




GhostNeXt: Rethinking Module Configurations for Efficient Model Design

Kiseong Hong ¹, Gyeong-hyeon Kim ² and Eunwoo Kim ^{1,2,*}

¹ Department of Artificial Intelligence, Chung-Ang University, Seoul 06974, Republic of Korea

² School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, Republic of Korea

* Correspondence: eunwoo@cau.ac.kr

Abstract: Despite the continuous development of convolutional neural networks, it remains a challenge to achieve performance improvement with fewer parameters and floating point operations (FLOPs) as a light-weight model. In particular, excessive expressive power on a module is a crucial cause of skyrocketing the computational cost of the entire network. We argue that it is necessary to optimize the entire network by optimizing single modules or blocks of the network. Therefore, we propose GhostNeXt, a promising alternative to GhostNet, by adjusting the module configuration inside the Ghost block. We introduce a controller to select channel operations of the module dynamically. It holds a plug-and-play component that is more useful than the existing approach. Experiments on several classification tasks demonstrate that the proposed method is a better alternative to convolution layers in baseline models. GhostNeXt achieves competitive recognition performance compared to GhostNet and other popular models while reducing computational costs on the benchmark datasets.

Keywords: module configuration; resource-efficient network; network design



Citation: Hong, K.; Kim, G.-h.; Kim, E. GhostNeXt: Rethinking Module Configurations for Efficient Model Design. *Appl. Sci.* **2023**, *13*, 3301. <https://doi.org/10.3390/app13053301>

Academic Editor: Dimitris Mourtzis

Received: 26 January 2023

Revised: 25 February 2023

Accepted: 3 March 2023

Published: 4 March 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Deep convolutional neural networks have been continuously developed in image recognition [1], object detection [2], and semantic segmentation [3] tasks, to name a few. The networks have been designed toward deeper learning [4] to achieve additional performance gain. However, the network design requires a large amount of computational costs and is difficult to use universally in edge devices. To deploy it in edge devices that require real-time operation, studies on a light-weight model have been conducted [5–7].

The basic idea for designing a light-weight model is to reduce the computational cost in the convolution operation. Since spatial convolution with a 3×3 kernel size has a limitation in reducing network complexity, the network complexity has been reduced by minimizing input and output channel dimensions through 1×1 convolution [4]. Furthermore, depth-wise convolution [8] effectively reduces network complexity; thus, it has been widely used in most light-weight models.

Recently, GhostNet [9] has been proposed, which contains a Ghost module that can generate redundant feature maps at a low computational cost. The study [9] introduces two components in deep neural networks: extracting intrinsic feature maps and linear transforming them. When extracting the intrinsic feature maps, the existing 1×1 convolution [4] is used. Then, redundant feature maps are generated through a cheap linear transformation in the feature maps. It has become one of the representative light-weight models in that it contains a plug-and-play component that can replace the existing convolution layer and has the efficiency of extracting feature maps at a low computational cost.

Despite the fact that it has been widely used as a light-weight model, the module configuration of GhostNet raises concerns for designing a light-weight model. The Ghost block comprises an expansion module that significantly increases the input channel and

a reduction module that compresses the channel to match the output dimension of the block. This may seem appropriate for a light-weight model structure considering only the input and output channel dimensions. However, it involves up to six times channel expansion from the modules, resulting in unnecessary computational costs and limiting the expressiveness of the network [10,11].

In this paper, we propose a cost-efficient approach for designing light-weight convolutional neural network by introducing dynamic module configuration using a block-level controller based on the popular model. Our work aims to enhance the balance between the performance and computational complexity of GhostNet by reducing parameter consumption and improving its performance. Inspired by the rank analysis performed by ReXNet [11], we conjecture that, if the expansion module is used unnecessarily, the computational complexity of the network will be significant and additional performance gain may not be achieved due to the lack of expressive power. From this, we introduce a controller that can dynamically select channel operations of the module. We first define the set of channel reduction ratios for each block by referring to the design guide of the single expansion layer [11]. The controller selects one of the channel reduction ratios and determines the channel operation to construct the dynamic module configuration of each block. Accordingly, we propose GhostNeXt, which can effectively reduce the network complexity while improving the representation power by redesigning the module configuration of GhostNet.

To compare with existing light-weight models [6,7,9], we conduct experiments on diverse image classification benchmark datasets. Diverse fine-grained datasets, CUBS [12], Stanford Cars [13], Flowers [14], Wikiart [15], and Sketch [16] are used as large-scale datasets. Furthermore, CIFAR-100 [17], STL-10 [18], and CIFAR-10 [17] datasets are used as small-scale datasets. From the experiments, GhostNeXt is more efficient in terms of the trade-off between performance and computational complexity than light-weight models, including GhostNet. We also employ Grad-CAM [19] to visualize the importance of where the network focuses during prediction as a heat map. Through this, we empirically show that the dynamic module configuration enhances the expressive power of the network.

2. Related Work

The computation cost of a network highly depends on how we design the network configuration. Xception [5] was proposed by modifying the depth-wise separable convolution and using it instead of the Inception module [20]. MobileNet [8] was developed through the depth-wise separable convolution. MobileNetV2 [7] proposed an inverted residual block and MobileNetV3 [21] further designed a more efficient network in terms of performance and computational complexity using AutoML [22,23] strategy. ShuffleNet [24] used pointwise group convolution and channel shuffle. ShuffleNetV2 [6] proposed a network design to improve practical inference speed.

GhostNet [9] argued that similar feature maps can be extracted through convolution operations. It collects such feature maps at a low computational cost while preserving the intrinsic information through a cheap linear transformation. It is a plug-and-play component that can replace convolution layers of other convolutional neural networks while relieving the computational burden. Nonetheless, designing the network by stacking Ghost blocks weakens the advantages of the light-weight module due to the static module configuration inside the block. Therefore, we overcome the weakness in this work by presenting a better design configuration for a highly efficient model.

Recently, light-weight models based on neural architecture search (NAS) [25] have been proposed [11,26–28], which have achieved competitive performance by designing the network architecture automatically. EfficientNet [28] proposed a network design through compound scaling of network width, depth, and resolution. FBNet [26] proposed a gradient-based differentiable NAS framework to optimize the network. FBNetV2 [27] designed a memory-efficient network with an expanded search space of DNAS (differentiable NAS [26]), which contains a search for spatial and channel dimensions. ReXNet [11] redesigned the channel configuration of the inverted residual block. However, they do

not effectively consider the correlation and redundancy between feature maps. More recently, there have been proposals for light-weight vision transformers (ViTs) [29] such as MobileViT [30] and EdgeViTs [31]. MobileViT is mobile-friendly due to its efficient MobileViT block that encodes local and global information. EdgeViTs introduced the LGL bottleneck block to create a cost-effective transformer architecture. Unlike light-weight convolutional neural networks, ViT-based light-weight networks incorporate self-attention mechanisms, which pose a challenge in achieving a suitable balance between accuracy and computational complexity for mobile devices.

3. The Proposed Method

3.1. GhostNet

We study a dynamic module configuration to construct an improved light-weight module from GhostNet [9]. GhostNet is a stack of Ghost blocks and each block is composed of two modules. The module M inside the Ghost block is defined as:

$$M = L_{i,j}(X \circ f) \in \mathbb{R}^{h' \times w' \times d^{out}}, \quad (1)$$

where $X \in \mathbb{R}^{h \times w \times d^{in}}$ is an input, \circ is a channel operation that performs channel expansion or reduction on the module, and $f \in \mathbb{R}^{d^{mid} \times d^{in} \times k \times k}$ is a convolution filter. h , w , and d^{in} are the height, width, and the number of channels of the input, respectively. k and d^{mid} are the kernel size of the filter f and the number of channels in the output from the channel operation, respectively. $L_{i,j}(x)$ is a linear transformation function for x , which is a major component in GhostNet. i and j are in the ranges $[1, d^{mid}]$ and $[1, r]$, respectively. r is the number of linear transformations including one identity mapping. $L_{i,j}(x)$ performs linear transformation for each channel of d^{mid} and operates as an identity mapping when $j = r$. As a result, $L_{i,j}(x)$ always increases the channel number regardless of the module.

The static module configuration has significant issues that need to be addressed. First, the difference between the input and output of each module is significant, which leads to an increase in computational complexity inside the block, making it unsuitable for light-weight model design. Second, the channel expansion of up to six times inside the Ghost block is excessive, as it has been empirically proven that scaling between two and three times is optimal for designing convolutional neural networks [10]. Last, a recent work [11] has revealed that incorporating an excessive number of expansion layers in networks can result in bottleneck problems in representation. We design a block with a dynamic module configuration to tackle these challenges.

3.2. The Proposed Method: GhostNeXt

Our goal is to design a better lightweight block by adjusting the module configuration inside the Ghost block to reduce unnecessary computational costs:

$$\begin{aligned} \mathbf{B}^* &= \arg \max_{\mathbf{B}_1, \dots, \mathbf{B}_j} \text{Acc}(N(\mathbf{B}_1, \dots, \mathbf{B}_j)) \\ \text{s.t. } &\text{Params}(N) \leq P, \quad \text{FLOPs}(N) \leq F, \end{aligned} \quad (2)$$

where $k_i, s_i, c_i, r_i \in \mathbf{B}_i$, and Acc and \mathbf{B}_i denote the network validation accuracy and the i -th block configuration of the network N . k_i , s_i , c_i , and r_i denote the kernel size of the convolution filter, stride, and the number of channel and linear transformation of the i -th block, respectively. P and F denote the number of target parameters and FLOPs, respectively.

Since most parameters and FLOPs of the network are determined by c_i , we design the module configuration by introducing a controller to adjust c_i dynamically. We design the block by considering only c_i while k_i , s_i , and r_i are held fixed. In particular, we focus on adjusting d_1^{mid} , which is the number of channels of the first module M_1 . To optimize a block with two modules in a light-weight manner, we reduce the computational complexity

by controlling d_1^{mid} , a major component that increases the complexity of the blocks. We first define the set of channel reduction ratios for each block that serves as a criterion for dynamically adjusting c_i . Then, the controller selects one of the channel reduction ratios to design the dynamic module configuration. We define the set of channel reduction ratios as follows:

$$T_i = \left\{ t \mid \frac{d_1^{mid}}{t} \geq d_2^{mid}, t \in \mathbb{Z}^+ \right\}, \tag{3}$$

where T_i denotes the set of channel reduction ratios of the i -th block. t denotes the channel reduction ratios for the i -th block. We define the set of channel reduction ratios for light-weight blocks based on the design guide of the network [11]. Ref. [11] empirically proved that drastic channel expansion disrupts the expressive power of an expansion layer. Therefore, we allow the channel reduction ratio to reduce d_1^{mid} . In order to keep the block in a form of the inverted residual block [7], d_1^{mid}/t is always greater than or equal to d_2^{mid} . Finally, the module configuration for each block is constructed by the controller selecting one of the channel reduction ratios and determining the channel operation based on it, as shown in Figure 1. We define the controller C_i as:

$$C_i(t) = \begin{cases} R, & \text{if } d_1^{in} > \frac{d_1^{mid}}{t} \\ E, & \text{otherwise,} \end{cases} \tag{4}$$

where d_1^{in} denotes the input channel dimension of M_1 . R and E are the channel reduction and expansion operations, respectively. If channel operation is E for all blocks, the network architecture is equivalent to GhostNet. Note that the largest network in our module configuration is GhostNet. The computational cost of the i -th block decreases whenever the channel operation of E is not selected. If T_i satisfies the conditions described in Equation (3), it signifies that the stage has been reached where it is possible to manually examine all possible combinations for each block. In other words, it implies that the number of combinations is not large. Thus, the controller manually selects the channel reduction ratio to explore the most suitable module configuration, considering validation accuracy. This process is reminiscent of the approach taken in a previous study [32], where an empirical distribution function (EDF) was employed to narrow the search space.

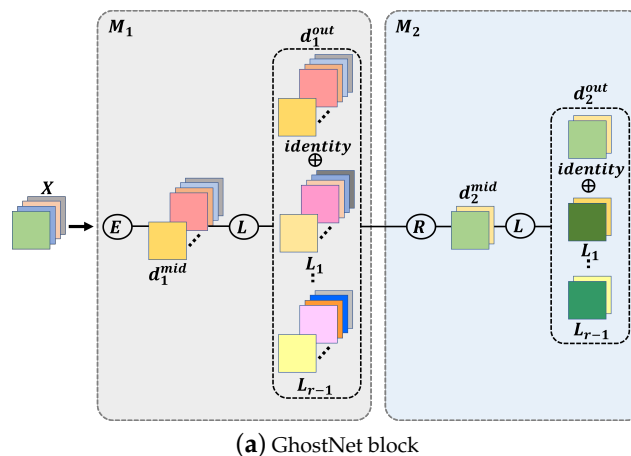


Figure 1. Cont.

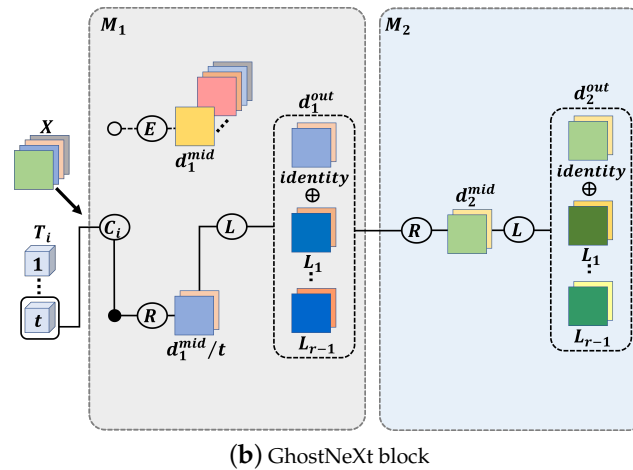


Figure 1. (a) The i -th GhostNet block. E , R , and L denote the channel expansion operation, reduction operation, and linear transformation function, respectively. The input X is expanded through E and L , where the difference between the output channel d_1^{out} of the first module and X is limited up to six times. d_1^{out} is the input of the second module and is reduced up to 12 times through R . (b) The i -th GhostNeXt block with the controller C_i . T_i denotes the set of channel reduction ratios of the i -th block. C_i selects t , one of the ratios in T_i to perform one of the channel operations E and R . If R is selected, d_1^{mid} is reduced by t and, if E is selected, it leads to the existing module configuration. Through L , the output channel d_1^{out} of the first module is generated by performing $r - 1$ linear transformation and one identity mapping. We use the channel operation R in the second module.

The presented dynamic module configuration has three positive effects over GhostNet. First, it is more general since the module configuration is not fixed and can be flexibly changed. Second, the presented controller can reduce unnecessary computational costs inside the Ghost block. Last, the proposed method can enhance the expressive power of the network (see Section 4.4 for the results). Table 1 shows the required parameters and FLOPs of M_1 and M_2 in the proposed method, respectively. Since the channel reduction ratio decreases the d^{mid} of two modules to d^{mid}/t , the parameters and FLOPs can be reduced. Note that when $t = 1$, it is equivalent to the number of parameters and FLOPs of GhostNet. Since t is greater than or equal to 1, the required parameters and FLOPs in GhostNeXt are always less than or equal to those of GhostNet.

Table 1. Parameters and FLOPs in the proposed method.

Module	Parameters (M)	FLOPs (M)
M_1	$\frac{d_1^{in} d_1^{mid} + (r-1)(d_1^{mid} k^2)}{t}$	$h'w'(\frac{d_1^{in} d_1^{mid} + (r-1)(d_1^{mid} k^2)}{t})$
M_2	$\frac{d_2^{in} d_2^{mid}}{t} + (r-1)(d_2^{mid} k^2)$	$h'w'(\frac{d_2^{in} d_2^{mid}}{t} + (r-1)(d_2^{mid} k^2))$

4. Experiments

4.1. Setup

We conducted experiments on large-scale datasets [12–16] with 224×224 pixels and small-scale datasets [17,18]. The datasets we used for the experiments are summarized in Table 2. We initially set the learning rate of the proposed method to 0.01 in all experiments and scheduled it by the cosine learning rate. The batch size and weight decay were set to 32 and 0.0005, respectively. We trained all the models using stochastic gradient descent (SGD) [33]. For data augmentation, we used random crop and random horizontal flip. The first layer of GhostNeXt consists of a standard convolutional layer with 16 filters. Then, the GhostNeXt blocks are stacked. The last global average pooling and convolutional layer serve to transform into a 1280-dimensional feature vector for final classification. The proposed architecture has a channel operation strategy (between E and R) for the three

divided groups (shallow, mid, and deep parts). We set P and F to the same number of parameters and FLOPs of GhostNet in all experiments. Furthermore, we set the hyperparameter r to 2 according to the experimental results of [9]. We regulate the width multiplier that can vary the network width to make FLOPs similar to competitors in all experiments. We represent GhostNeXt with width multiplier α as GhostNeXt ($\alpha \times$).

Table 2. Datasets used for experiments.

Dataset	CIFAR-10 [17]	CIFAR-100 [17]	STL-10 [18]	CUBS [12]	Stanford Cars [13]	Flowers [14]	WikiArt [15]	Sketch [16]
# Train	50,000	50,000	5000	5994	8144	2040	42,129	16,000
# Test	10,000	10,000	8000	5794	8041	6149	10,628	40,000
# Class	10	100	10	200	196	102	195	250
Image Size	32×32	32×32	96×96	224×224	224×224	224×224	224×224	224×224

4.2. Small-Scale Datasets

We first compared GhostNeXt with other light-weight models [6,7,9] on small-scale datasets. We used GhostNeXt (1.0 \times) as our base model with the module configuration $E - R - R$ representing the best trade-off between performance and computational complexity, based on experimental results in Section 4.4. Note that the module configuration $E - R - R$ is constructed by the controller selecting E , R , and R channel operations for each part. In Table 3, GhostNeXt outperforms other competitors with fewer parameters and FLOPs for most computational complexity levels and tasks. It consumes about three times fewer parameters than GhostNet. We can see that dynamic module configuration not only reduces unnecessary computational costs inside the block but also improves the expressiveness of the network. Figure 2 shows the results on CIFAR-10 under different parameters and FLOPs. Regardless of the number of parameters or FLOPs, GhostNeXt outperforms all other models. While GhostNet cannot maintain the performance for a small number of parameters, GhostNeXt exhibits a better trade-off between the number of parameters and performance than GhostNet. Furthermore, since GhostNet has a high parameter count compared to its FLOPs, when comparing performance with models that have similar FLOPs, it is observed that its performance is comparable to that of ShuffleNetV2. However, when models are compared based on similar parameters, GhostNet performs worse than ShuffleNetV2. In contrast, GhostNeXt shows the best trade-off among all models, indicating its flexibility in handling resource constraints. These results suggest that GhostNeXt is a promising alternative to the compared approaches for various applications that require efficient and accurate image classification.

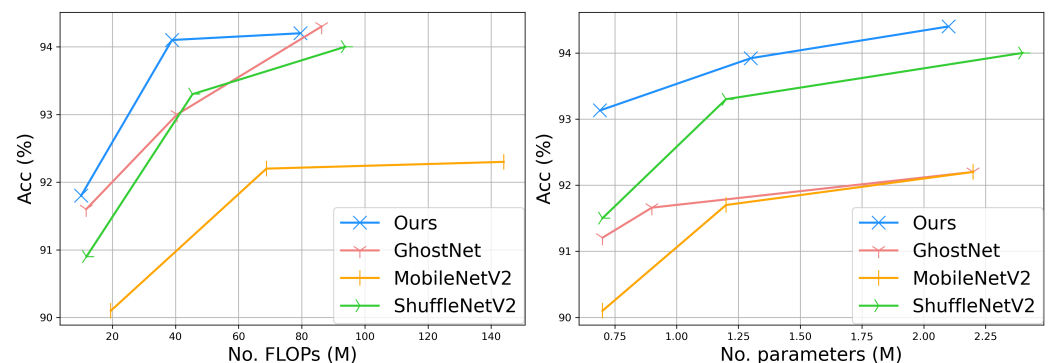


Figure 2. Performance under different parameter budgets and FLOPs on CIFAR-10.

We also conducted an experiment based on ResNet-56 to demonstrate the proposed method as a plug-and-play component, whose results are shown in Table 4. Ours-ResNet-56 denotes ResNet-56, in which a Ghost module replaces the convolutional layer with

a dynamic module configuration. Furthermore, Ghost-ResNet-56 denotes Ours-ResNet-56 without dynamic module configuration. Our proposed method provides comparable performance while reducing the parameter consumption and FLOPs by about 2.5 times and 1.5 times compared to Ghost ResNet-56. It can be employed for other convolutional neural networks stacked by modules or blocks and effectively reduce the computation cost of the network.

Table 3. Performance on the CIFAR-100, STL-10, and CIFAR-10 datasets.

Model	ACC (%)	CIFAR-100			STL-10			CIFAR-10		
		Params (M)	FLOPs (M)	ACC (%)	Params (M)	FLOPs (M)	ACC (%)	Params (M)	FLOPs (M)	
VGG-16	74.4	14.8	314.0	75.4	14.7	-	93.6	14.7	313.0	
ResNet-56	72.5	0.8	127.3	77.7	0.8	290.0	93.0	0.8	125.0	
ResNeXt-29 (2 × 64d)	79.8	7.6	263.6	80.6	7.5	606.6	96.0	7.5	263.5	
ResNeXt-29 (4 × 64d)	81.0	14.7	511.4	81.5	14.6	1160.0	96.2	14.6	511.3	
MobileNetV2 (0.5×)	68.1	0.8	19.6	71.3	0.7	49.6	90.1	0.7	19.5	
ShuffleNetV2 (0.5×)	69.2	0.4	11.9	71.4	0.3	26.6	90.9	0.3	11.8	
GhostNet (0.5×)	71.6	1.4	11.8	70.1	1.3	25.2	91.6	1.3	11.7	
Ours (0.7×)	71.5	0.5	10.3	71.2	0.4	22.5	91.8	0.4	10.1	
MobileNetV2 (1.0×)	72.9	2.3	68.9	73.2	2.2	161.0	92.2	2.2	68.8	
ShuffleNetV2 (1.0×)	73.7	1.3	45.5	74.7	1.2	102.3	93.3	1.2	45.4	
GhostNet (1.0×)	75.3	4.0	40.6	71.6	3.9	87.7	93.0	3.9	40.5	
Ours (1.5×)	75.4	1.4	39.1	75.7	1.3	87.0	94.1	1.3	38.9	
MobileNetV2 (1.5×)	72.8	5.1	144.7	74.4	4.9	355.0	92.3	4.9	144.0	
ShuffleNetV2 (1.5×)	75.7	2.5	93.9	75.5	2.4	211.0	94.0	2.4	93.8	
GhostNet (1.5×)	77.2	7.9	86.4	74.4	7.7	187.7	94.3	7.7	86.3	
Ours (2.2×)	76.2	2.6	79.8	77.1	2.5	178.1	94.2	2.5	79.6	

Table 4. Performance of the compared methods on CIFAR-10.

Model	ACC (%)	Params (M)	FLOPs (M)
ResNet-56 [4]	93.0	0.8	125.0
Ghost-ResNet-56	92.6	0.5	53.5
Ours-ResNet-56	92.5	0.2	38.0

4.3. Large-Scale Datasets

We also compared with other light-weight models [6,7,9] to demonstrate the superiority of GhostNeXt for recognizing large-scale images, as described in Table 2. From CUBS to Sketch, each is a fine-grained classification dataset consisting of birds, cars, flowers, paintings from artists, and sketches. Since these require better preservation of the fine-grained features of an image than a general coarse-grained classification, we considered that these are appropriate for determining whether detailed feature information is well preserved. Furthermore, we set it the same way, except that the batch size was increased to 64 in the setting of small-scale datasets. We resized all images to 256×256 pixels and random cropping to 224×224 pixels.

Table 5 shows the experimental results on the large-scale datasets. GhostNeXt outperforms other lightweight models, including GhostNet, with fewer parameters and FLOPs. In particular, it consumes about three times fewer parameters than the competitor and has the same number of parameters as MobileNetV2 but fewer FLOPs. The dynamic module configuration for each block effectively reduces the parameter consumption of GhostNet and preserves detailed feature information, resulting in performance improvements.

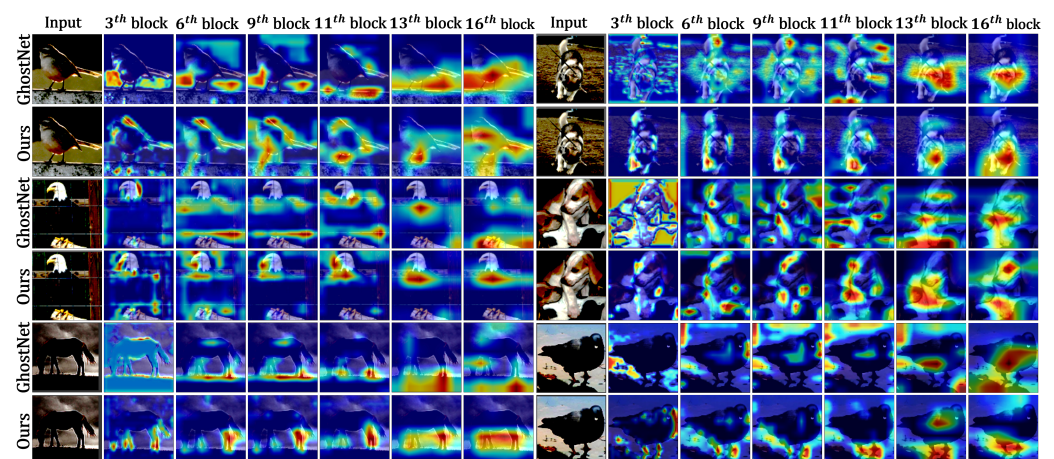
Table 5. Performance of the compared methods on the large-scale datasets.

Dataset	MobileNetV2 (0.6×)			ShuffleNetV2 (1.0×)			GhostNet (1.0×)			Ours (1.3×)		
	ACC (%)	Params (M)	FLOPs (M)	ACC (%)	Params (M)	FLOPs (M)	ACC (%)	Params (M)	FLOPs (M)	ACC (%)	Params (M)	FLOPs (M)
CUBS [12]	53.6	1.2	164.0	53.1	1.5	149.0	55.9	4.1	150.0	58.2	1.2	136.0
Stanford Cars [13]	76.0	1.2	164.0	75.7	1.5	149.0	79.0	4.1	150.0	80.3	1.2	136.0
Flowers [14]	58.2	1.1	164.0	60.5	1.4	149.0	64.8	4.0	150.0	66.3	1.1	136.0
WikiArt [15]	64.2	1.2	164.0	59.4	1.5	149.0	62.0	4.1	150.0	65.0	1.2	136.0
Sketch [16]	72.8	1.3	164.0	71.7	1.5	149.0	71.6	4.2	150.0	72.3	1.3	136.0

4.4. Analysis

We conducted an experiment to demonstrate the performance under different module configurations. The results of different strategies for selecting channel operations in each part are shown in Table 6. As the same channel operation can be chosen when the channel reduction ratio varies, we report the average accuracy, parameters, and FLOPs in the table. The table shows that dynamic module configuration effectively reduces the number of parameters and FLOPs compared to the existing module configuration. Most of the network configurations constructed by dynamic module configuration outperform or perform similarly to GhostNet. It can be observed from the table that selection in the deep part (close to the classification head) has a key influence on the number of parameters and FLOPs of the network. Moreover, continuous channel expansion can increase the complexity of the network.

We also experimented with demonstrating that the dynamic module configuration preserves the key feature of an image. We empirically prove this by providing a visualization of the output features of six different blocks using Grad-CAM [19]. We make notable observations from the results in Figure 3. GhostNet loses significant feature information. It primarily focuses on the background in the early layers and the object in the deep layers. In contrast, the proposed method can preserve meaningful information about object shapes from the early layer. It continuously preserves the meaningful shape information of an object throughout the network and thus gains additional performance improvement.

**Figure 3.** Visualization of features extracted from GhostNet and GhostNeXt on STL-10.

We conducted experiments to evaluate the expressiveness of the GhostNet and GhostNeXt quantitatively. This evaluation is to see how the dynamic module configuration affects the expressiveness of the network. We visualize the expressiveness represented by the nuclear norm with respect to the consumption of FLOPs and parameters. We used the validation set of CIFAR-10 to analyze the expressiveness of final features extracted from the trained networks in Table 6. The result in Figure 4 shows that GhostNeXt's selection of different channel operations for each part are similar to or more expressive than

GhostNet. We especially observed that networks with expansion operations selected in the deep part had higher expressiveness than those without the operations. Dynamic module configuration encourages expanding the expressiveness of the network while consuming fewer parameters and FLOPs than GhostNet. Furthermore, we analyzed the expressiveness by block in the network used in the Grad-CAM experiment. As shown in Figure 5, the expressiveness of GhostNeXt is higher than that of GhostNet in all blocks.

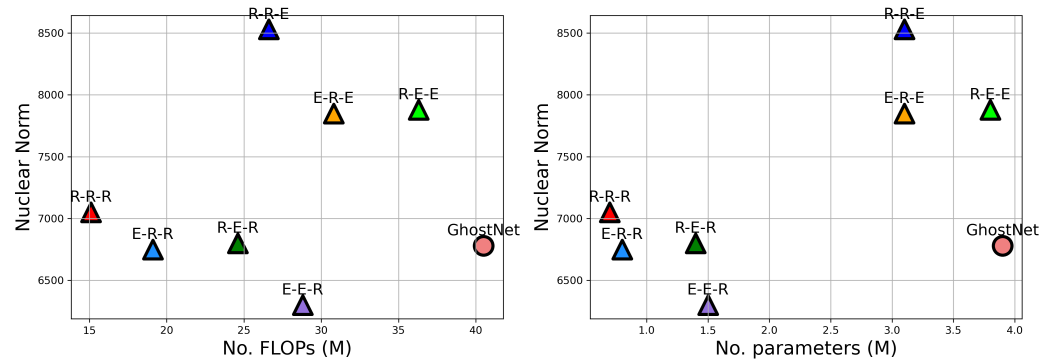


Figure 4. Visualization of the nuclear norm with respect to parameters and FLOPs on CIFAR-10.

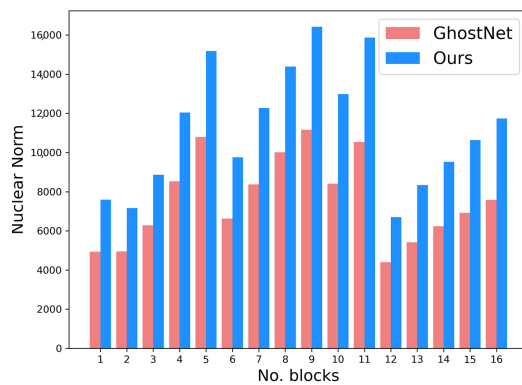


Figure 5. Visualization of the nuclear norm of the blocks of GhostNet and GhostNeXt on STL-10.

Table 6. Results with respect to different module configurations on CIFAR-10.

Model	ACC (%)	Params (M)	FLOPs (M)
$E - E - E$ (GhostNet)	93.0	3.9	40.5
$E - E - R$	93.4	1.5	28.8
$E - R - R$	93.1	0.8	19.1
$E - R - E$	93.2	3.1	30.8
$R - R - R$	92.5	0.7	15.1
$R - R - E$	93.0	3.1	26.6
$R - E - R$	92.9	1.4	24.6
$R - E - E$	93.3	3.8	36.3

5. Conclusions

We have proposed GhostNeXt, a simple yet effective dynamic module configuration approach. We argue that static module configuration results in unnecessary computational costs that are unsuitable for the light-weight network and also causes the lack of expressive power of the network. From this, we have introduced a controller that dynamically selects channel operations inside the module. The controller discovers the channel reduction ratio with a better trade-off between computational complexity and performance inside

the block. It can flexibly design the network structure depending on the selected channel reduction ratio. GhostNeXt is more efficient in terms of computational complexity than existing light-weight models and outperforms them in several experiments.

Author Contributions: Conceptualization, K.H.; funding acquisition, E.K.; investigation, K.H. and G.-h.K.; methodology, K.H.; resources, E.K.; software, K.H.; supervision, E.K.; validation, K.H. and G.-h.K.; visualization, K.H. and G.-h.K.; writing—original draft, K.H. and E.K.; writing—review and editing, K.H., G.-h.K. and E.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by BK21 FOUR (Fostering Outstanding Universities for Research) Program funded by Ministry of Education of Korea (No. I22SS7609062), in part by the Chung-Ang University Graduate Research Scholarship in 2023, and in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2021-0-01341, Artificial Intelligence Graduate School Program (Chung-Ang University)).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
2. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
3. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv* **2014**, arXiv:1412.7062.
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
5. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
6. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 116–131.
7. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
8. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
9. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. Ghostnet: More features from cheap operations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 1580–1589.
10. Radosavovic, I.; Kosaraju, R.P.; Girshick, R.; He, K.; Dollár, P. Designing network design spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 10428–10436.
11. Han, D.; Yun, S.; Heo, B.; Yoo, Y. Rethinking Channel Dimensions for Efficient Model Design. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 732–741.
12. Wah, C.; Branson, S.; Welinder, P.; Perona, P.; Belongie, S. The Caltech-Ucsd Birds-200-2011 Dataset. 2011. Available online: <https://www.kaggle.com/datasets/wenewone/cub2002011> (accessed on 20 January 2023).
13. Krause, J.; Stark, M.; Deng, J.; Fei-Fei, L. 3d object representations for fine-grained categorization. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 1–8 December 2013; pp. 554–561.
14. Nilsback, M.E.; Zisserman, A. Automated flower classification over a large number of classes. In Proceedings of the 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, Bhubaneswar, India, 16–19 December 2008; pp. 722–729.
15. Saleh, B.; Elgammal, A. Large-scale classification of fine-art paintings: Learning the right metric on the right feature. *arXiv* **2015**, arXiv:1505.00855.
16. Eitz, M.; Hays, J.; Alexa, M. How do humans sketch objects? *ACM Trans. Graph. (TOG)* **2012**, *31*, 1–10. [CrossRef]
17. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. 2009. Available online: <https://www.kaggle.com/competitions/cifar-10/data> (accessed on 20 January 2023).

18. Coates, A.; Ng, A.; Lee, H. An analysis of single-layer networks in unsupervised feature learning. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 215–223.
19. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
20. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
21. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for mobilenetv3. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
22. Yang, Z.; Wang, Y.; Chen, X.; Shi, B.; Xu, C.; Xu, C.; Tian, Q.; Xu, C. Cars: Continuous evolution for efficient neural architecture search. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 1829–1838.
23. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8697–8710.
24. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6848–6856.
25. Zoph, B.; Le, Q.V. Neural architecture search with reinforcement learning. *arXiv* **2016**, arXiv:1611.01578.
26. Wu, B.; Dai, X.; Zhang, P.; Wang, Y.; Sun, F.; Wu, Y.; Tian, Y.; Vajda, P.; Jia, Y.; Keutzer, K. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 10734–10742.
27. Wan, A.; Dai, X.; Zhang, P.; He, Z.; Tian, Y.; Xie, S.; Wu, B.; Yu, M.; Xu, T.; Chen, K.; et al. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 12965–12974.
28. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning. PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
29. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
30. Mehta, S.; Rastegari, M. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv* **2021**, arXiv:2110.02178.
31. Pan, J.; Bulat, A.; Tan, F.; Zhu, X.; Dudziak, L.; Li, H.; Tzimiropoulos, G.; Martinez, B. Edgevits: Competing light-weight cnns on mobile devices with vision transformers. In Proceedings of the Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, 23–27 October 2022; Proceedings, Part XI; Springer: Switzerland, 2022; pp. 294–311.
32. Radosavovic, I.; Johnson, J.; Xie, S.; Lo, W.Y.; Dollár, P. On network design spaces for visual recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1882–1890.
33. Robbins, H.; Monro, S. A stochastic approximation method. In *The Annals of Mathematical Statistics*; 1951; pp. 400–407. Available online: <https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-22/issue-3/A-Stochastic-Approximation-Method/10.1214/aoms/1177729586.full> (accessed on 10 December 2022).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

Reproduced with permission of copyright owner. Further reproduction prohibited without permission.