Low-Complexity Dynamic Channel Scaling of Noise-Resilient CNN for Intelligent Edge Devices

Younghoon Byun, Minho Ha, Jeonghun Kim, Sunggu Lee and Youngjoo Lee
Pohang University of Science and Technology (POSTECH), Pohang, Republic of Korea
{byh1321, mh0205, salmon, slee, youngjoo.lee}@postech.ac.kr

***Abstract*—In this paper, we present a novel channel scaling scheme for convolutional neural networks (CNNs), which can improve the recognition accuracy for the practical distorted images without increasing the network complexity. During the training phase, the proposed work first prepares multiple filters under the same CNN architecture by taking account of different noise models and strengths. We then newly introduce an FFT-based noise classifier, which determines the noise property in the received input image by calculating the partial sum of the frequency-domain values. Based on the detected noise class, we dynamically change the filters of each CNN layer to provide the dedicated recognition. Furthermore, we propose a channel scaling technique to reduce the number of active filter parameters if the input data is relatively clean. Experimental results show that the proposed dynamic channel scaling reduces the computational complexity as well as the energy consumption, still providing the acceptable accuracy for intelligent edge devices.**

Keywords—convolutional neural network, intelligent edge devices, noise-resilient processing

I. Introduction

Among the various machine learning (ML) algorithms, in the last decade, the convolutional neural network (CNN) has been actively applied to the recognition system by improving its accuracy continuously [1]. Following to the high-performance computing servers, dominating the ML market with massive-parallel operations [2], the edge devices using the raw inputs are now expected to have the intelligence by including CNN inference engines [3]. However, it is generally reported that raw images at the edge devices are subjected to various noise sources like illumination variation, non-linearity of sensors, temperature fluctuation, and camera defocusing [4]. As the distorted images severely degrade accuracy [5], the advanced noise-resilient scheme should be utilized in the intelligent edge devices.

Unfortunately, only few works are reported to reduce the effects of input noises for CNN-based recognition. Using the different noise models, the data augmentation is applied to extend the training set by including manually generated noisy images, retraining part of the network [6]. For the dynamic approach [7], the ensemble architecture is applied to the noise-resilient CNN by introducing the gating network that automatically adjusts the contribution ratio of CNNs dedicated to the different types of input noises. The concept of adding noise reduction networks is used for enhancing the quality of input images, achieving the accuracy of the original CNN targeting the clean images [4]. However, the previous dynamic methods basically add complex processing steps associated with the numerous convolution operations and filter parameters. As the hardware resources are limited in edge devices [8], it is required to develop a practical solution that overcomes the distorted inputs with the acceptable computing costs.

For the noise-resilient CNN system, in this work, we first separate the input noises into two types; Gaussian type and blur type. By changing the noise strength, we categorize input images into 7 noise classes, each of which has own trained filter parameters under the same CNN architecture. Then, we newly introduce an FFT-based noise classifier that reveals the noise class of the input image at the run time. Switching the dedicated parameters for each noise class, the proposed scheme achieves the attractive accuracy independent of the practical input noises. To realize the proposed noise-resilient CNN system in the resource-limited devices, the dynamic channel scaling is proposed to further reduce the number of parameters while providing the acceptable accuracy. By appending the channels, our noise-resilient CNN system remarkably relaxes the required memory usages and the computing overheads.

II. Noise Modeling of Input Images

In this work, we adopt two noise types for modeling the common and practical distorted images in edge devices. The first noise type is based on Gaussian noise caused by thermal variations at image sensors and interface circuits [4]. The pixel value of the reference image is denoted as $P_R(x, y)$, where (x, y) stands for the coordinate of the corresponding pixel. Using the histogram of $P_R(x, y)$, we calculate the mean and variance of the reference, which are represented as μ_R and σ_R, respectively. The image with Gaussian noise is then expressed as follows.

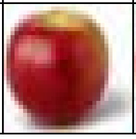
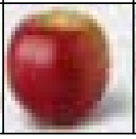
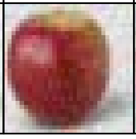
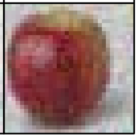
$$P_G(x,y) = P_R(x,y) + N_G(x,y) \tag{1}$$

where $P_G(x, y)$ and $N_G(x, y)$ represent the pixel values of the noisy image subjected to Gaussian type and the noise kernel containing the effects of Gaussian noise, respectively. Note that the kernel $N_G(x, y)$ follows a normal distribution of $(0, \sigma_G)$. By considering the practical noisy images, as depicted in Fig. 1(a), we define three noise classes by setting σ_G to $0.1\sigma_R$, $0.2\sigma_R$, and $0.3\sigma_R$, denoted as G1, G2, and G3, respectively.

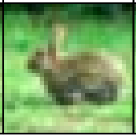
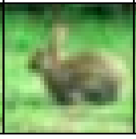
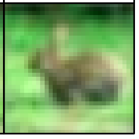
The second noise type is based on blurring effects. To formulate this type, we define the noise kernel $N_B(x, y)$ as

$$N_B(x,y) = \frac{1}{2\pi\sigma_B^2} e^{-\frac{(x-\lfloor 4\sigma_B+0.5\rfloor)^2+(y-\lfloor 4\sigma_B+0.5\rfloor)^2}{2\sigma_B^2}} \tag{2}$$

This work was supported by the Samsung Research Funding & Incubation Center of Samsung Electronics under Project Number SRFC-TB1703-07.

Class	R	G1	G2	G3
σ_G	0	$0.1\sigma_R$	$0.2\sigma_R$	$0.3\sigma_R$
Sample image				

(a)

Class	R	B1	B2	B3
σ_B	0	$0.9\sigma_R$	$1.35\sigma_R$	$1.7\sigma_R$
Sample image				

(b)

Fig. 1. Examples of noise classes for (a) Gaussian type and (b) blur type noises, where the reference images are from CIFAR-100 data [9].

The distorted image $P_B(x, y)$ including blurring effects is then modeled by using the two-dimensional (2D) convolution of $P_C(x, y)$ and $N_B(x, y)$, i.e., $P_B(x, y) = P_R(x, y) * N_B(x, y)$. Similar to Gaussian-type classes, as depicted in Fig. 1(b), three blur-type classes denoted as B1, B2, and B3 are defined by setting σ_B to $0.9\sigma_R$, $1.35\sigma_R$, and $1.8\sigma_R$, respectively. By adding the class R with the reference images, which are considered as clean ones, we finally categorize images into 7 classes by considering the practical noise types and strengths. With the noise kernels based on the variance of each reference image, we can construct more realistic noise models related to the actual sensing values, where the previous approaches only consider the deterministic amount of noises [6], [7]. We assume that our modeling is independent of the color as the color-aware model can be easily developed by multiplying the weighting functions to the original modeling. Based on the proposed noise models, to verify the noise-resilient CNNs, we manually generate the distorted images from the reference images at CIFAR-10 and CIFAR-100 data sets [9]. Therefore, each image in the extended data set always has one noise class used for training and validating operations.

III. FFT-Based Noise Classifier

To classify the input noises dynamically, in the previous work in [7], the convolution layers are trained and added to the original CNN architecture, which increases the recognition complexity and the energy consumption significantly. Before activating the CNN processing, we transfer the input pixel values in spatial domain $P_I(x, y)$ to the frequency domain values $F_I(x, y)$ by adopting the 2D fast Fourier transform (FFT) for selecting the input noise class. In the practical image having a clean object, in fact, the values of high-frequency regions in $F_I(x, y)$ tend to be in a certain range. Hence, analysis in the frequency domain is an acceptable pre-processing as the noise models in Section II actively modify the high-frequency signals. More precisely, the Gaussian-type kernel $N_G(x, y)$ whose magnitude in frequency domain is conceptually depicted in Fig. 2(a) is added to the reference image, increasing high-frequency signals depending on σ_G. In contrast, the blurred image is computed by the 2D convolution of $P_R(x, y)$ and $N_B(x, y)$, i.e., the multiplication in frequency domain. Considering the magnitude,

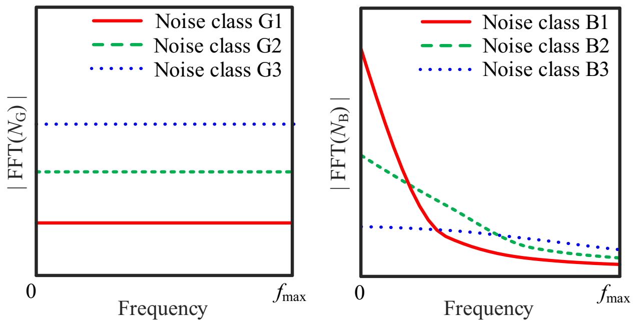


Fig. 2. Conceptual illustrations in the frequency domain of (a) Gaussian-type noise kernels and (b) blur-type noise kernels.

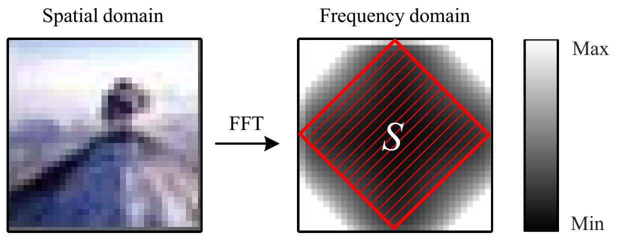


Fig. 3. The process of selecting the set S for PFSUM calculation.

as shown in Fig. 2(b), the blur-type kernel clearly reduces the high-frequency regions of reference images. Hence, checking the frequency domain of $P_I(x, y)$ leads to the run-time selection of the input noise class.

In the proposed FFT-based noise classifier, we calculate the partial frequency-sum (PFSUM) to simply observe the high-frequency values of input images. If the resolution of input image is $M \times M$, in detail, the PFSUM is computed as

$$\text{PFSUM} = \sum_{(i,j)\in S} F_I(i,j), \tag{3}$$

where S is the set of coordinates defined as

$$S = \left\{(i,j) \,\middle|\, \left|\frac{M-1}{2} - i\right| + \left|\frac{M-1}{2} - j\right| \le \frac{M}{2}\right\}. \tag{4}$$

Fig. 3 shows the process of PFSUM calculation where the set S covering high-frequency areas is denoted as a colored block. For the extended training set generated by the noise models from CIFAR-100 training data, the PFSUM distributions of noise classes are illustrated in Fig. 4. As we expected, PFSUM values of Gaussian-type classes are right-shifted compared to the distribution of the class R. Note that blur-type noise classes definitely reduce the PFSUM values. By setting the proper thresholds, the proposed classifier successfully finds the input noise class. In this work, we set thresholds to minimize the classification errors of noise classes for the extended training set. Then, these thresholds are verified by using the extended validation sets, which are also generated by the validation data of CIFAR-100. As shown in Fig. 5, the FFT-based classifier adopting the proposed thresholds achieves the classification accuracy of 68% on average. Note that the proposed classifier makes wrong decisions for some cases, which are caused by the

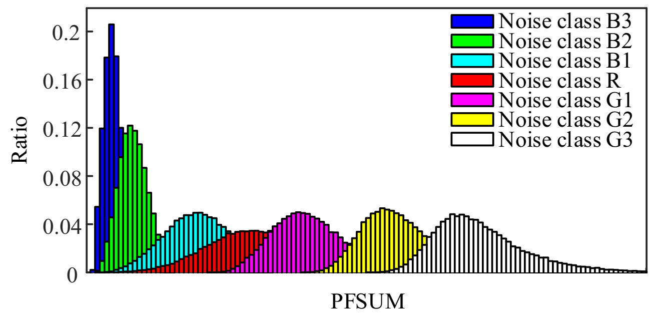


Fig. 4. The histogram of PFSUM values for each nosie class.

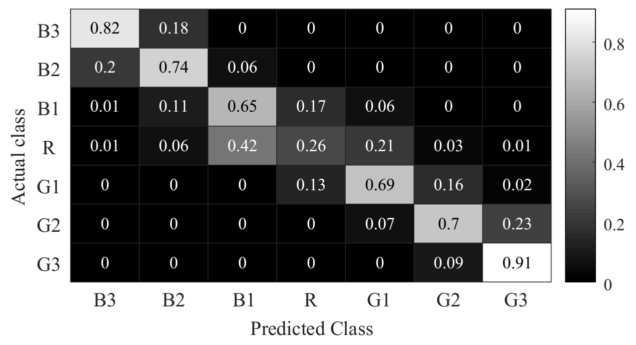


Fig. 5. The classification results of input noise classes using the proposed FFT-based noise classifier.

overlapped areas of adjacent PFSUM distributions shown in Fig. 4. However, the most of wrong decisions belong to the adjacent error classes, as depicted in Fig. 5, and the proposed noise-resilient CNNs, which will be discussed at the following section, can overcome these misdetections successfully.

The classification accuracy of noise types can be enhanced by utilizing gating networks [7] or convolution layers [4], [10], however, these schemes require much more computations and parameters. As the proposed classifier only introduces a small-sized FFT operation followed by the simple comparison unit, therefore, it is definitely attractive for the practical edge device where the hardware resources are strictly limited.

IV. Noise-Resilient CNN for Intelligent Edge Devices

A. *Swtiching the Dedicated Filter Parameters*

Fig. 6 illustrates the conceptual diagram of the proposed noise-resilient CNN system for embedded edge devices. After receiving a raw input from the image sensor, the FFT-based noise classifier first checks the class of the input noise, which is used for selecting the optimal filters by activating the memory controller. More precisely, different filter parameters are prepared in the parameter memory (PM), which is a high-speed DRAM in general. The noise-resilient CNN process is then started by accessing the dedicated filter parameters to CNN accelerator in Fig. 6, which consists of SRAM buffers and processing elements (PEs) including the registerfile (RF) and the computing unit. Note that only active parameters are loaded from DRAM to serve the dedicated process.

To train the parameters for each noise class, except for the reference class, the dedicated training set is constructed by combining the images in the corresponding noise class and the

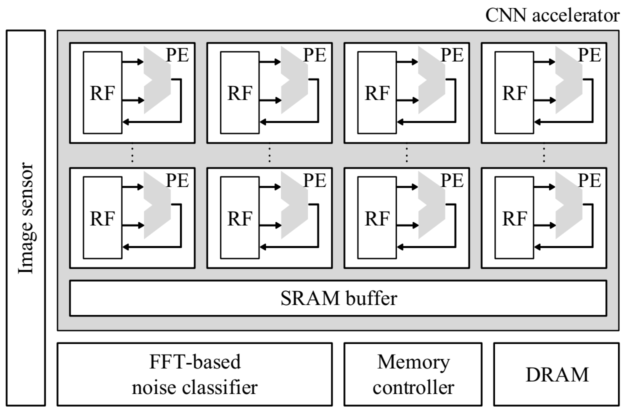


Fig. 6. The noise-resilient CNN systm at the embedded edge device.

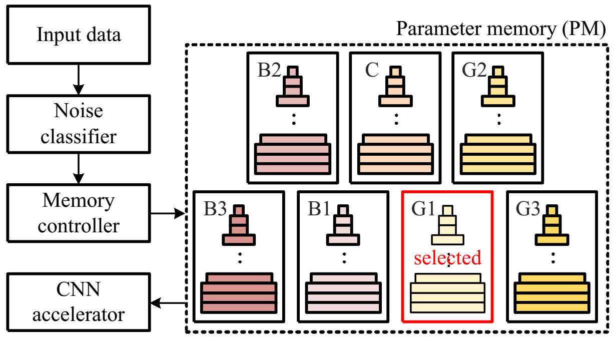


Fig. 7. The example of the dedicated filter selection from PM.

reference class. For example, the filter parameters dedicated to the class G1 is based on the extended images in G1 and the original reference images in R. Note that filter parameters for the class R are trained by only using the reference images, which makes the identical results to the original CNN architecture. By including the images in the class R for training the noise-aware channels, conceptually, we can capture both the essential features of target objects and the specific effects of noise models. As the FFT-based noise classifier can single out the noise type, shown in Fig. 7, the filter parameters are dynamically switched to the dedicated ones, leading to the attractive accuracy for any noisy input. That is, the proposed method provides the noise-resilient CNN system based on the real-time noise detection.

It is noticeable that all the dedicated filter parameters in this work are trained for the same CNN architecture, i.e., the networks for different noise classes requires the same number of layer as well as the same configurations of filters as shown in Fig. 7. It might be possible to train more dedicated CNN structures, which are separately optimized for different noise types. However, adjusting the network structure at the run time may request to change the number of layers, the order of layers, the size of filters, and even the inter-layer connections [11]. Therefore, it is impractical to support the network-level flexibility with the optimized CNN accelerator developed for battery-operated edge devices, using the fixed hardware to minimize the energy consumption in general [12], [13]. Hence, the proposed noise-resilient CNN is suitable for resource-limited systems as we just switch the filter parameters based on the same CNN architecture. Considering the size of PM, however, the proposed scheme clearly has a critical drawback as it uses 7 different networks, requiring 7 times larger storage space than the general CNN system having a single network. For the cost-effective embedded edge devices, therefore, more advanced technique is necessary to reduce the overall number of filter parameters with the acceptable accuracy.

B. Dynamic Channel Scaling for Edge Devices

As the switching-based CNN uses numerous networks, keeping their parameters at PM is another overhead at the edge device. To solve this issue, in this work, we additionally propose the dynamic channel scaling that enlarges the widths of CNN channels depending on the input noise class. Basically, the proposed technique relies on that the clean images in the class R are relatively easier to be recognized than the distorted ones. If the FFT-based classifier detects the class R, hence, it is possible to reduce the channel width of each layer, resulting the low-complexity network. More precisely, we reduce the channel width from C to $0.5C$ where C is the channel width of the original layer. By reducing the channel width in half, we can eliminates the half of filters, and at the same time, the depth of each filter is also halved as depicted in Fig. 8. As a result, the number of parameters for the class R is reduced by 4 times, remarkably relaxing the memory usages.

For the noisy classes, we then use stronger CNNs having the wider channel widths. In contrast that the noise-resilient CNN system in the previous subsection trains filter parameters for different noise classes individually as shown in Fig. 7, the stronger CNNs generated by the proposed channel scaling re-utilize the filters of the weak CNNs. To enlarge the channel, we only train the additional parameters and append them to the previously-trained filters, relaxing the required storage. As shown in Fig. 9(a), for example, the network for the weakest Gaussian noise, i.e., the class G1, is introduced by adding $0.125C$ channels to the network with $0.5C$ channels, which is already trained for the class R. Similarly, we prepare additional filters regarding the channel widths of $0.125C$ and $0.25C$ to make the dedicated channels for the class G2 and G3, respectively. Then, the stronger CNNs, that can tolerate more noises, are developed by adding these channels incrementally. The blur-type class is also considered as the same strategy. For example, as shown in Fig. 9(b), the dedicated CNN for B3 uses the full channel width of C by merging all the channels trained for blur-type classes. Hence, the proposed scheme dynamically scales the channel widths depending on the detected noise type, relaxing the PM accesses with the reduced total channel widths.

Although we adopt the different channel widths by taking into account input noises, we can still use the optimized CNN accelerator in Fig. 6. More precisely, the channel scaling only changes active parameters for each dedicated network, and the other CNN configurations are maintained regardless of noise classes. As reported in [11], the typical CNN accelerator easily supports the different numbers of parameters by adjusting the accessing patterns of PM. Therefore, the proposed channel scaling is suitable for the embedded edge devices to provide the low-complexity noise-resilient CNN process.

V. Experimental Results

For the given baseline network, we first define several CNN architectures used for our simulations as described in Table I. Note that CNN_{FS} and CNN_{CS} are the proposed noise-resilient CNN systems applying the filter switching and the channel scaling schemes, respectively. In addition, as depicted in Table I, we implement two previous noise-aware networks denoted as CNN_{D1} and CNN_{D2} based on the data augmentation from the extended noisy images [6]. Note that the number of active filter parameters in CNN_{D1} or CNN_{D2} is comparable to that of CNN_{FS} or CNN_{CS}, respectively, so that we can fairly compare the proposed noise-resilient CNN systems to the previous works having the similar complexities.

Based on the noise models, to develop the noise-resilient CNN system, Table II shows the recognition accuracy of each CNN architecture dedicated to each noise class. The baseline network is from the VGG-16 network for the CIFAR-100 data set [14], which is identical to CNN_R in the table. As CNN_R is only trained by using the clean images, it cannot be used for processing the noisy inputs. By training the filters dedicated to each noise type, as shown in Table II, we can remarkably improve the accuracy for the target error class. For example, 67.47% of the noisy data in G2 can be correctly categorized by

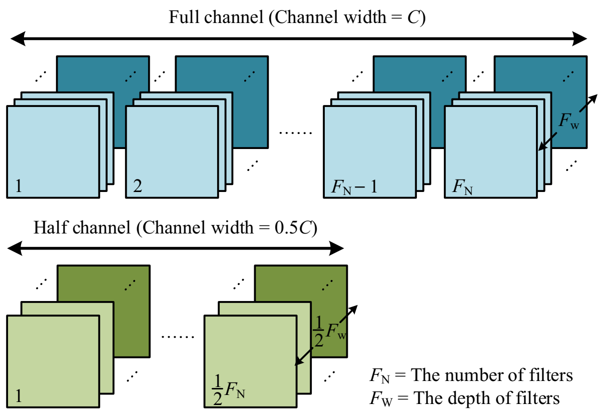

Full channel (Channel width = C)
1
2
......
F_N − 1
F_N
F_w
Half channel (Channel width = 0.5C)
1
......
1/2 F_N
1/2 F_w
F_N = The number of filters
F_W = The depth of filters

Fig. 8. The reduced number of parmaters by reducing the channel width.

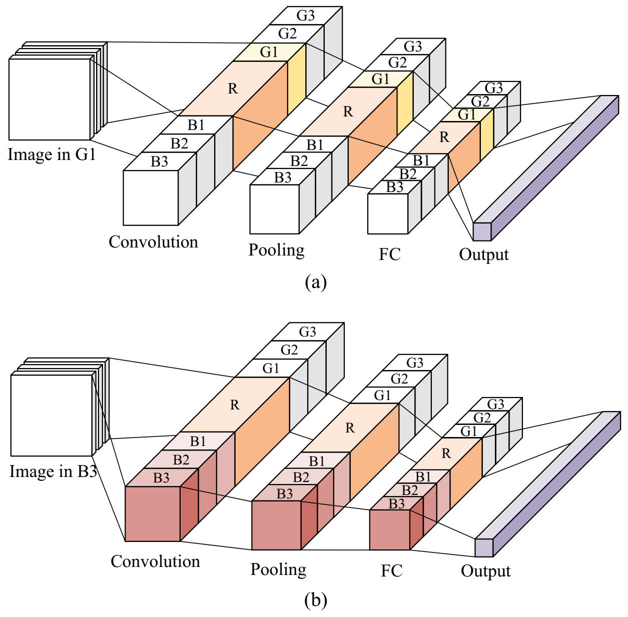


Fig. 9. The concept of the proposed channel scaling for the dedicated CNN architecture targeting (a) G1 class and (b) B3 class.

TABLE I. Definitions of CNN Architectures

Network	Definition
CNN_R	The network trained by using images in the class R
CNN_{Gx}	The networks trained by using images in the class G*x*
CNN_{Bx}	The networks trained by using images the class B*x*
CNN_{FS}	The proposed network based on the filter switching
CNN_{CS}	The proposed network based on the channel scaling
CNN_{D1}	The network using the data augmentation (Compared to CNN_{FS})
CNN_{D2}	The network using the data augmentation (Compared to CNN_{CS})

TABLE II. Performances of the Dedicated Networks

Network	Input noise classes							Average
	B3	B2	B1	R	G1	G2	G3	
CNN_{B3}	**67.75**	69.33	69.74	70.08	63.19	43.74	25.46	58.47
CNN_{B2}	64.15	**69.49**	70.66	71.08	62.12	41.08	23.49	57.44
CNN_{B1}	45.26	62.75	**71.38**	71.40	58.98	35.26	18.28	51.90
CNN_R	34.16	52.47	69.76	**72.03**	59.06	34.59	17.58	48.52
CNN_{G1}	35.05	53.28	68.96	71.06	**69.95**	62.31	46.25	58.12
CNN_{G2}	38.65	55.28	68.42	70.34	69.82	**67.47**	62.73	61.82
CNN_{G3}	40.72	54.65	65.69	67.34	66.45	65.05	**63.13**	60.43

TABLE III. Performances of the Noise-Resilient CNNs

Network	Input noise classes							Average
	B3	B2	B1	R	G1	G2	G3	
CNN_{FS}	66.80	68.88	70.44	71.54	67.12	66.93	63.03	67.82
CNN_{D1}	64.18	66.30	67.88	68.23	67.18	65.45	62.96	66.03
CNN_{CS}	61.88	64.14	66.74	67.33	64.44	62.44	59.86	63.83
CNN_{D2}	45.71	47.35	48.35	48.13	48.06	46.94	44.99	47.08

CNN_{G2}, whereas CNN_R achieves the recognition accuracy of 34.59% for the same class. Note that, the dedicated network always provides the best accuracy for the corresponding class.

Adopting the proposed FFT-based noise classifier, hence, the proposed noise-resilient CNN system switching the filter parameters dynamically, i.e., CNN_{FS}, can provide an attractive recognition accuracy even for the noisy inputs as depicted in Table III. Due to the misdetections of input noise classes, as shown in Fig. 5, the accuracy of CNN_{FS} is slightly degraded compared to the ideal accuracy from each dedicated network. However, the accuracy drop is negligible as the errors on the input noise class always select the adjacent networks, which also provide the acceptable accuracies as shown in Table II. Having the same number of active filters, it is also possible to develop a noise-aware network trained by using all the noisy images, i.e., CNN_{D1}. Shown in Table III, however, it is clear that the proposed CNN_{FS} achieves better accuracies for all the classes as CNN_{D1} cannot provide the dedicated filters. By scaling the channel widths at the run time, the proposed CNN_{CS} still offers the noise-tolerable recognition process with the acceptable accuracy. More precisely, the average accuracy of CNN_{CS} is reduced by 4% and 2.2% compared to those of CNN_{FS} and CNN_{D1}, respectively. Considering the complexity, however, the proposed channel scaling is superior to the other networks as summarized in Table IV. In terms of the total number of parameters, which are initialized to PM in Fig. 6, the proposed

TABLE IV. Complexity Comparisons of Different CNNs

Network	Normalized channel width	Number of total parameters	Average number of active parameters
CNN_R	1.00	15.3M (1.00)	15.3M (1.00)
CNN_{FS}	1.00	107.1M (7.00)	15.3M (1.00)
CNN_{D1}	1.00	15.3M (1.00)	15.3M (1.00)
CNN_{CS}	0.75	22.95M (1.50)	9.1M (0.77)
CNN_{D2}	0.91	12.6M (0.82)	12.6M (0.82)

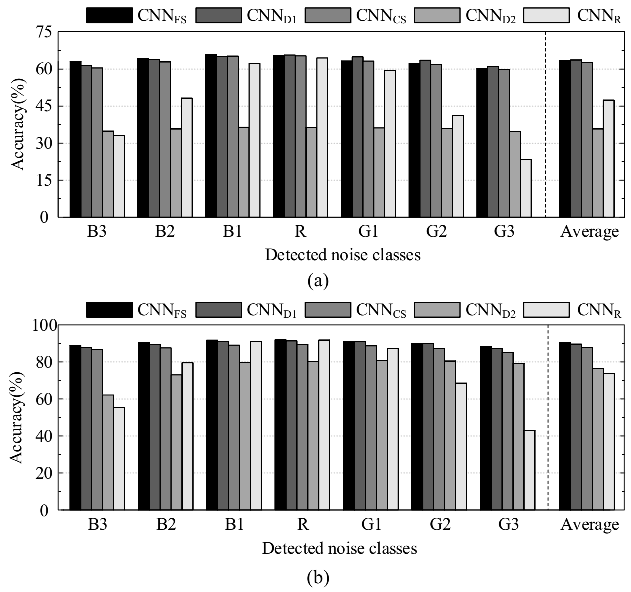


Fig. 10. The performance of CNNs based on (a) the ResNet-18 newtork for CIFAR-100 data, and (b) the VGG-16 network for CIFAR-10 data.

CNN_{FS} necessitate 7 times more variables than the reference network CNN_R although it uses the same number of active parameters. Note that CNN_{D1} always uses the same amount of parameters as CNN_R.

The proposed channel scaling still increases the overall PM size by 50% to store the additional filters. However, the channel width of CNN_{CS} varies depending on the input class, and we can finally reduce the normalized channel width as depicted in Table IV, i.e., utilizing 0.5-width channel for R class and increasing the width up to 1.0 for noise classes. As the number of active parameters directly affects to the number of multiply-accumulate (MAC) operations [15], therefore, the proposed channel scaling greatly relaxes the overall CNN complexity. The noise-aware network using the augmented noisy images can be also designed for the low-complexity system with the reduced channel width, which is denoted as CNN_{D2} in Table I. Even though the complexity of CNN_{D2} is comparable to that of CNN_{CS}, the accuracy of CNN_{D2} cannot be acceptable to the real application as shown in Table III.

In addition to the noise-resilient CNN system using the VGG-16 network for CIFAR-100 data set, we verify that the proposed approaches are still acceptable to the general system by performing different case studies. Using the same noise models in Section II, more precisely, we extend the given data set to make the distorted classes, and develop noise-resilient CNN systems starting from different baselines; the ResNet-18 for CIFAR-100 set [16], and the VGG-16 for CIFAR-10 set [9], where the recognition accuracies are compared in Fig. 10 (a) and

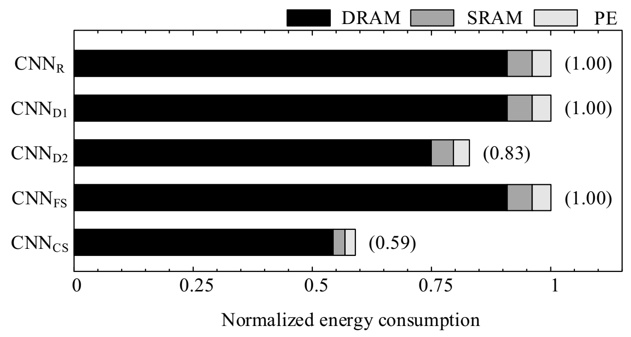


Fig. 11. The relative energy requirements of different CNNs.

(b), respectively. Note that the reference network CNN_R always suffers from the noise sources significantly, so the noise-resilient CNN system is meaningful for the devices accepting the raw inputs. By switching the filter parameters for each input image properly, the proposed CNN_{FS} provides the attractive recognition accuracy regardless of the detected noise classes. To further optimize the complexity of CNN_{FS}, the proposed channel scaling approach, i.e., CNN_{CS}, slightly degrades the average accuracy by reducing the number of active parameters. Therefore, the proposed CNN_{CS} is much more attractive to the embedded devices compared to the other CNN architectures. Due to the poor accuracy, as depicted in Fig. 10, CNN_{D2} cannot be the practical solution even it has a similar computational complexity to CNN_{CS}. As a result, the proposed dynamic channel scaling is effective to support the low-complexity and noise-resilient CNN system targeting the resource-limited embedded edge devices.

In order to analyze the advantage of the proposed dynamic channel scaling in terms of energy consumption, which is also a critical check-point for designing the embedded devices, we additionally develop the energy estimator of the typical CNN accelerator architecture shown in Fig. 6. By considering the previous works in [11], in the estimator, the data flow for a single inference process is modeled in 65nm CMOS process by tracking the operations of the external DRAM, the internal SRAM buffers, and the array of PEs. More precisely, the data flows inside of the CNN accelerator are scheduled to reuse the filter parameters maximally. Therefore, the energy estimator tries to minimize the energy of DRAM accesses in Fig. 6, which dominates the total CNN accelerator energy consumption [15].

Adopting the reported energy consumption for each part [17], as shown in Fig. 11, our energy estimator successfully generates the valid comparison of different CNN architectures mapping to the same CNN accelerator structure [11]. Note that CNN_{FS} and CNN_{D1} consume the same amount of energy as the reference network CNN_R, since they are using the same number of active parameters as depicted in Table IV. On the other hand, the proposed CNN_{CS}, which dynamically reduces the number of active filter parameters for low-level noises, directly saves the energy of the accelerator. According to the narrowed channel widths, each internal component, i.e., DRAM, SRAM and PE, saves the energy consumption remarkably as depicted in Fig. 11. As a result, the proposed CNN_{CS} saves the energy consumption for the noise-resilient CNN processing by 41% and 28% compared to the CNN_{FS} and CNN_{D2}, respectively, which is acceptable to the energy-efficient intelligent edge devices.

VI. Conclusion

In this paper, we have proposed the low-complexity CNN system that can be tolerable to the practical input noises. To understand the property of input noise, we use the theoretical models to develop the extended data sets from the original clean data, and categorize them into several classes. Then, for the first time, we introduce the FFT-based noise classifier that detects the noise class of input image at the run time. For each noise, the dedicated CNN architecture is trained to provide the accurate recognition process. In order to reduce the number of parameters, furthermore, we propose the dynamic channel scaling, which adjusts the channel width by appending the filters depending on the detected noise class. As a result, the proposed noise-resilient CNN system provides the attractive accuracy with the minimum hardware costs, leading to the cost-effective recognition process for intelligent edge devices.

References

[1] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. CVPR*, 2017. pp. 4700–4708.
[2] M. Imani, D. Peroni, Y. Kim, A. Rahimi, and T. Rosing, "Efficient neural network acceleration on GPGPU using content addressable memory," in *Proc. IEEE DATE*, 2017, pp. 1026–1031.
[3] J. Mao, X. Che, K. W. Nixon, C. Krieger, and Y. Chen, "Modnn: Local distributed mobile computing system for deep neural network," in *Proc. IEEE DATE,* 2017, pp. 1396–1401.
[4] S. Diamond, V Sitzmann, S. Boyd, G. Wetzstein, and F. Heide, "Dirty pixels: Optimizing image classification architectures for raw sensor data," [Online]. Available: https://arxiv.org/abs/1701.06487
[5] S. Dodge and L. J. Karam, "Understanding how image quality affects deep neural networks," in *Proc. IEEE QoMEX,* pp 1–6. 2016.
[6] Y. Zhou, S. Song, and N.-M. Cheung, "On classification of distorted images with deep convolutional neural networks," in *Proc. IEEE ICASSP*, 2017, pp 1213–1217.
[7] S. Dodge and L. J. Karam, "Quality robust mixtures of deep neural networks," *IEEE Trans. Image Process*., vol. 27, no. 11, pp. 5553–5562, Nov. 2018.
[8] S. Hong. I. Lee, and Y. Park, "NN compactor: Minimizing memory and logic resources for small neural networks," in *Proc. IEEE DATE*, 2018, pp. 581–584.
[9] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Technical report, University of Toronto, vol. 1, no. 4, 2009.
[10] J. Kim, A.-D. Nguyen, and S. Lee, "Deep CNN-based blind image quality predictor," *IEEE Trans. Neural Netw. Learn. Syst.*, in press.
[11] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits (JSSC)*, vol. 52, no. 1, pp. 127-138, Nov. 2017.
[12] J. Yue *et al*., "A 3.77TOPS/W convolutional neural network processor with priority-driven kernel optimization," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, in press.
[13] J. Lee *et al*., "UNPU: A 50.6TOPS/W unified deep neural network accelerator with 1b-to-16b fully-variable weight bit-precision," in *Proc. IEEE ISSCC*, 2018, pp. 218–220.
[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," [Online]. Available: https://arxiv.org/abs/ 1409.1556
[15] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: a tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295-2329, Nov. 2017.
[16] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. CVPR*, 2016, pp. 770–778.
[17] F. Tu *et al*., "Deep convolutional neural network architecture with reconfigurable computation patterns," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., vol. 25, no. 8, pp. 2220–2233, Aug. 2017.