ICIC Express Letters
Part B: Applications
Volume **12**, Number **7**, July **2021**

ICIC International ©2021 ISSN 2185-2766
pp. **627–633**

IMPROVEMENT OF SHRINKING CNN ARCHITECTURE USING WEIGHT SHARING AND KNOWLEDGE DISTILLATION FOR TACTILE OBJECT RECOGNITION

Pornthep Sarakon[1,*], Hideaki Kawano[1], Kazuhiro Shimonomura[2] and Seiichi Serikawa[1]

[1]Department of Electrical and Electronic Engineering
Kyushu Institute of Technology
1-1 Sensui-cho, Tobata-ku, Kitakyushu-shi, Fukuoka 804-8550, Japan
[*]Corresponding author: sarakon.pornthep486@mail.kyutech.jp
kawano@ecs.kyutech.ac.jp; serikawa@elcs.kyutech.ac.jp

[2]Department of Robotics
College of Science and Engineering
Ritsumeikan University
1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan
skazu@fc.ritsumei.ac.jp

Received November 2020; accepted February 2021

Abstract. *An intelligent sensing system is mainly in assistive technology for supporting the elderly and people with disparities. A smart optical tactile sensor plays an important role in robot's sense of touch. As processing system, an embedded system has fixed resource budget and is unsuitable for modern convolutional neural networks. However, the performances of convolutional neural networks have improved as their structure has become more complicated. This means that they incur higher computational costs and become slower. We proposed a smart optical tactile sensor, called "tactile object recognition system" in previous work. This paper extends the previous work by improving learning ability, reducing save storage and analyzing to improve compression rate of save storage. The proposed method consists of three steps: (i) shrinking CNN architecture, (ii) knowledge transfer by knowledge distillation, and (iii) weight sharing by quantization and K-means clustering algorithm. The performance of this method compares with shrinking, knowledge distillation, quantization technique and soft filter pruning. The result shows that the proposed method is able to improve the compression rate of save storage, and it outperforms the other compression network techniques.*
Keywords: Image classification, Convolutional neural network, Network compression, Embedded systems, Tactile object recognition

1. **Introduction.** Nowadays, much effort has gone into developing assistive technology for supporting the elderly and people with disabilities in their lives, such as necessity, facilitation and safety. An intelligent sensing system plays an important role in the assistive technology to understand the surrounding environment. One of the intelligent sensing systems is a smart optical tactile sensor having its perception and manipulation. The smart optical tactile sensor is used in robot hand as sense of touch, and it can fulfill people without hands or replace people in dangerous jobs. The smart optical tactile sensor consists of two parts: (i) optical tactile sensor and (ii) processing system. Firstly, the optical tactile sensor makes tactile image sensors using camera and the most popular type is a marker displacement-based sensor [1], such as a random-dot optical tactile sensor [2], full-resolution optical tactile sensor [3] and soft optical tactile sensors [4]. The optical tactile sensors are used to recognize shape, texture and force. Secondly, the processing system is

DOI: 10.24507/icicelb.12.07.627

the smart optical tactile sensor's brain. As an embedded system is small and budget, it is the most popular processing system. The embedded system has a fixed resource budget being low memory and small processors, while modern recognition system becomes expensive computational cost. As deep learning has become a powerful tool for recognition, one of the modern recognition systems is convolutional neural network (CNN). A high performant CNN usually becomes complicated architecture, including a wider, deeper, and a larger number of residual connection than a low performant one. For this reason, they encounter higher computational cost and become slower.

As mentioned above, accurate CNN is unsuitable for the embedded system. Network compression (NC) overcomes the need for high computational cost [5-13]. The well-known NC includes low-rank factorization, quantization, pruning and knowledge distillation. Each method compresses a network in different ways. Methods based on low-rank factorization [5] can reduce save storage of network using matrix decomposition to reduce a number of parameters. The well-known low-rank factorization is singular value decomposition (SVD) which can significantly compress fully connected layers, while modern CNN replaces fully connected layers with global average pooling layer. The result of SVD has lower correctness than the original network. Convolutional layers can be compressed by adaptive mixture [6], and it solves the accuracy drop problem but gets higher multiplier-accumulators (Madds) than original one. Quantization-based methods [7,8] decrease the number of bits associated with each weight to increase memory bandwidth. This method accelerates the speed but offers lower accuracy. Binary, eight-bit and sixteen-bit quantization affect the accuracy and speed, i.e., (i) more bit CNN becomes, more accuracy CNN gets and (ii) more bit CNN becomes, slower CNN becomes. Pruning-based method takes advantage of the narrow network to reduce the number of parameters based redundant parameters or structures removal. Parameter pruning [9] relies on the support of hardware and a computing library; however, structure pruning [10] resolves the unfriendly hardware and Basic Linear Algebra Subprograms (BLAS library). The accuracy drop is commonly encountered in both pruning methods. Methods based on knowledge distillation [11,12] transfer knowledge from a teacher CNN (a pre-trained network) to a student CNN (a smaller network). The methods based on knowledge distillation employ narrow and shallow network with knowledge transfer, but require domain knowledge to create a high performant student network. When the student network is unsuitable, the performance of student network is worse than that of accuracy. As mentioned above, almost methods had a good compression rate although it could hurt in accuracy of the result network and a compact method hinder use. To solve these problems, we proposed a smart optical tactile sensor, called "tactile object recognition system", to bridge the gap between trade-off accuracy and the inference time (latency). We searched for the best ration of compound multipliers to achieve the best network performance.

Inspired by weight sharing, the present paper extends our previous work [2] by improving learning ability, reducing save storage and analysis. The improvement of the shrinking approach leads to a more-efficient save storage without accuracy drop. This approach consists of three steps: namely (i) network compression by shrinking CNN architecture, (ii) learning ability improvement by knowledge distillation, and (iii) weight sharing using quantization and K-means clustering algorithm.

The paper is organized as follows. The proposed method is described in Section 2. In Section 3, we evaluate the performance of this method and compare it with those of shrinking, knowledge distillation, quantization technique and soft filter pruning. Finally, Section 4 concludes.

2. Improvement of Shrinking CNN Architecture Using Weight Sharing and Knowledge Distillation Method.

The method consists of four aspects, namely, (i) problem formulation for a trade-off between accuracy and latency, (ii) shrinking CNN architecture, (iii) learning ability improvement by knowledge distillation, and (iv) weight sharing by quantization and K-means clustering algorithm.

2.1. **Problem formulation for trade-off between accuracy and latency.** In order to adapt to embedded systems, the final network is expected to be accurate and timely as called accuracy-latency trade-off task. This paper evaluates the performance of student networks with a trade-off between accuracy and latency. The latency is the inference time of the child network on NVIDIA Jetson TXII. Similar to [2], our objective function deals with two factors, namely (i) accuracy and (ii) latency on the NVIDIA Jetson TXII. Thus, a multi-objective function is defined as below:

$$maximize_m ACC(m) \times \left[\frac{LAT(m)}{TAR}\right]^w \tag{1}$$

where m is the student network, $ACC(m)$ is the accuracy of each student network on the target task, $LAT(m)$ is the latency on the target embedded system, $TAR = 16$ ms is target latency, and $w = -0.09$ is a hyper-parameter that controls the trade-off between accuracy and latency. In the experiments, the higher the objective score, the better the student network.

2.2. **Shrinking CNN architecture.** According to [2,14], the shrinking CNN architecture approach is a network-structure compression technique to compromise between accuracy and latency. The shrinking approach can reduce the number of parameters, the network size, and the computational cost to work on embedded systems, because it has effect on three dimensions (e.g., the width, resolution, and depth) of the networks where the redundant architecture is removed. Figure 1 shows the baseline network (a large network) and the shrunk network (a small network). With the shrinking technique, the computational cost of CNN at each block is described by

$$cost = (\beta D_K)^2(\beta D_F)^2\left(\alpha^2 N\right)(M + (\gamma L - 1)(N)) \tag{2}$$

where α is the width multiplier, β is the resolution multiplier, γ is the depth multiplier, D_K is the kernel size, M is the number of inputs, N is the number of outputs, L is the number of layers, and D_F is the feature-map size. With balanced multipliers and a

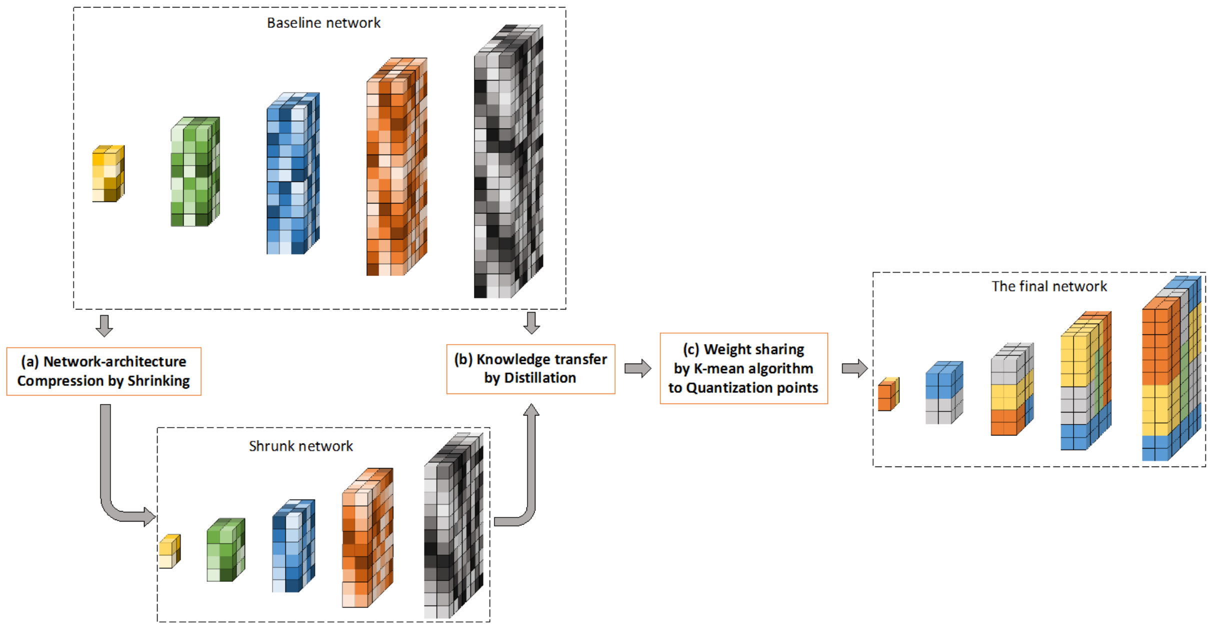
Baseline network
(a) Network-architecture Compression by Shrinking
(b) Knowledge transfer by Distillation
(c) Weight sharing by K-mean algorithm to Quantization points
The final network
Shrunk network

FIGURE 1. Flowchart of improvement of shrinking CNN architecture using weight sharing and knowledge distillation approach

shallow network, shrunk networks provide timely performance with lossless. For example, the reduced computational cost of VGG16's third block is 8 times less computational cost than the original one as described below:

$$\frac{5.76 \times 10^9}{4.62 \times 10^{10}} = 1.25 \times 10^{-1}$$

The shrinking multipliers are $\alpha = 0.75$, $\beta = 0.75$, and $\gamma = 0.75$. The parameters on the CNN are $D_K = 3$, $M = 128$, $N = 256$, $L = 3$, and $D_F = 56$. In case of Inverted Bottleneck Conv network [15], its architecture is more complex than CNN and the computational cost is described by

$$cost = \alpha E(\beta D_F)^2 \left(M\left(\alpha M + \alpha N + \beta D_K^2\right) + (\gamma L - 1)\left(N\left(2\alpha N + \beta D_K^2\right)\right)\right) \tag{3}$$

For example, the reduced computational cost of Inverted Bottleneck Conv is 38 times less computational cost than the original one as described below:

$$\frac{3.77 \times 10^6}{1.45 \times 10^9} = 2.60 \times 10^{-2}$$

where E is the expansion ratio. The shrinking multipliers are $\alpha = 0.5$, $\beta = 0.5$, and $\gamma = 0.5$, and the parameters on the Inverted Bottleneck Conv are $E = 6$, $D_K = 3$, $M = 40$, $N = 64$, $L = 4$, and $D_F = 28$.

2.3. **Learning ability improvement by knowledge distillation.** Knowledge distillation [11] transfers the knowledge of a large and computationally expensive network (a teacher network) to a computationally efficient network (a student network). In this paper, the teacher and student networks are a baseline and a shrunk network, respectively. By joint training with the teacher network, the student network is guided by the transferred knowledge, called a logit distribution. For this reason, knowledge distillation can improve the learning ability of the student network, such as no accuracy drop and short training time. In the training of the student network, softmax is used to generate the classification probability [16] q_i of the i-th category, as described by (4)

$$q_i = \frac{\exp(z_i/T)}{\sum_{j=1}^{K} \exp(z_j/T)} \tag{4}$$

where i represents the i-th category, z_i is the output of the logits layer of the network, $\sum_{j=1}^{K} \exp(z_j/T)$ is the normalization term, K is number of categories in the multi-class classifier, and $T = 5$ is introduced to produce a softer probability distribution over classes [11]. Loss function ($\mathcal{L}$) is described as follows:

$$\mathcal{L} = \alpha CE(Q_S, y) + (1 - \alpha) KL\left(Q_S^t, Q_T^t\right) \tag{5}$$

where $\alpha = 0.9$ is a hyperparameter controlling the compromise between the two losses, CE is the cross-entropy loss, KL is Kullback-Leibler (KL) divergence loss, Q_S^t and Q_T^t are the soft target of the student and teacher network, Q_S is the predicted output, and y is the actual label.

2.4. **Weight sharing.** Embedded systems have a limited resource budget, which is a fixed capability. Weight sharing reduces the requirement for weight storage of the final network to be suitable for the embedded systems. Similar to [9], K-means clustering algorithm is used to realize the weight value clustering, and adopt the centroids as quantization points for a cluster. Compression rate is described by (6)

$$r = \frac{nb}{n \log_2(k) + kb} \approx \frac{b}{\log_2(k)} \tag{6}$$

where k is the number of weight clusters and it needs $\log_2(k)$ bits to encode the index. n is network-connection ($n \gg k$) and each connection is represented by b bits. The quantization points are 4 times lighter than the original weights when 8 bits are applied.

3. **Experimental Result.** We used the tactile image dataset [2] to experiment and were training on four NVIDIA GTX 1080Ti GPUs. A tactile image was captured while the surface of the experimental object was in contact with the opaque layer of the random-dot sensor. Displacing the opaque layer changed the positions of the dots in the transparent layer. Examples of dataset were shown as Figure 2. Random seed was fixed as 42. Training and testing parts were randomly separated by holdout technique with a ratio of 60 : 40. The training part includes training and validating datasets, which were randomly separated by three-folds cross-validation. The batch size was 128 images and was shuffled every epoch. The initial weights of the experimental networks were set by Kaiming technique [17]. The latency measures on NVIDIA Jetson TXII with GPU. We compare the performance of the networks using the objective function, mentioned in Section 2.1 and a higher score means better performance. For a fair comparison, we selected VGG16 and MobileNetV2, the CNN employed in the previous version of this paper, as a pre-trained network. The learning rate is set to 1×10^{-3} for VGG16 [18] and to 2.56×10^{-2} for MobileNetV2 [15]. The shrinking multipliers, referring to [2], are set to 0.25 : 0.75 : 0.5 (width : resolution : depth multipliers) for VGG16 and to 0.5 : 0.5 : 0.25 (width : resolution : depth multipliers) for MobileNetV2. Hyperparameters are described as follows: image size = 224 × 224, number of epochs = 50, optimizer = Adam, loss function = cross-entropy, bits = $\{8, 32\}$ and distillation = {True, False}.

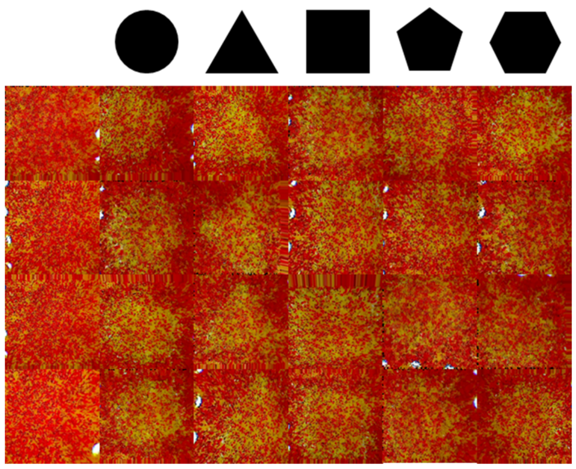

FIGURE 2. Examples of augmented tactile images: (from left to right) default, circle, triangle, square, pentagon, hexagon

3.1. **Performance of learning ability.** Figure 3 illustrates the effectiveness of knowledge distillation of 8-bit shrunk MobileNetV2 compared with normal training (no distillation). The accuracy curves of 8-bits shrunk MobileNetV2 with two kinds of training methods describe that knowledge distillation method is able to reach 4.65% in accuracy from normal training as shown zoomed in on Figure 3. This means that the distillation is able to prevent accuracy drop although bits are reduced. This article shows the accuracy of 8-bit shrunk MobileNetV2 is 95.70%, while 32-bit shrunk MobileNetV2 without distillation has 95.69% as shown in Table 1.

3.2. **Weight sharing sensitivity analysis of shrunk networks.** Table 1 shows that the weight sharing effects on the accuracy of the network, i.e., the higher number of clusters the weight sharing is set, the higher accuracy the network gets. For example, 8-bit shrunk

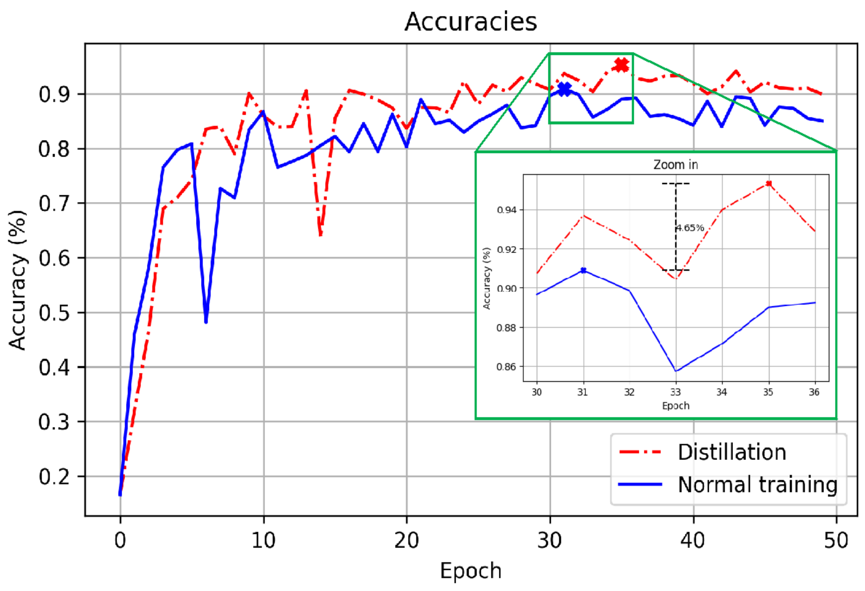

Figure 3. Accuracy comparison of 8-bits shrunk MobileNetV2 with distillation and normal training. Solid line shows accuracy curves of normal training. Dash-dot line shows accuracy curves of distillation. Cross marker shows the highest accuracy of each training.

Table 1. Compression performance of different methods (S: Shrinking, WS: Sharing weights, KD: Distillation, Q: Quantization, SFP: Soft filter pruning, +: Strategy combination)

Model	Method	Params. (M)	Madds. (M)	Network size (M)	Acc. (%)	Latency (ms)	Score
VGG16	Baseline	134	15,500	512	85.76	106.04	72.32
	Q [7]	134	15,500	128	82.84	64.57	73.06
	SFP [10]	134	7,653	512	84.50	86.21	72.52
	KD [11]	43	645	164	84.79	41.95	77.74
	WS + KD [12]	43	645	41	83.72	41.95	76.76
	S [2]	43	378	164	93.51	16.16	93.43
	S + WS	43	378	41	89.33	16.16	89.25
	S + WS + KD	43	378	41	93.54	16.16	**93.46**
MobileNetV2	Baseline	2.24	300	8.51	94.13	26.48	89.96
	Q [7]	2.24	300	2.13	85.65	12.63	87.49
	SFP [10]	2.24	147	8.51	92.89	21.62	90.41
	KD [11]	0.30	44	1.15	93.82	12.92	95.64
	WS + KD [12]	0.30	44	0.29	91.74	12.92	93.52
	S [2]	0.30	12	1.15	95.69	12.07	98.15
	S + WS	0.30	12	0.29	91.04	12.07	93.38
	S + WS + KD	0.30	12	0.29	95.70	12.07	**98.16**

MobileNetV2 has accuracy approximately 91.04%, which is 4.65% less accurate than 32-bit one (shrinking method). On the other hand, the save storage of network reduces when the weight sharing is set as the low number of clusters. 8-bit shrunk MobileNetV2 has save storage approximately 0.29 M, being 3.98× smaller than 32-bit one.

3.3. **Overall performance analysis.** Table 1 shows the performance of the overall experimental result in this paper with different strategies. The highest score is performed by the shrunk MobileNetV2 with weight sharing and knowledge transfer; i.e., it has 95.70%

accuracy with 0.29 M network size, and it is 0.01-1.39% more accurate and 3.97-29.34× smaller than those in the baseline and shrunk network. Moreover, Table 1 shows a performance comparison between the proposed method and other network compression methods. The proposed method shows the highest score and smallest network size and it outperforms the other network compression techniques.

4. **Conclusions.** This paper proposed the improvement of shrinking CNN architecture using weights sharing and knowledge transfer by knowledge distillation. The main motivation behind the proposed method was the need to develop the assistive technology for the elderly and people with disabilities. The result shows that the save storage of network decreases and trade-off between accuracy and latency is improved by combined techniques; e.g., shrinking CNN architecture, weight sharing and knowledge transfer by knowledge distillation. The proposed method outperforms when compared with those of other network compression techniques. We established that the improvement of the shrinking approach is able to reduce save storage without performance drop. In future work, the performance of this method should be improved.

REFERENCES

[1] K. Shimonomura, Tactile image sensors employing camera: A review, *Sensors*, vol.19, no.18, 2019.

[2] P. Sarakon, H. Kawano, K. Shimonomura and S. Serikawa, Efficient and small network using multi-trim network structure for tactile object recognition on embedded systems, *IEEE Access*, vol.8, pp.144277-144291, 2020.

[3] C. Sferrazza and R. D'Andrea, Design, motivation and evaluation of a full-resolution optical tactile sensor, *Sensors*, vol.19, no.4, 2019.

[4] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter and N. F. Lepora, The TacTip family: Soft optical tactile sensors with 3D-printed biomimetic morphologies, *Soft Robotics*, vol.5, no.2, pp.216-227, 2018.

[5] X. Yu, T. Liu, X. Wang and D. Tao, On compressing deep models by low rank and sparse decomposition, *Proc. of IEEE Comput. Vis. Pattern Recognit. (CVPR)*, pp.67-76, 2017.

[6] T. Chen, J. Lin, T. Lin, S. Han, C. Wang and D. Zhou, Adaptive mixture of low-rank factorizations for compact neural modeling, *Proc. of Neural Inf. Process. Syst. (NIPS)*, pp.1-4, 2018.

[7] R. Krishnamoorthi, Quantizing deep convolutional networks for efficient inference: A whitepaper, *arXiv: 1806.08342*, http://arxiv.org/abs/1806.08342, 2018.

[8] M. Courbariaux, Y. Bengio and J. P. David, BinaryConnect: Training deep neural networks with binary weights during propagations, *Proc. of Neural Inf. Process. Syst. (NIPS)*, pp.3123-3131, 2015.

[9] S. Han, J. Pool, J. Tran and W. Dally, Learning both weights and connections for efficient neural network, *Proc. of Neural Inf. Process. Syst. (NIPS)*, pp.1135-1143, 2015.

[10] Y. He, G. Kang, X. Dong, Y. Fu and Y. Yang, Soft filter pruning for accelerating deep convolutional neural networks, *Proc. of the 27th Int. Joint Conf. Artif. Intell.*, pp.2234-2240, 2018.

[11] G. Hinton, O. Vinyals and J. Dean, Distilling the knowledge in a neural network, *NIPS 2014 Deep Learning Workshop*, 2015.

[12] A. Polino, R. Pascanu and D. Alistarh, Model compression via distillation and quantization, *Proc. of Int. Conf. Learn. Represent. (ICLR)*, pp.1-21, 2018.

[13] A. S. Agoes, Z. Hu and N. Matsunaga, LICODS: A CNN based, lightweight RGB-D semantic segmentation for outdoor scenes, *International Journal of Innovative Computing, Information and Control*, vol.15, no.5, pp.1935-1946, 2019.

[14] P. Sarakon, H. Kawano, K. Shimonomura and S. Serikawa, Shrinking model and random-dot sensor on embedded systems, *ICIC Express Letters*, vol.14, no.7, pp.703-710, 2020.

[15] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, *Proc. of IEEE Comput. Vis. Pattern Recognit. (CVPR)*, pp.4510-4520, 2018.

[16] C. Guo, G. Pleiss, Y. Sun and K. Q. Weinberger, On calibration of modern neural networks, *Proc. of Int. Conf. on Machine Learning (ICML)*, vol.70, pp.1321-1330, 2017.

[17] K. He, X. Zhang, S. Ren and J. Sun, Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification, *Proc. of IEEE Int. Conf. Comput. Vis. (ICCV)*, 2015.

[18] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *Proc. of Int. Conf. Learn. Represent. (ICLR)*, 2014.