Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5573031, 7 pages
https://doi.org/10.1155/2021/5573031

Research Article

Integrated Image Sensor and Light Convolutional Neural Network for Image Classification

Cheng-Jian Lin,[1,2] **Chun-Hui Lin**,[3] **and Shyh-Hau Wang**[3,4]

[1]*Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan*
[2]*College of Intelligence, National Taichung University of Science and Technology, Taichung 404, Taiwan*
[3]*Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan*
[4]*Intelligent Manufacturing Research Center, National Cheng Kung University, Tainan 701, Taiwan*

Correspondence should be addressed to Cheng-Jian Lin; cjlin@ncut.edu.tw

Received 27 January 2021; Revised 1 March 2021; Accepted 10 March 2021; Published 17 March 2021

Academic Editor: Teen-Hang Meen

Deep learning has accomplished huge success in computer vision applications such as self-driving vehicles, facial recognition, and controlling robots. A growing need for deploying systems on resource-limited or resource-constrained environments such as smart cameras, autonomous vehicles, robots, smartphones, and smart wearable devices drives one of the current mainstream developments of convolutional neural networks: reducing model complexity but maintaining fine accuracy. In this study, the proposed efficient light convolutional neural network (ELNet) comprises three convolutional modules which perform ELNet using fewer computations, which is able to be implemented in resource-constrained hardware equipment. The classification task using CIFAR-10 and CIFAR-100 datasets was used to verify the model performance. According to the experimental results, ELNet reached 92.3% and 69%, respectively, in CIFAR-10 and CIFAR-100 datasets; moreover, ELNet effectively lowered the computational complexity and parameters required in comparison with other CNN architectures.

1. Introduction

Convolutional neural network (CNN) was firstly introduced in the 1980s. At that time, Lecun et al. [1] proposed a simply constructed CNN architecture which contains three convolutional layers, two subsampling layers, and a fully connected layer. LeNet was mainly used for handwriting recognition in the MNIST dataset and obtained the lowest error rate. However, the hardware equipment was not advanced, and graphics processing units had not been invented which led to the development of CNN being greatly restricted. In 2012, Krizhevsky et al. [2] developed AlexNet and won the first place in the ImageNet large-scale visual recognition competition by achieving a top-5 error of 15.3%. Compared with LeNet, AlexNet uses rectified linear unit (ReLU) to replace the conventional sigmoid activation function in order to resolve the vanishing gradient problem. Moreover, the dropout [3] regularization technique was also introduced to reduce overfitting in neural networks. In general, AlexNet extends its network architecture resulting in the requirement of nearly 60 million parameters, and the floating-point operations (FLOPs) have reached 0.7 giga FLOPs. Subsequently, researchers have continued to deepen networks to improve the accuracy such as VGGNet [4].

Instead of deepening the CNN architecture, some researchers expand the width of the network architectures. For instance, Szegedy et al. [5] firstly came up with a concept of inception block in the CNN which encapsulates different sizes of kernels for extracting global and local features. It adjusts the computations by adding a bottleneck layer of a 1×1 convolutional filter before applying large-size kernels. Furthermore, Srivastava et al. [6] designed a new architecture to moderate gradient-based training of very deep networks which is called highway network. This network imitates the horizontal expansion concept using the gating function to adaptively bypass the input so that the network can go deeper. In addition, He et al. [7] proposed ResNet by taking inspiration from the bypass and bottleneck layer approaches for reducing the amount of operations. Many improved designs of network architectures are proposed and applied in many applications, such as object detection [8] and semantic analysis [9]. However, regardless of deepening or widening the network architectures, high computational cost and memory requirement are the two main concerns observed with these architectures.

To further alleviate these two primary concerns of the network, designing a lightweight architecture without compromising the performance is necessary, especially when the CNN model is implemented in resource-constrained hardware. Howard et al. [10] adopted depthwise separable convolution in the MobileNet to reduce the model parameters so that the model can be embedded in portable devices for mobile and embedded vision applications. Juefei-Xu et al. [11] proposed the local binary convolutional neural network which adopts local binary convolution (LBC) as a substitute for the conventional CNN. The experimental results showed that the LBC module performs a good approximation of a conventional convolutional layer and results in a major reduction in the number of learnable parameters while training the network. Iandola et al. introduced SqueezeNet [12] which replaces 3×3 filters with 1×1 filters and decreases the number of input channels to 3×3 filters. These strategies are desirable to decrease the quantity of parameters in a CNN while attempting to maintain accuracy. According to the experimental results reported by Iandola et al., the parameters used in SqueezeNet are 50x fewer than those in AlexNet; besides, it preserves AlexNet-level accuracy on ImageNet. Others such as parameter pruning and quantization can reduce redundant parameters which reduces the network complexity and addresses the overfitting problem. Furthermore, without decreasing accuracy, more improvements of YOLO were also proposed [13, 14] to prove that light CNN can reduce training time and make applications more diverse without being limited by hardware.

The three modules provide capabilities and advantages: saving computations when kernel size and the number of kernels are large using depthwise separable convolution, expanding the field of view (FOV) of filters without increasing parameters by atrous convolution, and extracting local and global features simultaneously adopted by the inception module to reduce the parameters and operations of the CNN. In this study, the proposed model, efficient light convolutional neural network (ELNet) with the three modules, is no longer limited by memory and computational constraints.

The rest of the paper is organized as follows. In Section 2, the conventional CNN architecture is briefly reviewed. The ELNet is introduced in Section 3. The experimental results using CIFAR-10 and CIFAR-100 datasets are revealed in Section 4 and compared with other state-of-the-art CNN architectures such as GoogLeNet, ResNet-50, and MobileNet. Lastly, Section 5 draws conclusions.

2. Convolutional Neural Network (CNN)

The concept of neural networks mainly comes from biological neural network systems; however, neural networks are connected in a fully connected manner which causes a great amount of calculations when the input size is large. Therefore, in the 1980s, convolution kernel was first introduced and then was widely applied in image processing. There are four main parts of the CNN: convolutional layer, pooling layer, activation function, and fully connected layer. The function of feature extraction depends on the first three parts, and the fully connected layer is used to classify the obtained features. More descriptions of these parts are explained as follows.

2.1. Convolutional Layer. A convolutional layer consists of a set of learnable filters (or kernels) which have a small receptive field; however, feature extraction can be acquired by extending filters through the full depth of the input volume. The formula is as follows:

$$O_{r,c} = \sum_{k=1}^{n} \sum_{i=1}^{k_h} \sum_{j=1}^{k_w} I^{k}_{r-k_h+i,c-k_w+j} \times W^{k}_{i,j} + b, \qquad (1)$$

where r and c represent the row and column of the feature map, n is the number of input channels, k_w and k_h are the width and height of a convolution kernel, $W^k_{i,j}$ is the weight of the ith row and jth column convolution kernel in the kth channel, $I^k_{i,j}$ is the input of the ith row and jth column in the kth channel, and b is the bias.

2.2. Pooling Layer. In order to effectively extract features, most of the moving strides are set as 1; yet, this setting causes relatively more operations. Therefore, pooling layer is usually added in the CNN for effectively reducing the amount of operations. Equation (2) shows the calculation of max pooling and average pooling:

$$O_{r,c} = \begin{cases} \text{max pooling}\left(\max\left(I_{i,j}\right) \Big| \begin{array}{r} r \le i < r + P_h \\ \\ c \le j < c + P_w \end{array} \right), \\ \\ \text{average pooling}\left(\displaystyle\sum_{i=r}^{r+P_h} \sum_{j=c}^{c+P_w} \frac{I_{i,j}}{\left(P_w \times P_h\right)} \right), \end{cases} \qquad (2)$$

where $O_{r,c}$ is the output row and column, $I_{i,j}$ is the row and column of the input image, and P_w and P_h are the width and height of the pooling kernel.

2.3. Activation Function. The conventional operation of the convolution kernel is a linear operation; LeNet adopts sigmoid function as an activation function to solve nonlinear problems. Along with the development of deeper network, researchers found out that gradient disappearance occurs when the sigmoid function approaches to 0 in the saturation region. Then, ReLU is introduced in AlexNet to address this problem. Moreover, the operations using ReLU are simpler than those of the sigmoid function. Later, many scholars made various improvements based on ReLU and sigmoid functions. For instance, Leaky ReLU [15] solves the problem that ReLU is not activated when x is less than 0, PReLU [16] adds a parameter to make ReLU more accurate when x is less than 0, and RReLU [17] learns parameters automatically via the neural network. Here, PReLU is selected as the activation function which is shown in Figure 1, and its equation is given as follows:

$$f_\beta(x) = \begin{cases} \beta \cdot x, & x \leq 0, \\ x, & x > 0. \end{cases} \quad (3)$$

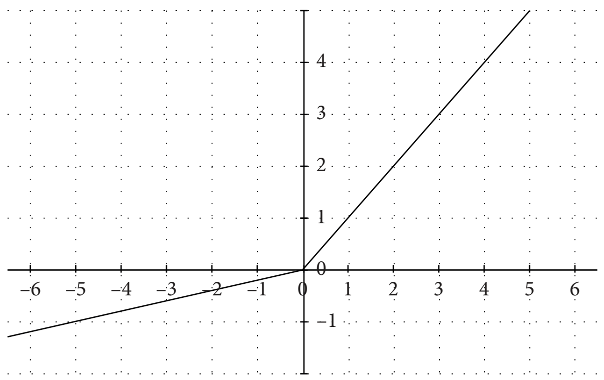

Figure 1: PReLU.

2.4. Fully Connected Layer. After convolutional computation, the high-dimensional feature maps will be classified and predicted through a fully connected neural network. This layer is often used in many network architectures such as LeNet, AlexNet, and GoogLeNet. The equation is given as follows:

$$P = O_c \times (I_c + 1), \quad (4)$$

where I_c and O_c represent the number of input and output channels.

From equation (4), the number of parameters in the fully connected layer depends on the input dimensions. If dimension reduction is not performed, the number of input channels might be massive, and many parameters will be generated. According to Lin et al. [18], the fully connected layer is prone to overfitting which hampers the generalization ability of the overall network. Therefore, the later CNN architectures usually replace fully connected layers with global average pooling.

3. Efficient Light Convolutional Neural Network (ELNet)

An efficient light convolutional neural network (ELNet) is proposed to make the network architecture suitable for resource-constrained hardware. A schematic view of the network is depicted in Figure 2, where the red block is a depthwise separable convolution, the black dash line block represents an inception module, and the brown block is a depthwise separable convolution combining with atrous convolution. The details of the architecture are described in Table 1.

In Table 1, Conv dw represents a depthwise separable convolution, and d means stride in the atrous convolution. The three convolutional modules used in ELNet are described as follows.

3.1. Depthwise Separable Convolution. Depthwise separable convolution separates the original convolution into two parts for the purpose of reducing operations as shown in Figure 3.

Compared with the conventional convolution method, one convolutional kernel will generate only one feature map according to its input dimensions. However, depthwise separable convolution performs multiple feature maps corresponding to each dimension, and then a 1×1 convolutional layer is used to combine all the feature maps into one output. Although there is no difference between the output of the depthwise separable convolution and conventional convolution, the parameters of the depthwise separable convolution using one 3×3 convolutional kernel are much less than those of the conventional convolution method. The calculations are listed as follows:

$$\begin{aligned} &\text{conventional convolution}: I_w \times I_h \times I_c \times k_w \times k_h \times k_c, \\ &\text{depthwise separable convolution}: I_w \times I_h \times I_c \times k_w \times k_h + I_c \times k_c \times I_w \times I_h, \end{aligned} \quad (5)$$

where I_w, I_h, and I_c represent the width, height, and channel of the input, respectively, k_w and k_h are the width and height of the convolutional kernel, and k_c is the number of convolutional kernels in the convolutional layer.

3.2. Atrous Convolution. Atrous convolution [9], as shown in Figure 4, enlarges the FOV of filters by incorporating the larger context without growing parameters. The advantages of using atrous convolution are allowing the user to filter a larger context instead of using a bigger size of kernel and reducing the usage of pooling layers which brings less operation consumption and accuracy improvement; besides, using less parameters can also avoid an overfitting problem.

3.3. Inception Module. Inception module uses various convolution kernels to extract features so that the feature maps are able to contain local features and global features. The schematic view of conventional convolutional layers and inception module are displayed in Figure 5 as comparison. Although both of the methods can map to the same size of FOV, local features in Figure 5(a) might be washed out at the end. On the contrary, the wash-out problem will not be

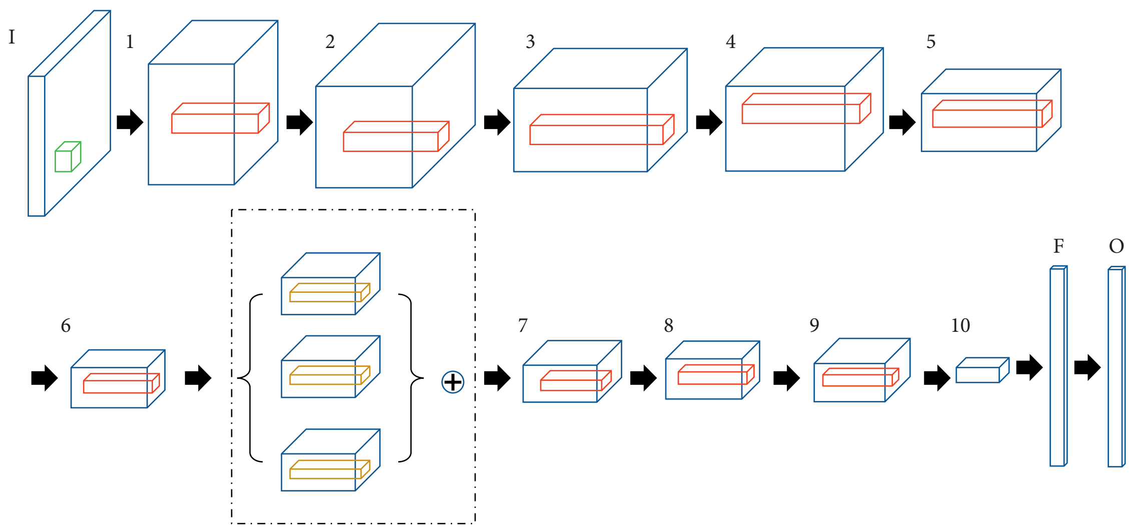

Figure 2: ELNet architecture.

Table 1: ELNet architecture.

	Type/stride	Filter shape $(H \times W \times C \times N, d)$	Input size
I	Conv/2	$3 \times 3 \times 3 \times 32$	$W \times H \times 3$
1	Conv dw/1	$3 \times 3 \times 32$	$W/2 \times H/2 \times 32$
	Conv/1	$1 \times 1 \times 32 \times 64$	$W/2 \times H/2 \times 32$
2	Conv dw/2	$3 \times 3 \times 64$	$W/2 \times H/2 \times 64$
	Conv/1	$1 \times 1 \times 64 \times 128$	$W/4 \times H/4 \times 64$
3	Conv dw/1	$3 \times 3 \times 128$	$W/4 \times H/4 \times 128$
	Conv/1	$1 \times 1 \times 128 \times 128$	$W/4 \times H/4 \times 128$
4	Conv dw/2	$3 \times 3 \times 128$	$W/4 \times H/4 \times 256$
	Conv/1	$1 \times 1 \times 128 \times 256$	$W/8 \times H/8 \times 256$
5	Conv dw/1	$3 \times 3 \times 256$	$W/8 \times H/8 \times 256$
	Conv/1	$1 \times 1 \times 256 \times 256$	$W/8 \times H/8 \times 512$
6	Conv dw/2	$3 \times 3 \times 256$	$W/8 \times H/8 \times 256$
	Conv/1	$1 \times 1 \times 256 \times 512$	$W/16 \times H/16 \times 512$
	(a) Atrous dw/1	$3 \times 3 \times 512 \quad d = 1$	
	(b) Atrous dw/1	$3 \times 3 \times 512 \quad d = 2$	$W/16 \times H/16 \times 512$
	(c) Atrous dw/1	$3 \times 3 \times 512 \quad d = 3$	
	Add	$(a) + (b) + (c)$	$W/16 \times H/16 \times 512$
7	Conv/1	$1 \times 1 \times 512 \times 512$	$W/16 \times H/16 \times 512$
8	Conv dw/2	$3 \times 3 \times 512$	$W/16 \times H/16 \times 512$
	Conv/1	$1 \times 1 \times 512 \times 1024$	$W/32 \times H/32 \times 512$
9	Conv dw/1	$3 \times 3 \times 1024$	$W/32 \times H/32 \times 1024$
	Conv/1	$1 \times 1 \times 1024 \times 1024$	$W/32 \times H/32 \times 1024$
10	Average pooling	Global pooling	$W/32 \times H/32 \times 1024$
F	Fully connected	$1024 \times$ Classes	1×1024
O	Softmax	Classification answer	$1 \times$ Class Numbers

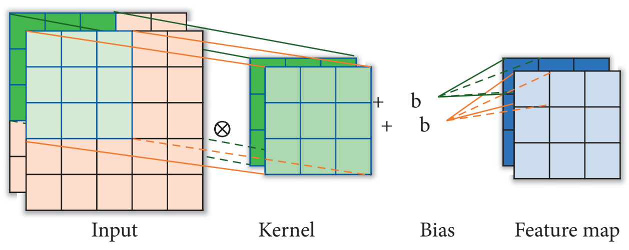

Figure 3: Depthwise separable convolution.

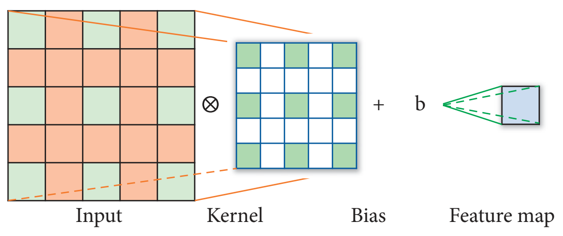

Figure 4: Atrous convolution.

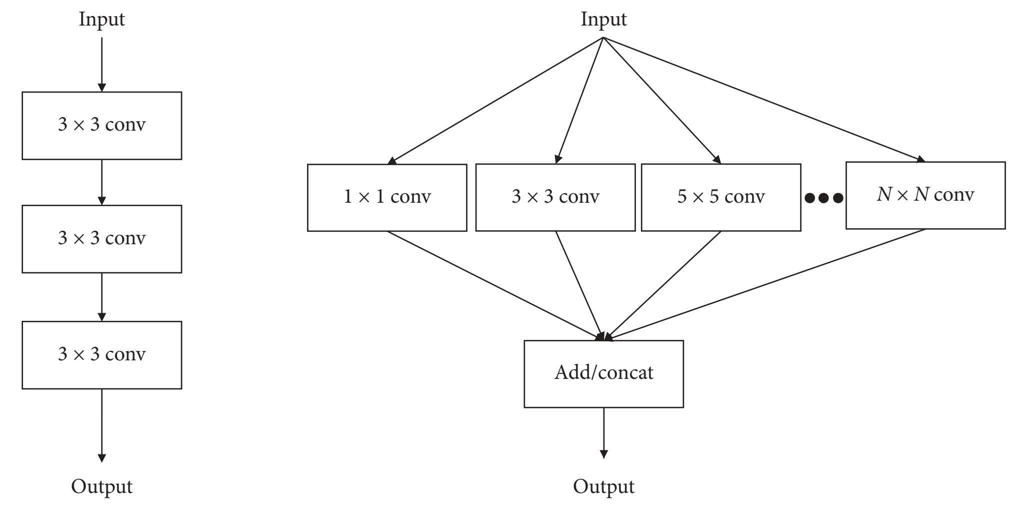

Figure 5: (a) Conventional convolutional layers. (b) Inception module.

considered when using the inception module (Figure 5(b)); however, fusing multiple feature maps is another question. In general, concatenation (Concat) and addition (Add) are two common methods; the former can retain characteristics of each convolution output but produce high-dimensional problems; in contrast, the latter does not have dimensional problems, yet relatively might lose the independence of each output.

4. Results and Discussion

To deploy the systems on resource-constrained hardware for real-time data processing, large-scale datasets such as PASCAL VOC, ImageNet, and COCO are not considered. Thus, CIFAR-10 and CIFAR-100, two well-understood and widely used datasets, were provided to verify the performance of ELNet. The experimental results including parameters, FLOPs, and accuracy were compared, respectively, with the other state-of-the-art CNN architectures such as GoogLeNet [5], ResNet-50 [7], MobileNet [10], and All Convolutional Net (All-CNN-C) [19]. The hardware specifications and predefined parameters used in this study are listed in Tables 2 and 3.

Table 2: Hardware specifications.

Hardware	Specification
GPU	NVidia GTX1080-Ti 11G
CPU	Intel Xeon E3-1225 v3 @ 3.2 GHz

Table 3: Predefined parameters.

Parameter	Value
Epoch	120
Optimizer	Nesterov's accelerated gradient
Learning rate	0.01
Learning rate decay	0.9
Learning rate decay frequency	40 (epochs/time)
Momentum	0.9
Batch size	100

4.1. CIFAR-10 Dataset. The CIFAR-10 dataset includes 60,000 colour images with the size of 32×32 in a total of 10 classes. To fit into the proposed network, bilinear interpolation is used to resize the images into 224×224 which provides more features than using the padding method. Table 4 shows the results in which the parameters and MFLOPs required in larger CNN models such as GoogLeNet and ResNet-50 models are very large. In other words, these models need longer training time and higher operations. To make the model suitable for general hardware equipment, models with less operations and lower complexity are more favourable. Therefore, the proposed model is also compared with MobileNet and All-CNN-C which are also called light models. According to the results, MobileNet uses less parameters and MFLOPs than others; yet, the accuracy is lower than that of ELNet. Even though All-CNN-C has the least parameter requirements, its MFLOPs are the highest which means the training time could be decreased by using better graphics processing units, but this increases the cost of hardware equipment. ELNet reaches a tradeoff between accuracy and parameters/MFLOPs which is closer to the purpose of this study than that of other methods.

4.2. CIFAR-100 Dataset. The CIFAR-100 dataset contains 100 classes which are more than in the CIFAR-10 dataset. Therefore, the accuracy shown in Table 5 is obviously relatively lower than the accuracy of classifying the CIFAR-10 dataset; yet, the accuracy of ELNet is still the highest.

To evaluate the effectiveness of three convolutional modules used in ELNet, Tables 6 and 7 show the results of classifying the CIFAR-100 dataset. Table 6 shows that using atrous convolution can not only widen the FOV which increases the accuracy from 67% to 69% but also reach the same accuracy (69%) as using a bigger kernel size. Additionally, the inception module has the ability to extract

Table 4: Experimental results using the CIFAR-10 dataset.

Model	Parameter (10^6)	MFLOPs	Accuracy (%)
GoogLeNet [5]	6.9	1,582	83.1
ResNet-50 [7]	25.6	3,857	88.1
MobileNet [10]	4.2	569	85.6
All-CNN-C [19]	1.37	13,965	90.9
ELNet (proposed)	2.1	257	92.3

Table 5: Experimental results using the CIFAR-100 dataset.

Model	Accuracy (%)
GoogLeNet [5]	56
ResNet-50 [7]	57.3
MobileNet [10]	65
All-CNN-C [19]	66.3
ELNet (proposed)	69

Table 6: The comparisons of using the atrous convolution.

Model	Parameter (10^6)	MFLOPs	Accuracy (%)
ELNet (no atrous convolution)	2.1	257	67
ELNet (7×7 kernel size)	2.2	262	69
ELNet	2.1	257	69

Table 7: The comparisons of using the inception module.

Model	Parameter (10^6)	MFLOPs	Accuracy (%)
ELNet (Add)	2.1	257	69
ELNet (Concat)	2.6	359	69.4
ELNet (no inception module)	2.1	257	68

features using different convolution kernel sizes. In order to keep the features, different fusion methods may display distinct results. From the experimental results (Table 7), concatenation shows better accuracy than the other two methods; however, it requires more parameters and MFLOPs; thus, the addition method might be the better choice for implementing the network in a resource-constrained environment.

Overall, the proposed ELNet showed better performance in comparison with either relatively larger CNN architectures (GoogLeNet and ResNet-50) or light CNN architectures (MobileNet and All-CNN-C). The accuracy of ELNet is acceptable if the environment of the deployed system is considered. Although the proposed ELNet reaches 92.3% and 69% in the CIFAR-10 and CIFAR-100 datasets, respectively, the accuracy can be improved by using more complex networks. The three convolution modules with depthwise separable convolution, atrous convolution, and inception modules can also be extended to these complex networks to lower the number of parameters and operations and preserve the accuracy of classification as well.

5. Conclusions

The contributions of this study listed in the following confirm that the ELNet can effectively reduce model complexity but maintain fine accuracy:

(1) ELNet successfully combines three convolutional modules, depthwise separable convolution, atrous convolution, and inception module, for reducing the number of parameters and operations in the model

(2) ELNet requires only 2.1 million training parameters and 2.57 mega FLOPs based on the input image size that is equal to 224×224

(3) The accuracy of ELNet reached 92.3% and 69% in CIFAR-10 and CIFAR-100 datasets, respectively

Therefore, the proposed ELNet can be applied on embedded systems for image classification applications. In addition, the architecture can integrate other methods such as parameter pruning, recursion, or other learning methodologies to optimize the network for further research.

Data Availability

The CIFAR-10 and CIFAR-100 datasets are available to access from https://www.cs.toronto.edu/~kriz/cifar.html.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank the support of the Intelligent Manufacturing Research Center (iMRC) from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan. This research was funded by the Ministry of Science and Technology of the Republic of China (Grant no. MOST 109-2221-E-167-027).

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the International Conference on Neural Information Processing Systems*, Lake Tahoe, Nevada, USA, December 2012.

[3] N. Srivastava, G. Hinton, A. Krizhevsky et al., "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, vol. 15, pp. 1929–1958, 2014.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[5] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, June 2015.

[6] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, http://arxiv.org/abs/1505.00387.

[7] K. He, X. Zhang, S. Ren et al., "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.

[8] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot MultiBox detector," in *Proceedings of the European Conference on Computer Vision*, Amsterdam, The Netherlands, October 2016.

[9] L. C. Chen, G. Papandreou, I. Kokkinos et al., "DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 834–848, 2017.

[10] A. G. Howard, M. Zhu, B. Chen et al., "MobileNets: efficient convolutional neural networks for mobile vision applications," 2017, http://arxiv.org/abs/1704.04861.

[11] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Local binary convolutional neural networks," 2017, http://arxiv.org/abs/1608.06049.

[12] F. N. Iandola, S. Han, M. W. Moskewicz et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, http://arxiv.org/abs/1602.07360.

[13] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 2017.

[14] J. Redmon and A. Farhadi, "YOLOv3: an incremental improvement," 2018, http://arxiv.org/abs/1804.02767.

[15] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the International Conference on Machine Learning*, Atlanta, GA, USA, June 2013.

[16] K. He, X. Zhang, S. Ren et al., "Delving deep into rectifiers: surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, December 2015.

[17] B. Xu, N. Wang, T. Chen et al., "Empirical evaluation of rectified activations in convolutional network," 2015, http://arxiv.org/abs/1505.00853.

[18] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proceedings of the International Conference on Learning Representations*, Banff National Park, Canada, April 2014.

[19] J. Springenberg, A. Dosovitskiy, T. Brox et al., "Striving for simplicity: the all convolutional net," in *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA, May 2015.