish object instances in an image. Liu et al. [32] proposed a bottom-up path to conveniently deliver low-level feature information to a high level. By reducing the information flow through which information was passed, low-level information was preserved, and the network performance was improved.

2) Software-based Low Complexity Schemes

The network compression technique reduces the computation amount (i.e., TOPS) required in the network. It involves low-complexity architecture design [5-9], quantization [33-37], pruning [38-41], and knowledge distillation [42-45], which can eliminate unnecessary operations while minimizing accuracy loss.

For low-complexity architecture design, Howard et al. [5] proposed MobileNets, designed for mobile, and embedded vision applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build lightweight deep neural networks (DNNs). They introduced two simple global hyperparameters that efficiently solved the trade-off between latency and accuracy. Sandler et al. [6] proposed MobileNetV2, where a new neural network architecture tailored for mobile and resource-constrained environments was introduced. Tan et al. [8] proposed an automated mobile neural architecture search approach, which explicitly incorporates model latency into the main objective to ensure that the search can identify a model that achieves a good trade-off between accuracy and latency. To further achieve a balance between flexibility and search space size, they proposed a novel factorized hierarchical search space that encourages layer diversity throughout the network.

In terms of the quantization approach, Zhou et al. [33] proposed DoReFa-Net for training CNNs with low-bit weights, activations, and gradients. Specifically, in the backward pass, the gradient is stochastically quantized with low bits. By allowing low-bit activation and gradient transfer in forward and backward passes, both training and inference processes can be accelerated using a bit convolution kernel. Choi et al. [34] proposed the parameterized clipping activation (PACT) method, which uses an activation clipping parameter optimized during training to identify the appropriate quantization scale. PACT allows activations to be quantized to arbitrary bit precisions while achieving outstanding accuracy. Li et al. [35] proposed the additive powers-of-two (APoT) quantization method, an efficient non-uniform quantization scheme for the bell-shaped and long-tailed distribution of weights and activations in CNNs. By constraining all quantization levels as the sum of powers-of-two terms, APoT quantization offers high computational efficiency and a close match with the distribution of weights.

In terms of the pruning approach, Yu et al. [38] introduced switchable batch normalization to stably train slimmable networks. Compared to individually trained models of the same width, slimmable networks demonstrate similar or better performances in terms of classification, object detection, instance segmentation, and keypoint detection. He et al. [39] proposed a soft filter pruning (SFP) approach to accelerate deep CNNs. SFP selects filters for every epoch based on the L2 norm, and the selected filters are then removed in softly. Yu et al. [41] proposed the scalpel, which customizes DNN pruning for the underlying

hardware by matching the pruned network structure with the data-parallel hardware organization via two approaches: (1) SIMD-aware weight pruning, which utilizes aligned fixed-size groups and (2) node pruning, which removes unnecessary nodes.

In terms of the knowledge distillation approach, Hinton et al. [42] proposed the notion of knowledge distillation based on the "temperature" parameter and "specialist" model training for confusing data to compress ensemble models and improve generalization performance. Romero et al. [43] proposed the "hint (intermediate hidden layer)" distillation method, which uses a regressor to distill feature maps. This method allows the training of deeper and thinner students that can generalize better or run faster. Park et al. [45] proposed a relational knowledge distillation method that defines distance and angle difference to transfer mutual relations of data examples from a teacher network. This method allows the teacher to transfer relational information between instances to the student network.

C. Architecture Level

At the architecture level, hardware accelerators must be designed appropriately such that the optimal network designed via the algorithm-level technique can be operated efficiently. In general, studies regarding CNN hardware accelerator design based on FPGA and ASIC platforms are conducted based on two aspects: the streaming hardware structure [21, 24-26] and scalable/recursive hardware structure [27-30].

In the streaming hardware approach structure, Nakahaka et al. [25] proposed a lightweight YOLOv2 composed of a binarized CNN for feature extraction and parallel support vector regression for classification and localization. Nguyen et al. [21] proposed an efficient TOPS streaming architecture design to reduce external memory access and achieve high throughput and power efficiency. In this design, a new data path was proposed to maximize the efficiency of weight parameter reuse, and the binary weights of the entire network were stored in on-chip memory using binary weights and flexible low-bit activations. Wu et al. [26] implemented MobileNet with reduced operations and parameters using depthwise separable convolution as a hardware accelerator. A MobileNet designed with channel augmentation architecture can be flexibly deployed to devices with different configurations to balance resources and computational performance.

In terms of the scalable hardware structure approach, Ding et al. [27] proposed a method of designing a basic computing block using the recursive property of the fast Fourier transform (FFT). This method allows the FFT to be implemented as a basic computing block because the FFT exhibits the recursive property and demonstrates an intrinsic role, thereby affording a small-footprint implementation. Furthermore, this method focuses on weight storage and memory management, where a typical network structure is leveraged to exploit the benefits of pipelines and parallelization to perform optimization across platforms. Ma et al. [29] proposed a specific dataflow for CNN hardware acceleration to minimize data communication while maximizing the resource utilization to achieve high performance. Lu et al. [30] proposed a weight-oriented dataflow to efficiently manage irregular connections and a weight layout to enable efficient on-chip memory access without conflicts.

Investigations regarding high-level synthesis (HLS) for automating this hardware design process are being actively conducted [46-49]. HLS is a technology that allows gate-level RTL designs to be realized through code implementations at high levels (i.e., C, PyTorch, TensorFlow, etc.). Canis et al. [47] introduced a new open-source HLS tool named LegUp, which allows software techniques to be used for hardware design. LegUp accepts a standard C program as input and automatically compiles the program to a hybrid architecture containing an FPGA-based MIPS soft processor and custom hardware accelerators that communicate through a standard bus interface. Pilato et al. [46] presented Bambu, a modular and open-source HLS framework that can synthesize complex applications in hardware. This framework allows numerous C constructs to be supported and is applicable to different technologies and devices. Furthermore, this framework allows the simulation results to be validated based on the software counterpart and appropriate system-level interfaces to be generated. Noronha et al. [49] proposed an open-source tool flow that maps numerical computation models written in TensorFlow to synthesizable hardware.

D. Memory Level

At the memory level, a memory system that reflects the characteristics of CNNs accurately should be developed. Most existing studies focus on low-power and high-speed memory platforms dedicated to CNNs as they offer power reduction and speed improvement to the CNN accelerator platform [22, 51-53].

Nguyen et al. [22] presented an approximate memory architecture that can significantly reduce both the refresh energy and dynamic energy consumption by applying a combination of soft and hard approximations to non-critical data while preserving the accuracy of error-resilient applications such as DNNs. Kim et al. [50] proposed a precision-controlled memory system that allows flexible management at the hard approximation level to reduce data movement, thereby improving energy saving and system performance. Nguyen et al. [51] presented an encoding scheme that enables refreshless DRAMs for a high-performance and energy-efficient deep learning system known as the ZEM. To reduce the

probability of a reload requirement, they presented a zero-cycle bit-masking technique that does not require changes to the DRAM interface and device. Koppula et al. [52] proposed the EDEN, the first general framework that reduces DNN energy consumption and DNN evaluation latency using approximate DRAM devices while satisfying the user-specified target DNN accuracy.

III. CONCLUSION

In this paper, we provided a review of the optimal design of CNN accelerator platforms for mobile devices. The CNN accelerator platform optimized across three levels (i.e., algorithm, architecture, and memory system) can satisfy the three requirements for operating CNNs on mobile devices (i.e., high accuracy, fast processing speed, and low power consumption) and facilitate the utilization of practical CNN applications in mobile environments, such as autonomous driving and smart factory. AI is expected to be commercialized rapidly through the development of optimal CNN accelerators that offer outstanding performance.

ACKNOWLEDGMENTS

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01304, Development of Self-learnable Mobile Recursive Neural Network Processor Technology) and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. NRF-2019R1A6A1A03032119).

REFERENCES

1. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770-778.
2. C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," in *Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 816-825.
3. L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: improve the performance of convolutional neural networks via self distillation," in *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019, pp. 3712-3721.
4. T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 558-567.
5. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: efficient convolutional neural networks for mobile vision applications," 2017 [Online]. Available: https://arxiv.org/abs/1704.04861.
6. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4510-4520.
7. A. Howard, M. Sandler, B. Chen, W. Wang, L. C. Chen, M. Tan, et al., "Searching for MobileNetV3," in *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019, pp. 1314-1324.
8. M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: platform-aware neural architecture search for mobile," in *Proceeding of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 2815-2823.
9. K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: more features from cheap operations," in *Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 1577-1586.
10. L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikainen, "Deep learning for generic object detection: a survey," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261-318, 2020.
11. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: single shot MultiBox detector," in *Computer Vision – ECCV 2016.* Cham, Switzerland: Springer, 2016, pp. 21-37.
12. A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," 2020 [Online]. Available: https://arxiv.org/abs/2004.10934.
13. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," 2015 [Online]. Available: https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.
14. I. Ulku and E. Akagunduz, "A survey on deep learning-based architectures for semantic segmentation on 2D images," 2022 [Online]. Available: https://doi.org/10.1080/08839514.2022.2032924.
15. J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *in Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 2015, pp. 3431-3440.
16. L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, 2018, pp. 801-818.
17. D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT: real-time instance segmentation," in *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019, pp. 9156-9165.

18. X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "SOLO: segmenting objects by locations," in *Computer Vision – ECCV 2020*. Cham, Switzerland: Springer, 2020, pp. 649-665.
19. K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 2961-2969.
20. J. Choi, D. Chun, H. Kim, and H. J. Lee, "Gaussian YOLOv3: an accurate and fast object detector using localization uncertainty for autonomous driving," in *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019, pp. 502-511.
21. D. T. Nguyen, T. N. Nguyen, H. Kim, and H. Lee, "A high-throughput and power-efficient FPGA implementation of YOLO CNN for object detection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 8, pp. 1861-1873, 2019.
22. D. T. Nguyen, N. H. Hung, H. Kim, and H. J. Lee, "An approximate memory architecture for energy saving in deep learning applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1588-1601, 2020.
23. J. Choi, D. Chun, H. J. Lee, and H. Kim, "Uncertainty-based object detector for autonomous driving embedded platforms," in *Proceedings of 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, Genova, Italy, 2020, pp. 16-20.
24. D. T. Nguyen, H. Kim, and H. J. Lee, "Layer-specific optimization for mixed data flow with mixed precision in FPGA design for CNN-based object detectors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 6, pp. 2450-2464, 2021.
25. H. Nakahara, H. Yonekawa, T. Fujii, and S. Sato, "A lightweight YOLOv2: a binarized CNN with a parallel support vector regression for an FPGA," in *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Monterey, CA, USA, 2018, pp. 31-40.
26. D. Wu, Y. Zhang, X. Jia, L. Tian, T. Li, L. Sui, D. Xie, and Y. Shan, A high-performance CNN processor based on FPGA for MobileNets," in *Proceedings of 2019 29th International Conference on Field Programmable Logic and Applications (FPL),* Barcelona, Spain, 2019, pp. 136-143.
27. C. Ding, S. Liao, Y. Wang, Z. Li, N. Liu, Y. Zhuo, et al., "CirCNN: accelerating and compressing deep neural networks using block-circulant weight matrices," in *Proceedings of 2017 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Boston, MA, USA, 2017, pp. 395-408.
28. K. Hegde, J. Yu, R. Agrawal, M. Yan, M. Pellauer, and C. Fletcher, "UCNN: exploiting computational reuse in deep neural networks via weight repetition," in *Proceedings of 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA),* Los Angeles, CA, USA, 2018, pp. 674-687.
29. Y. Ma, Y. Cao, S. Vrudhula, and J. Seo, "Optimizing the convolution operation to accelerate deep neural networks on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 7, pp. 1354-1367, 2018.
30. L. Lu, J. Xie, R. Huang, J. Zhang, W. Lin, and Y. Liang, "An efficient hardware accelerator for sparse convolutional neural networks on FPGAs," in *Proceedings of 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, San Diego, CA, USA, 2019, pp. 17-25.
31. H. Kim, "Implementation of optimal CNN accelerators for mobile devices: algorithm, architecture, and memory system co-design," in *Proceedings of 2021 18th International SoC Design Conference (ISOCC)*, Jeju Island, Korea, 2021, pp. 237-238.
32. S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 8759-8768.
33. S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: training low bitwidth convolutional neural networks with low bitwidth gradients," 2016 [Online]. Available: https://arxiv.org/abs/1606.06160.
34. J. Choi, Z. Wang, S. Venkataramani, P. I. J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: parameterized clipping activation for quantized neural networks," 2018 [Online]. Available: https://arxiv.org/abs/1805.06085.
35. Y. Li, X. Dong, and W. Wang, "Additive powers-of-two quantization: an efficient non-uniform discretization for neural networks," 2019 [Online]. Available: https://arxiv.org/abs/1909.13144.
36. Z. Song, B. Fu, F. Wu, Z. Jiang, L. Jiang, N. Jing, and X. Liang, "DRQ: dynamic region-based quantization for deep neural network acceleration," in *Proceedings of 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, Valencia, Spain, 2020, pp. 1010-1021.
37. S. Kim and H. Kim, "Zero-centered fixed-point quantization with iterative retraining for deep convolutional neural network-based object detectors," *IEEE Access*, vol. 9, pp. 20828-20839, 2021.
38. J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," 2018 [Online]. Available: https://arxiv.org/abs/1812.08928.
39. Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," 2018 [Online]. https://arxiv.org/abs/1808.06866.
40. X. Ding, T. Hao, J. Tan, J. Liu, J. Han, Y. Guo, and G. Ding, "ResRep: lossless CNN pruning via decoupling remembering and forgetting," 2020 [Online]. Available: https://arxiv.org/abs/2007.03260.
41. J. Yu, A. Lukefahr, D. Palframan, G. Dasika, R. Das, and S. Mahlke, "Scalpel: customizing DNN pruning to the underlying hardware parallelism," in *Proceedings of 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA),* Toronto, ON, Canada, 2017, pp. 548-560.
42. G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015 [Online]. Available: https://arxiv.org/abs/1503.02531
43. A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: hints for thin deep nets," 2014 [Online]. Available: https://arxiv.org/abs/1412.6550.
44. S. Zagoruyko and N. Komodakis, "Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer," 2016 [Online]. https://

arxiv.org/abs/1612.03928v1.
45. W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* Long Beach, CA, USA, 2019, pp. 3967-3971.
46. C. Pilato and F. Ferrandi, "Bambu: a modular framework for the high level synthesis of memory-intensive applications," in *Proceedings of 2013 23rd International Conference on Field programmable Logic and Applications*, Porto, Portugal, 2013, pp. 1-4.
47. A. Canis, J. Choi, M. Aldham, V. Zhang, A. Kammoona, J. H. Anderson, S. Brown, and T. Czajkowski, "LegUp: high-level synthesis for FPGA-based processor/accelerator systems," in *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, Monterey, CA, USA, 2011, pp. 33-36.
48. F. Winterstein, S. Bayliss, and G. A. Constantinides, "High-level synthesis of dynamic data structures: A case study using Vivado HLS," in *Proceedings of 2013 International Conference on Field-Programmable Technology (FPT)*, Kyoto, Japan, 2013, pp. 362-365.
49. D. H. Noronha, B. Salehpour, and S. J. E. Wilton, "LeFlow: enabling flexible FPGA high-level synthesis of TensorFlow deep neural networks," in *Proceedings of FSP Workshop 2018; Fifth International Workshop on FPGAs for Software Programmers*, Dublin, Ireland, 2018, pp. 1-8.
50. B. Kim, S. H. Lee, H. Kim, D. T. Nguyen, M. S. Le, I. J. Chang, et al., "PCM: precision-controlled memory system for energy efficient deep neural network training," in *Proceedings of 2020 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, Grenoble, France, 2020, pp. 1199-1204.
51. D. T. Nguyen, N. M. Ho, M. S. Le, W. F. Wong, and I. J. Chang, "ZEM: zero-cycle bit-masking module for deep learning refresh-less DRAM," *IEEE Access*, vol. 9, pp. 93723-93733, 2021.
52. S. Koppula, L. Orosa, A. G. Yaglikci, R. Azizi, T. Shahroodi, K. Kanellopoulos, and O. Mutlu, "EDEN: enabling energy-efficient, high-performance deep neural network inference using approximate DRAM," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, Columbus, OH, USA, 2019, pp. 166-181.

Hyun Kim https://orcid.org/0000-0002-7962-657X

Hyun Kim received the B.S., M.S. and Ph.D. degrees in Electrical Engineering and Computer Science from Seoul National University, Seoul, Korea, in 2009, 2011 and 2015, respectively. From 2015 to 2018, he was with the BK21 Creative Research Engineer Development for IT, Seoul National University, Seoul, Korea, as a BK Assistant Professor. In 2018, he joined the Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul, Korea, where he is currently working as an Assistant Professor. His research interests are in the areas of algorithm, computer architecture, memory, and SoC design for low-complexity multimedia applications and deep neural networks.