

PokerNet: Expanding Features Cheaply via Depthwise Convolutions

Wei Tang^{1,2} Yan Huang^{1,2} Liang Wang^{1,2}

¹National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

²Center for Research on Intelligent Perception and Computing (CRIPAC), Institute of Automation,
Chinese Academy of Sciences, Beijing 100190, China

Abstract: Pointwise convolution is usually utilized to expand or squeeze features in modern lightweight deep models. However, it takes up most of the overall computational cost (usually more than 90%). This paper proposes a novel Poker module to expand features by taking advantage of cheap depthwise convolution. As a result, the Poker module can greatly reduce the computational cost, and meanwhile generate a large number of effective features to guarantee the performance. The proposed module is standardized and can be employed wherever the feature expansion is needed. By varying the stride and the number of channels, different kinds of bottlenecks are designed to plug the proposed Poker module into the network. Thus, a lightweight model can be easily assembled. Experiments conducted on benchmarks reveal the effectiveness of our proposed Poker module. And our PokerNet models can reduce the computational cost by 7.1%–15.6%. PokerNet models achieve comparable or even higher recognition accuracy than previous state-of-the-art (SOTA) models on the ImageNet ILSVRC2012 classification dataset. Code is available at <https://github.com/diaomin/pokernet>.

Keywords: Deep learning, depthwise convolution, lightweight deep model, model compression, model acceleration.

Citation: W. Tang, Y. Huang, L. Wang. PokerNet: Expanding features cheaply via depthwise convolutions. *International Journal of Automation and Computing*, vol.18, no.3, pp.432–442, 2021. <http://doi.org/10.1007/s11633-021-1288-x>

1 Introduction

The past decade has witnessed the great progress of deep convolutional neural networks (DCNNs) in computer vision. DCNNs have made breakthroughs in various vision tasks such as image classification^[1–3], object detection^[4–6], and image segmentation^[7–9]. Although amazing accuracy has been achieved, the high computational cost of several to dozens of BFLOPs (billions of floating-point operations per second) is a headache in practical applications. Especially when dealing with visual recognition problems in mobile and embedded scenarios, such as mobile phones, robots, unmanned vehicles, etc., high computational cost often means high hardware requirements and high power consumption. Therefore, in recent years, how to design efficient lightweight deep models while ensuring high performance has been a hot topic. More and more researchers have been attracted to this area.

The research on lightweight deep models is of great significance. In addition to the benefits of reducing hard-

ware requirements and being energy-saving, lightweight deep models also have the following advantages:

- 1) It makes edge computing possible. This results in visual feedback being more timely. It is important for scenarios that require real-time feedback, such as self-driving cars.
- 2) Personal privacy is protected by avoiding uploading the original images data to the server.
- 3) It saves network bandwidth for there is no need to transfer images frequently between different hardware.
- 4) It reduces the cost of using visual recognition models so that it is beneficial to industry development.

A series of techniques have been proposed on lightweight deep models in recent years including model pruning^[10, 11], model quantification^[12, 13], lightweight model structure design^[14, 15], knowledge distillation^[16–18], etc. Among these, the lightweight model architecture design is one of the most attractive techniques. Lightweight model architecture design has two obvious advantages: It is not limited by the pre-trained model and it can be easily customized for different recognition tasks. MobileNet^[14], one of the representative works in this area proposes a kind of convolution operation called separable convolution layer. It decouples a traditional convolution layer into a pointwise convolution layer and a depthwise convolution layer. DCNNs are constructed with separable convolution layer and compact models with comparable performance are

Research Article

Manuscript received December 23, 2020; accepted February 1, 2021;
published online March 24, 2021

Recommended by Associate Editor Bin Luo

Colored figures are available in the online version at <https://link.springer.com/journal/11633>

© Institute of Automation, Chinese Academy of Sciences and Springer-Verlag GmbH Germany, part of Springer Nature 2021

achieved. ShuffleNet^[15] explores channel shuffle operations with grouped convolution to further improve the efficiency and accuracy of lightweight models. The latest GhostNet^[19] proposes a module called Ghost module to concatenate the output features from a pointwise convolution layer and a following cheap depthwise convolution layer. The Ghost module can reduce the overall computational cost by reducing the usage of pointwise convolutions while retaining rich features.

Abundant and even redundant features are proved to be essential to guarantee the performance^[19]. Thus, a block design with first feature expansion and then feature reduction has become a standard procedure in the lightweight model architecture design works. Feature expansion and reduction rely on pointwise convolutions because pointwise convolutions can vary the output channels of features. Especially with the introduction of inverted residual bottleneck^[20], the pointwise convolution is extensively utilized to expand or squeeze features. This leads to the pointwise convolution usually accounting for more than 90% of the overall computational cost in lightweight deep models.

This paper aims at solving the problem of excessive computational cost in the feature expansion stage. We explore effective ways to reduce the computational cost while ensuring to obtain abundant features. This results in a novel Poker module being proposed here. The proposed Poker module can generate abundant effective features via groups of cheap depthwise convolutions. To be specific, a small pointwise convolution layer does the channel correlation first, then groups of depthwise convolutions are employed and finally a self-attention mechanism is utilized to further enhance the expanded features. With the same input and output channels, the overall computational cost in the proposed Poker module gets $1.5\times$ to $3.0\times$ fewer computational cost compared with the module in previous works. Based on the Poker module, we design the efficient lightweight models, PokerNets. Experiments on the large scale image classification dataset verify that PokerNets surpass the SOTA lightweight models such as GhostNet^[19], MobileNet series^[14, 20, 21] and ShuffleNet series^[15, 22], etc. PokerNets can reduce the

computational cost with a large margin while obtaining comparable or even higher recognition accuracy.

The rest of the paper is organized as follows. Section 2 concludes the related work in the area. The proposed Poker module and PokerNets in Section 3. the experiments and the analysis in Section 4. Finally conclusions are given in Section 5.

2 Related work

Building lightweight deep convolutional neural networks with a good balance between computing efficiency and recognition accuracy has been a hot topic in recent several years. Hand-craft architecture design is the mainstream at first. However, with the rise of deep reinforcement learning, automatic neural architecture search has attracted more and more attention. These two branches are the most related to this paper.

As far as we know, NIN^[23] is the first work that adopts a large number of pointwise convolutions (also called 1×1 convolution) to build a deep convolutional neural network. It achieves comparable accuracy with AlexNet^[1] but obviously with far fewer parameters. Aiming directly at the goal of reducing the parameters of DCNNs, SqueezeNet^[24] employs tremendous numbers of pointwise convolutions with expansion and squeeze operations and obtains deep models with parameters of several or even less than 1 MB. Recent work begins focusing on multiply-addition operations (MAdds), a more direct and fair evaluation indicator of computing efficiency, instead of the number of parameters of the neural network. MobileNetV1^[14] creatively proposes a more efficient convolution structure, separable convolution layer. It decouples the traditional convolution layer into a pointwise convolution layer responsible for channel correlation and a depthwise convolution layer responsible for spatial correlation as illustrated in Fig. 1. Following this work, MobileNetV2^[20] and MobileNetV3^[21] further improve the efficiency as well as the accuracy by optimizing the residual bottleneck design and combining the hand-craft method with the automatic neural architecture search, respectively. ShuffleNet^[15] proposes grouped convolution integ-

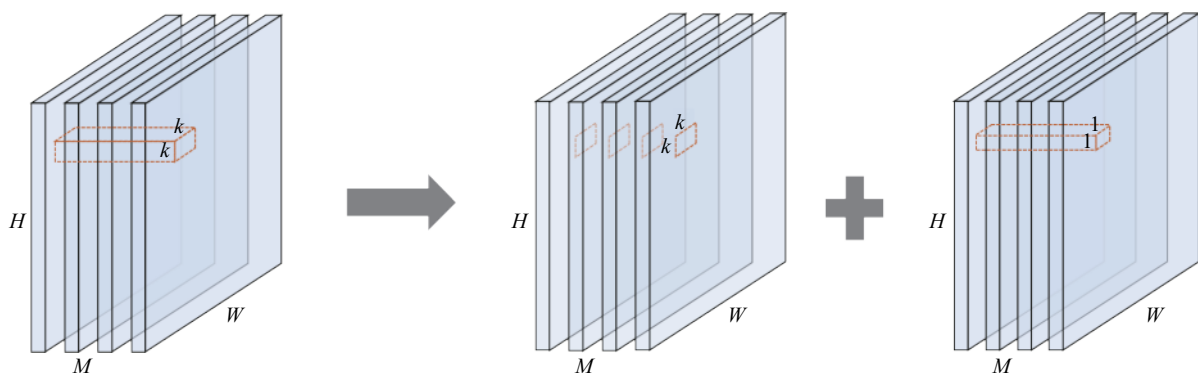


Fig. 1 Traditional convolution layer (left) and separable convolution layer consisting of a depthwise convolution layer (middle) and a pointwise convolution layer (right).

rated with channel shuffle to further reduce the computational cost. ShuffleNetV2^[22] drops the grouped convolution proposed in ShuffleNet^[15] and optimizes the network architecture based on the principles from statistics of practice experiments. ShiftNet^[25] employs the shift operation alternated with pointwise convolutions instead of expensive spatial convolution. The latest GhostNet^[19] concatenates the features from a small pointwise convolution layer and the features from a follow-up depthwise convolution layer to reduce the computational cost while obtaining abundant features.

The rising of deep reinforcement learning brings new vitality to lightweight network architecture design. Works^[26–32] are the pioneers to employ deep reinforcement learning to automate the architecture design. Limited by the exponentially growing search space, early works can only focus on cell level structure search. MnasNet^[27] makes block-level structure search possible by introducing multi-target Pareto optimal and simplifying the search space from cell-level to block-level (i.e., grouping convolutional layers into a number of predefined skeletons). Then, differentiable architecture framework with gradient-based optimization utilized in works^[26, 33, 34] further reduces the computational cost of structure search. Automatic neural architecture design algorithms especially focusing on lightweight deep model design have emerged in works^[34–38]. It is worth emphasizing that MobileNetV3^[21] integrates the hand-craft methods and automatic architecture design methods together, which presents a promising trend of the fusion of the two branches.

Our work can be treated as a hand-crafted method. We propose a novel module to reduce the MAdds of a network by decreasing the use of pointwise convolutions in the feature expansion stage, at the meanwhile to generate abundant effective features to guarantee performance. Methods such as automatic architecture search^[26–34], quantization^[12, 13, 39] and network pruning^[10, 11, 40–45] are orthogonal to our method and can be combined to further boost the performance.

3 PokerNet

In this section, we first illustrate how the Poker module is composed and can generate a large number of effective features via depthwise convolutions. Then, we design the bottlenecks according to different kinds of stride and channels. Finally, an efficient PokerNet architecture is developed with high recognition accuracy.

3.1 Poker module for feature expansion

Recent works on building lightweight deep convolutional neural networks such as MobileNet series^[14, 20, 21], ShuffleNet series^[15, 22] and GhostNet^[19] always combine the pointwise convolution and depthwise convolution as

the base module to develop efficient CNNs. In these works, pointwise convolutions usually take up more than 90% of the overall computational cost. Especially at the stage of feature expansion, previous works mainly rely on the pointwise convolution layer because depthwise convolution layer can only generate features with the output channels equal to input channels.

Considering a convolution layer with input features $X \in \mathbf{R}^{m \times w \times h}$, where m is input channels and h and w are the height and width of features, respectively. Assume the kernel size is $d \times d$, the stride equals to 1 and the number of output channels is n , where $n = km$ ($k = 1, 2, 3, \dots$) because we only consider the expansion stage.

3.1.1 Complexity of ghost module

Given the latest state-of-the-art (SOTA) Ghost module as an example, as illustrated in Fig. 2, the Ghost module firstly takes the m features as input and generates the first $n/2$ features by utilizing pointwise convolutions. Then, it generates another $n/2$ features by using depthwise convolutions with the former $n/2$ features as input. Finally, the two parts of $n/2$ features are concatenated together to form the final n output features.

The complexity of a Ghost module is

$$C_g = h \times w \times m \times \frac{n}{2} + h \times w \times d^2 \times \frac{n}{2} = h \times w \times m \times \left(\frac{km}{2} + \frac{kd^2}{2} \right). \quad (1)$$

As shown in (1), in practice, d is often taken as 3 and $m \gg d^2$. This leads to pointwise convolutions always occupying the most amount of the overall computational cost in lightweight deep models.

3.1.2 Proposed poker module

Based on the above observation, we have been thinking about whether there is another way to generate abundant effective features while reducing the usage of

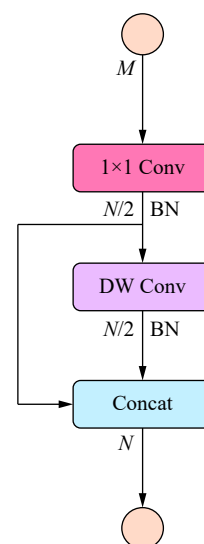


Fig. 2 Ghost module from GhostNet^[19]

pointwise convolutions. We point out that it is solvable and the Poker module we propose here can attain this goal.

As illustrated in Fig. 3, the proposed Poker module is composed of three parts. The first part is a small pointwise convolution layer with input channels and output channels keeping the same number of M . The second part is the crucial expansion part. It utilizes k groups of low-cost depthwise convolutions to expand the features from the number of M to kM , where k is the expansion factor as well as the number of groups and it can be customized in practice. A Squeeze-and-Excite (SE) layer^[46] is adopted in the third part, which is vital and responsible for enhancing or suppressing the features selectively according to the response value acquired through a self-attention mechanism.

In practice, the first and second parts are followed by batch normalization (BN)^[47] and ReLU activation^[1]. It should be emphasized that the third part is integral in our module, the importance of which will be further analyzed in the experimental part.

Our proposed Poker module has three major differences from the existing efficient model design. 1) Existing methods such as the MobileNet series^[14, 20, 21] and ShuffleNet series^[15, 22], utilize pointwise convolution to expand features, which is computationally expensive. In contrast, the Poker module first adopts a small pointwise convolution to do channel correlation and then groups of low-cost depthwise convolutions are utilized to expand features, which can largely reduce the computational cost as to be analyzed below. 2) Compared with the module in GhostNet^[19] which concatenates the features from a pointwise convolution layer and a followed depthwise con-

volution layer, the groups of depthwise convolutions in the Poker module can be customized. 3) the Squeeze-and-Excite layer is just a choice in existing methods, but it is an essential part in our design.

3.2 Design of poker bottlenecks

When it comes to bottleneck design, we classify them into 3 different situations. 1) Stride is equal to 1 and input channels are equal to output channels. 2) Stride is equal to 1 and input channels are not equal to output channels. 3) Stride is equal to 2 no matter what the input channels and output channels are.

Taking advantage of the proposed Poker module, we redesign the bottlenecks as shown in Fig. 4. Following the inverted residual bottleneck^[20] design concept, in the main branch, we first expand the input features from the previous bottleneck by utilizing the Poker module, then the Ghost module is adopted to narrow the features because it is more efficient under this condition. Particularly, when the stride is equal to 2, a depthwise convolution with stride equals to 2 is inserted between two modules to downsample the feature maps as Fig. 4(c) shows. In the shortcut branch of situations 2) and 3), a depthwise convolution with stride equals to 1 or 2 accordingly followed by a pointwise convolution is added to the shortcut path as Figs. 4(b) and 4(c) illustrate. As for situation i), the input is added to the output of the main branch directly.

With the design of different bottlenecks, a light-weight deep model can be assembled through choosing and stacking these bottlenecks according to the input channels, the output channels and the size of feature maps.

3.3 Architecture of PokerNet

Following the basic architecture of GhostNet^[19] and MobileNetV3^[21] for their preponderance, we propose the PokerNet by replacing the Ghost bottlenecks with the Poker bottlenecks. Table 1 shows the overall architecture of our proposed PokerNet. Some bottlenecks are kept unchanged because the computational cost of Ghost bottlenecks in these parts is lower than that of Poker bottlenecks. The selection criteria between traditional Ghost bottleneck and Poker bottleneck will be given in detail later. PokerNet can be divided into three parts, i.e., the head, the body, and the end. The head is a standard convolution layer with stride of 2 and 20 channels. The body is a stack of Poker bottlenecks or Ghost bottlenecks. At the last layer of each stage, the bottleneck will increase the channels and downsample the features gradually except for the last stage. The last layer in the last stage adopts a pointwise convolution directly to expand features from the number of 160 to 960. It should be noticed that,

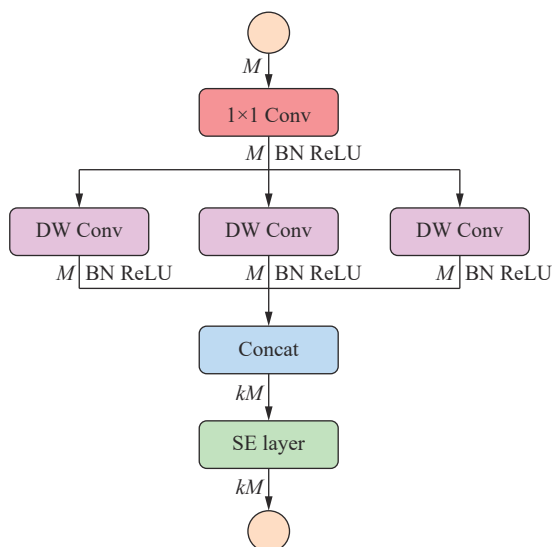


Fig. 3 An illustration of the proposed Poker module. Features are expanded from the number of M generated by pointwise convolution to the number of kM by using k groups of cheap depthwise convolutions in parallel. A SE layer^[46] is followed to enhance or suppress the expanded features.

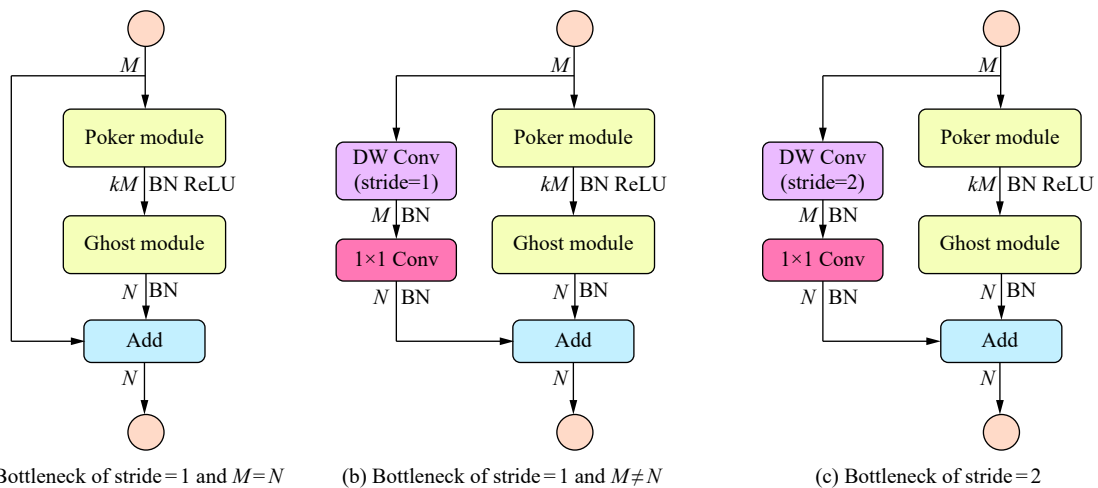


Fig. 4 Design of bottlenecks. Left: Poker bottleneck with stride equal to 1 and M equal to N ; Middle: Poker bottleneck with stride equal to 1 and M not equal to N ; Right: Poker bottleneck with stride equal to 2.

Table 1 Overall architecture of PokerNet-1.0 \times . Size, #in, #exp and #out denote the input feature map size, input channels, expansion channels and output channels of a bottleneck, respectively. P-bneck and G-bneck denote Poker bottleneck and Ghost bottleneck, respectively. Act denotes the activation type.

Size	Ops	#in	#exp	#out	SE	Stride	Act
224	Conv2d 3x3	3	–	20	–	2	ReLU6
112	G-bneck	20	20	20	–	1	ReLU6
112	G-bneck	20	40	20	–	2	ReLU6
56	G-bneck	20	60	20	–	1	ReLU6
56	G-bneck	20	120	40	–	2	ReLU6
28	P-bneck	40	120	40	1	1	ReLU6
28	P-bneck	40	240	80	1	2	ReLU6
14	P-bneck	80	160	80	1	1	ReLU6
14	G-bneck	80	160	80	1	1	ReLU6
14	G-bneck	80	160	80	1	1	ReLU6
14	P-bneck	80	240	120	1	1	ReLU6
14	P-bneck	120	480	120	1	1	ReLU6
14	P-bneck	120	720	160	1	2	ReLU6
7	P-bneck	160	960	160	1	1	HS
7	P-bneck	160	960	160	1	1	HS
7	P-bneck	160	960	160	1	1	HS
7	P-bneck	160	960	160	1	1	HS
7	Conv2d 1x1	160	–	960	–	1	HS
7	AvgPool 7x7	960	–	960	–	1	ReLU6
1	Conv2d 1x1	960	–	1 280	–	1	ReLU6
1	Conv2d 1x1	1 280	–	1 000	–	1	ReLU6

in the first two stages and the second and third bottlenecks in the fourth stage, Ghost bottlenecks are adopted for their lower computational cost according to the feature size and channels. The end is an average pooling layer followed by a pointwise convolution layer to transform the feature vector to 1 280 dimensions before being put to the final classifier. The expansion channels of the first

three bottlenecks at the fourth stage are slightly different from the number in GhostNet and MobileNetV3 for the expansion factor k should be an integer. The architecture illustrated in Table 1 is just a reference. Other hyper-parameter tuning methods can be used for further boosting performance.

We also utilize a width multiplier α to customize the

width of the network for considering the limitation of computing resources of hardware in practical usage. It is denoted as PokerNet- $\alpha\times$ when using PokerNet with width multiplier α . With a smaller α , the computational cost of the model roughly decreases α^2 , but this will lead to lower performance with no doubt. In the experiments, we adopt two kinds of α , i.e., 0.5 and 1.0. Correspondingly, we denote the models as PokerNet-0.5 \times , PokerNet-1.0 \times .

The overall architecture of both PokerNet-0.5 \times and PokerNet-1.0 \times will be illustrated in detail in the supplementary materials.

4 Experiments

In this section, we first detail the training setup and evaluation metrics utilized in our experiments. Then the computational cost of the proposed Poker module will be analyzed in detail. PokerNet models built with the novel Poker module will be tested on the large scale image classification benchmark. Finally, an ablation study on the influence of the Squeeze-and-Excite layer on the performance and an ablation study on verifying the effectiveness of a kind of nonlinearity called *hard-swish* will be further explored.

4.1 Dataset and evaluation metric

As has become standard, ImageNet^[48] is adopted for all our classification experiments to verify the effectiveness of our proposed Poker module and Poker models. ImageNet is a large-scale image dataset. It contains over 1.2 M training images and 50K validation images belonging to 1 000 classes.

For the pre-processing strategy^[49, 50] of the ImageNet dataset, firstly, all the images are resized with the shorter side of 256. Then, before sending images to the network, we randomly crop 224×224 image patches with mean subtraction, randomly flipping and color jittering. No other data augmentation tricks such as multi-scale are adopted. The final model is evaluated on the validation dataset with only a single center crop.

Our method is evaluated in terms of multiply-addition operations (MAdds) and recognition accuracy. As for the latency, we compare our proposed PokerNet models with current SOTA GhostNet models under different width multipliers on an Intel CPU I7-7700K@4.2GHZ with 32GB RAM. It is worth emphasizing that the inference speed highly relies on the code optimization ability and the hardware platform. In our experiments, we just make the k groups of depthwise convolutions in parallel by using multithreading. Actually, our Poker module has excellent parallelism and all the kM convolutions can be processed in parallel. So the inference speed of PokerNet models can be further improved with better code optimization.

4.2 Training setup

To train PokerNet models, all convolutional layers are followed by batch normalization layers. Both the batch-normalization layer and activation layer are adopted after the first module (Poker module or Ghost module) in the bottlenecks. The second module only uses the batch-normalization layer.

Our models are trained on 8 GPUs using standard pytorch stochastic gradient descent (SGD) optimizer with 0.9 momentum. Most of the hyper-parameter settings in our experiments follow the work of ShuffleNet^[22]. We use a batch size of 1 024 (128 images per GPU), total epochs of 300, multi-step learning rate policy (i.e., [160, 240, 260, 280]), dropout of 0.2 on the features before the classifier layer, initial learning rate of 0.5 and label smoothing of 0.1. The weight decay of PokerNet-0.5 \times is set to 1×10^{-5} and that of PokerNet-1.0 \times is set to 4×10^{-5} because we find that they face different degrees of over-fitting.

4.3 Analysis on computational cost

From Fig. 3 illustrated above, we can get that the computational cost of our proposed Poker module is

$$C_p = h \times w \times m \times m + h \times w \times d^2 \times m \times k = h \times w \times m \times (m + k \times d^2). \quad (2)$$

For simplicity, the cost of the Squeeze-and-Excite part is omitted because it takes up very little computational cost compared with other parts.

When we compare it with the latest SOTA Ghost module, we can get the ratio

$$\frac{C_g}{C_p} = \frac{2}{k} \times \frac{m + kd^2}{m + d^2}. \quad (3)$$

When d is usually taken as 3 in practice, we plot the ratio curves under different k values and m values, which are shown in Fig. 5.

As Fig. 5 illustrates, only when it comes to $k = 2$ and some cases where m and k are small, the ratio is less than 1. In most cases, the Poker module can achieve 1.5 \times to 3 \times fewer computational cost than the Ghost module. This can explain why we utilize Poker bottlenecks in most of the situations in our PokerNet but Ghost bottlenecks in the first two stages and the second and third bottlenecks in the fourth stage as shown in Table 1. Fig. 5 can be treated as the selection criteria between Poker bottlenecks and other bottlenecks when building different kinds of PokerNets.

4.4 PokerNets on large scale image recognition task

We mainly evaluate our models on the ImageNet 2012

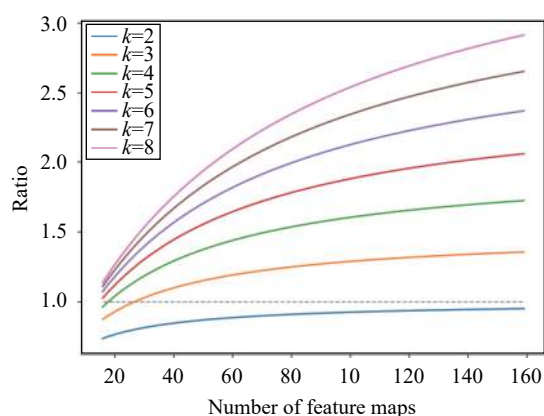


Fig. 5 Computational cost comparison between the Ghost module and the proposed Poker module under different k (expansion factor) and m (number of feature channels) values.

classification dataset. Several modern lightweight network architectures are selected as competitors, including MobileNet series[14, 20, 21], ShuffleNet series[15, 22], GhostNet[19], etc. The results are summarized in Table 2. Models, such as ProxylessNAS[26], FBNet[34] and MnasNet[27], are not listed in the table because their computational cost is much higher than our models (greater than 200M).

In terms of computational cost, we follow [19] and group these models into two levels, i.e., $\sim 50\text{M}$ MAdds, $\sim 150\text{M}$ MAdds. A consistent observation we can get from the table is that larger MAdds usually mean higher accuracy in lightweight models.

As can be seen from the results illustrated in Table 2, PokerNet outperforms all the competitors at every computational cost level. Compared with the previous SOTA Ghost models, our models achieve 7.1% (PokerNet-0.5 \times) and 15.6% (PokerNet-1.0 \times) MAdds reduction with even higher recognition accuracy.

The advantage of computational cost of our proposed PokerNet models can also be seen from the latency test shown in Table 3. By simply making the k groups of depthwise convolutions in parallel using multithreading, our PokerNet models run much faster than current SOTA GhostNet models. If all the kM convolutions are optimized to run in parallel, this advantage will be more obvious. This is due to the efficient Poker module we design. With the combination of groups of cheap depthwise convolution SE layer, the Poker module we propose can ensure the generation of abundant effective features, while minimizing the computational cost.

Fig. 6 gives a more intuitive display of the comparison of PokerNets and different SOTA models in terms of computational cost and Top-1 accuracy.

4.5 Ablation study on SE layer

The core idea of the Poker module is to utilize groups of depthwise convolutions to generate a large number of features and adopt the SE layer to enhance or suppress the generated features. In Section 3.1, we point out that the SE layer is essential to guarantee the performance in our design of Poker module.

To evaluate the importance of the SE layer in the Poker module, we conduct the ablation study experiments on PokerNet-0.5 \times and PokerNet-1.0 \times . We compare the final recognition accuracy with the SE layer being added to the Poker module or omitted. All other settings are kept exactly the same.

The results are shown in Table 4. From the results, we can see that the Poker modules with the SE layer slightly increase the MAdds (0.4M in PokerNet-0.5 \times and 2M in PokerNet-1.0 \times), but they outperform the one

Table 2 Comparison of SOTA lightweight models over parameters, MAdds and accuracy on ImageNet dataset. M stands for millions.

Models	Params (M)	MAdds (M)	Top-1 Acc. (%)	Top-5 Acc. (%)
ShuffleNetV1 0.5 \times (g=8) ^[15]	1.0	40	58.8	81.0
MobileNetV2 0.35 \times ^[20]	1.7	59	60.3	82.9
ShuffleNetV2 0.5 \times ^[22]	1.4	41	61.1	82.6
MobileNetV3 Small 0.75 \times ^[21]	2.4	44	65.4	–
GhostNet 0.5 \times ^[19]	2.6	42	66.2	86.6
PokerNet 0.5\times	2.4	39	66.8	87.0
MobileNetV1 0.5 \times ^[14]	1.3	150	63.3	84.9
MobileNetV2 0.6 \times ^[20]	2.2	141	66.7	–
ShuffleNetV1 1.0 \times (g=3) ^[15]	1.9	138	67.8	87.7
ShuffleNetV2 1.0 \times ^[22]	2.3	146	69.4	88.9
MobileNetV3 Large 0.75 \times ^[21]	4.0	155	73.3	–
GhostNet 1.0 \times ^[19]	5.2	141	73.9	91.4
PokerNet 1.0\times	5.7	119	73.6	91.5

Table 3 Comparison of inference speed between PokerNet models and GhostNet models (current SOTA lightweight models) under different width multipliers. The batch size is equal to 1.

Models	Latency (ms)
PokerNet 0.5×	32.9
GhostNet 0.5×	36.7
PokerNet 1.0×	46.2
GhostNet 1.0×	51.3

without the SE layer with a large margin (3.3% in PokerNet-0.5× and 3.0% in PokerNet-1.0× for Top-1 accuracy). It strongly favors why we take the SE layer as an important part of the proposed Poker module.

4.6 Ablation study on nonlinearity

In [23], a nonlinearity called *hard-swish* was introduced, which proves to significantly improve the accuracy of lightweight deep models. The hard version of swish is defined as follows:

$$\text{hard-swish}(x) = x \frac{\text{ReLU6}(x + 3)}{6}.$$

To study the influence of *hard-swish* on PokerNet models, we follow the work[21] and replace the nonlinearities of deeper layers of PokerNet models from ReLU to *hard-swish*. This is because *hard-swish* can get the most benefits while occupying less computational cost in deeper layers as claimed in [21]. In practice, *hard-swish* is utilized in the layers of fifth stage as shown in Table 1.

We compare the PokerNet models under 3 kinds of settings of nonlinearities, i.e., ReLU, ReLU6 and ReLU6 + *hard-swish*. Table 5 illustrates the final results. It can be seen from the table that the setting of ReLU6 + *hard-swish* beats the other two kinds of settings in both PokerNet-0.5× and PokerNet-1.0×. Compared with the models with ReLU, *hard-swish* version achieves 0.4% and 0.6% accuracy improvement respectively. It proves the effectiveness of *hard-swish* nonlinearities in lightweight deep models.

5 Conclusions

Pointwise convolutions usually occupy most of the

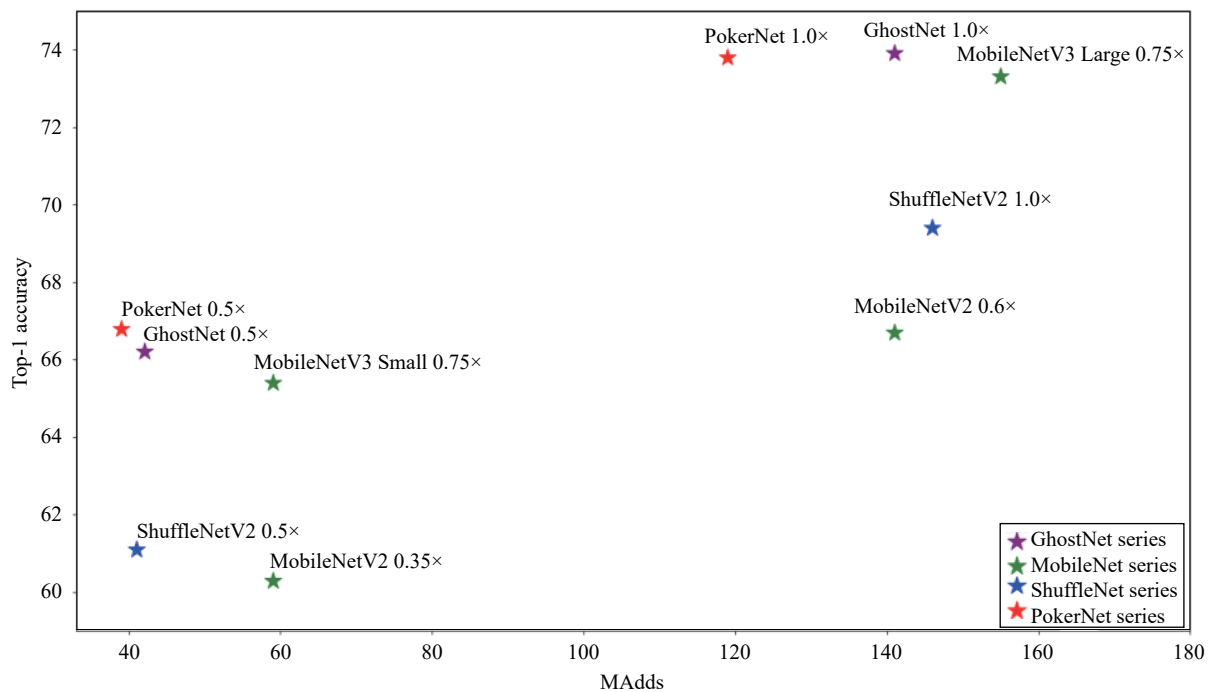


Fig. 6 Computation cost and recognition accuracy comparison between PokerNets and previous SOTA models

Table 4 Ablation study of Squeeze-and-excite layer in our proposed Poker module

Models	SE	Params (M)	MAdds (M)	Top-1 Acc. (%)	Top-5 Acc. (%)
PokerNet 0.5×	Yes	2.39	39.4	66.8	87.0
PokerNet 0.5×	No	1.81	39.0	63.5	84.4
PokerNet 1.0×	Yes	5.70	119	73.6	91.5
PokerNet 1.0×	No	3.41	117	70.6	89.5

Table 5 Ablation study of nonlinearities in our proposed Poker module

Models	Act	Params (M)	MAdds (M)	Top-1 Acc. (%)	Top-5 Acc. (%)
PokerNet 0.5×	ReLU	2.39	39.4	66.4	86.2
PokerNet 0.5×	ReLU6	2.39	39.4	66.5	86.5
PokerNet 0.5×	ReLU6+HS	2.39	39.4	66.8	87.0
PokerNet 1.0×	ReLU	5.70	119	73.0	90.4
PokerNet 1.0×	ReLU6	5.70	119	73.2	90.8
PokerNet 1.0×	ReLU6+HS	5.70	119	73.6	91.5

computational cost in lightweight deep models. For building more efficient deep models, this paper presents a novel Poker module to reduce the usage of pointwise convolutions. The proposed Poker module utilizes groups of cheap depthwise convolutions instead of pointwise convolutions to expand the features. SE layer is adopted in Poker module to enhance or suppress the expanded features in a self-attention manner. Experiments conducted on benchmark models and datasets prove that the proposed module is standardized and can be applied to any place where feature expansion is needed. Moreover, PokerNets built with the proposed Poker module beat the SOTA lightweight models in both the computational cost and the recognition accuracy.

In the future, we would like to apply our proposed PokerNets to other computer vision tasks such as object detection and image segmentation. As for the model itself, we would like to explore the customized expansion factor k and the self-attention manner to construct more efficient lightweight deep models while boosting the performance.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Nos. 61525306, 61633021, 61721004, 61806194, U1803261 and 61976132), Major Project for New Generation of AI (No. 2018AAA0100400), Beijing Nova Program (No. Z201100006820079), Shandong Provincial Key Research and Development Program (No. 2019JZZY010119), and CAS-AIR.

References

- [1] A. Krizhevsky, I. Sutskever, G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS, Lake Tahoe, USA, pp.1097–1105, 2012. DOI: [10.5555/2999134.2999257](https://doi.org/10.5555/2999134.2999257).
- [2] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. [Online], Available: <https://arxiv.org/abs/1409.1556>, 2015.
- [3] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, USA, pp.770–778, 2016. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [4] R. Girshick. Fast R-CNN. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Santiago, Chile, pp.1440–1448, 2015. DOI: [10.1109/ICCV.2015.169](https://doi.org/10.1109/ICCV.2015.169).
- [5] K. M. He, G. Gkioxari, P. Dollár, R. Girshick. Mask R-CNN. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp.2980–2988, 2017. DOI: [10.1109/ICCV.2017.322](https://doi.org/10.1109/ICCV.2017.322).
- [6] T. Y. Lin, P. Dollár, R. Girshick, K. M. He, B. Hariharan, S. Belongie. Feature pyramid networks for object detection. In *Proceedings of IEEE Conference On computer Vision and Pattern Recognition*, IEEE, Honolulu, USA, pp.936–944, 2017. DOI: [10.1109/CVPR.2017.106](https://doi.org/10.1109/CVPR.2017.106).
- [7] J. Long, E. Shelhamer, T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, USA, pp.3431–3440, 2015. DOI: [10.1109/CVPR.2015.7298965](https://doi.org/10.1109/CVPR.2015.7298965).
- [8] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. [Online], Available: <https://arxiv.org/abs/1412.7062>, 2016.
- [9] W. Y. Chen, X. Y. Gong, X. M. Liu, Q. Zhang, Y. Li, Z. Y. Wang. FasterSeg: Searching for faster real-time semantic segmentation. In *Proceedings of the 8th International Conference on Learning Representations*, OpenReview.net, Addis Ababa, Ethiopia, 2020.
- [10] H. Li, A. Kadav, I. Durdanovic, H. Samet, H. P. Graf. Pruning filters for efficient convNets. In *Proceedings of the 5th International Conference on Learning Representations*, OpenReview.net, Toulon, France, 2017.
- [11] Y. H. He, X. Y. Zhang, J. Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of IEEE International Conference on Computer Vision*, IEEE, Venice, Italy, pp.1398–1406, 2017. DOI: [10.1109/ICCV.2017.155](https://doi.org/10.1109/ICCV.2017.155).
- [12] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, Y. Bengio. Binarized neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp.4114–4122, 2016. DOI: [10.5555/3157382.3157557](https://doi.org/10.5555/3157382.3157557).
- [13] W. Tang, G. Hua, L. Wang. How to train a compact binary neural network with high accuracy? In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, San Francisco, USA, pp.2625–2631, 2017. DOI: [10.5555/3298483.3298617](https://doi.org/10.5555/3298483.3298617).
- [14] A. G. Howard, M. L. Zhu, B. Chen, D. Kalenichenko, W. J. Wang, T. Weyand, M. Andreetto, H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. [Online], Available: <https://arxiv.org/abs/1704.14861>, 2017.
- [15] X. Y. Zhang, X. Y. Zhou, M. X. Lin, J. Sun. Shufflenet: An extremely efficient convolutional neural network for mo-

- mobile devices. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp.6848–6856, 2018. DOI: [10.1109/CVPR.2018.00716](https://doi.org/10.1109/CVPR.2018.00716).
- [16] G. Hinton, O. Vinyals, J. Dean. Distilling the knowledge in a neural network. [Online], Available: <https://arxiv.org/abs/1503.02531>, 2015.
- [17] S. You, C. Xu, C. Xu, D. C. Tao. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, Halifax, Canada, pp. 1285–1294, 2017. DOI: [10.1145/3097983.3098135](https://doi.org/10.1145/3097983.3098135).
- [18] H. T. Chen, Y. H. Wang, C. Xu, Z. H. Yang, C. J. Liu, B. X. Shi, C. J. Xu, C. Xu, Q. Tian. Data-free learning of student networks. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Seoul, Korea, pp. 3513–3521, 2019. DOI: [10.1109/ICCV.2019.00361](https://doi.org/10.1109/ICCV.2019.00361).
- [19] K. Han, Y. H. Wang, Q. Tian, J. Y. Guo, C. J. Xu, C. Xu. Ghostnet: More features from cheap operations. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 1577–1586, 2020. DOI: [10.1109/CVPR42600.2020.00165](https://doi.org/10.1109/CVPR42600.2020.00165).
- [20] M. Sandler, A. Howard, M. L. Zhu, A. Zhmoginov, L. C. Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 4510–4520, 2018. DOI: [10.1109/CVPR.2018.00474](https://doi.org/10.1109/CVPR.2018.00474).
- [21] A. Howard, M. Sandler, B. Chen, W. J. Wang, L. C. Chen, M. X. Tan, G. Chu, V. Vasudevan, Y. K. Zhu, R. M. Pang, H. Adam, Q. Le. Searching for mobileNetV3. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Seoul, Korea, pp. 1314–1324, 2019. DOI: [10.1109/ICCV.2019.00140](https://doi.org/10.1109/ICCV.2019.00140).
- [22] N. N. Ma, X. Y. Zhang, H. T. Zheng, J. Sun. ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 122–138, 2018. DOI: [10.1007/978-3-030-01264-9_8](https://doi.org/10.1007/978-3-030-01264-9_8).
- [23] M. Lin, Q. Chen, S. C. Yan. Network in network. [Online], Available: <https://arxiv.org/abs/1312.4400>, 2014.
- [24] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 MB model size. [Online], Available: <https://arxiv.org/abs/1602.07360>, 2016.
- [25] B. C. Wu, A. Wan, X. Y. Yue, P. Jin, S. C. Zhao, N. Goltant, A. Gholaminejad, J. Gonzalez, K. Keutzer. Shift: A zero FLOP, zero parameter alternative to spatial convolutions. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 9127–9135, 2018. DOI: [10.1109/CVPR.2018.00951](https://doi.org/10.1109/CVPR.2018.00951).
- [26] H. Cai, L. G. Zhu, S. Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *Proceedings of the 7th International Conference on Learning Representations*, OpenReview.net, New Orleans, USA, 2019.
- [27] M. X. Tan, B. Chen, R. M. Pang, V. Vasudevan, M. Sandler, A. Howard, Q. V. Le. MnasNet: Platform-aware neural architecture search for mobile. In *Proceedings IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 2815–2823, 2019. DOI: [10.1109/CVPR.2019.00293](https://doi.org/10.1109/CVPR.2019.00293).
- [28] C. X. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L. J. Li, F. F. Li, A. Yuille, J. Huang, K. Murphy. Progressive neural architecture search. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 19–35, 2018. DOI: [10.1007/978-3-030-01246-5_2](https://doi.org/10.1007/978-3-030-01246-5_2).
- [29] B. Zoph, Q. V. Le. Neural architecture search with reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, OpenReview.net, Toulon, France, 2017.
- [30] B. Baker, O. Gupta, N. Naik, R. Raskar. Designing neural network architectures using reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, OpenReview.net, Toulon, France, 2017.
- [31] B. Zoph, V. Vasudevan, J. Shlens, Q. V. Le. Learning transferable architectures for scalable image recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 8697–8710, 2018. DOI: [10.1109/CVPR.2018.00907](https://doi.org/10.1109/CVPR.2018.00907).
- [32] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, J. Dean. Efficient neural architecture search via parameters sharing. In *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, pp. 4095–4104, 2018.
- [33] H. X. Liu, K. Simonyan, Y. M. Yang. Darts: Differentiable architecture search. In *Proceedings of the 7th International Conference on Learning Representations*, OpenReview.net, New Orleans, USA, 2019.
- [34] B. C. Wu, X. L. Dai, P. Z. Zhang, Y. H. Wang, F. Sun, Y. M. Wu, Y. D. Tian, P. Vajda, Y. Q. Jia, K. Keutzer. FbNet: Hardware-aware efficient convNet design via differentiable neural architecture search. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 10726–10734, 2019. DOI: [10.1109/CVPR.2019.01099](https://doi.org/10.1109/CVPR.2019.01099).
- [35] Y. H. He, J. Lin, Z. J. Liu, H. R. Wang, L. J. Li, S. Han. AMC: AutoML for model compression and acceleration on mobile devices. In *Proceedings of the 15th European Conference on Computer Vision*, Springer, Munich, Germany, pp. 815–832, 2018. DOI: [10.1007/978-3-030-01234-2_48](https://doi.org/10.1007/978-3-030-01234-2_48).
- [36] X. L. Dai, P. Z. Zhang, B. C. Wu, H. X. Yin, F. Sun, Y. H. Wang, M. Dukhan, Y. Q. Hu, Y. M. Wu, Y. Q. Jia, P. Vajda, M. Uyttendaele, N. K. Jha. ChamNet: Towards efficient network design through platform-aware model adaptation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, USA, pp. 11390–11399, 2019. DOI: [10.1109/CVPR.2019.01166](https://doi.org/10.1109/CVPR.2019.01166).
- [37] A. Wan, X. L. Dai, P. Z. Zhang, Z. J. He, Y. D. Tian, S. N. Xie, B. C. Wu, M. Yu, T. Xu, K. Chen, P. Vajda, J. E. Gonzalez. FbNetV2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, USA, pp. 12962–12971, 2020. DOI: [10.1109/CVPR42600.2020.01298](https://doi.org/10.1109/CVPR42600.2020.01298).
- [38] X. L. Dai, A. Wan, P. Z. Zhang, B. C. Wu, Z. J. He, Z. Wei, K. Chen, Y. D. Tian, M. Yu, P. Vajda, J. E. Gonzalez. FBNetV3: Joint architecture-recipe search using neural acquisition function. [Online], Available: <https://arxiv.org/abs/2006.02049>, 2020.
- [39] M. Z. Shen, K. Han, C. J. Xu, Y. H. Wang. Searching for accurate binary neural architectures. In *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision Workshop*, IEEE, Seoul, Korea, pp. 2041–2044, 2019. DOI: [10.1109/ICCVW.2019.00256](https://doi.org/10.1109/ICCVW.2019.00256).
- [40] S. Han, H. Z. Mao, W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. [Online], Available: <https://arxiv.org/abs/1510.00149>, 2016.
- [41] S. P. Gui, H. N. Wang, H. C. Yang, C. Yu, Z. Y. Wang, J.

- Liu. Model compression with adversarial robustness: A unified optimization framework. In *Proceedings of Advances in Neural Information Processing Systems*, NeurIPS, Vancouver, Canada, pp. 1283–1294, 2019.
- [42] J. H. Luo, J. X. Wu, W. Y. Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of IEEE International Conference On Computer Vision*, IEEE, Venice, Italy, pp. 5068–5076, 2017. DOI: [10.1109/ICCV.2017.541](https://doi.org/10.1109/ICCV.2017.541).
- [43] C. J. Liu, Y. H. Wang, K. Han, C. J. Xu, C. Xu. Learning instance-wise sparsity for accelerating deep models. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, China, pp. 3001–3007, 2019.
- [44] W. Wen, C. P. Wu, Y. D. Wang, Y. R. Chen, H. Li. Learning structured sparsity in deep neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 2082–2090, 2016. DOI: [10.5555/3157096.3157329](https://doi.org/10.5555/3157096.3157329).
- [45] Z. C. Liu, H. Y. Mu, X. Y. Zhang, Z. C. Guo, X. Yang, K. T. Cheng, J. Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, IEEE, Seoul, Korea, pp. 3295–3304, 2019. DOI: [10.1109/ICCV.2019.00339](https://doi.org/10.1109/ICCV.2019.00339).
- [46] J. Hu, L. Shen, G. Sun. Squeeze-and-excitation networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, USA, pp. 7132–7141, 2018. DOI: [10.1109/CVPR.2018.00745](https://doi.org/10.1109/CVPR.2018.00745).
- [47] S. Ioffe, C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, pp. 448–456, 2015.
- [48] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, F. F. Li. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Miami, USA, pp. 248–255, 2009. DOI: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [49] X. B. Fu, S. L. Yue, D. Y. Pan. Camera-based basketball scoring detection using convolutional neural network. *International Journal of Automation and Computing* vol. 18, no. 2, pp. 266–276, 2018. DOI: [10.1007/s11633-020-1259-7](https://doi.org/10.1007/s11633-020-1259-7).
- [50] K. Aukkapinyo, S. Sawangwong, P. Pooyoi, W. Kusakunniran. Localization and classification of rice-grain images using region proposals-based convolutional neural network. *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 233–246, 2020. DOI: [10.1007/s11633-019-1207-6](https://doi.org/10.1007/s11633-019-1207-6).



Wei Tang received the B.Sc. degree in automation from Harbin Engineering University (HEU), China in 2013. Currently, he is a Ph.D. degree candidate in National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), China under the guidance of professor Liang Wang. He has published papers in the conferences

such as *AAAI Conference on Artificial Intelligence (AAAI)*, *Chinese Conference on Computer Vision (CCCV)*. He has won

the Best Student Paper Award in CCCV 2015 and the Star of Tomorrow Award in internship of Microsoft Research Asian (MSRA).

His research interests include deep learning and computer vision, model compression and acceleration.

E-mail: tangweirichard@163.com

ORCID iD: 0000-0001-5784-2995

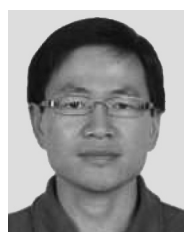


Yan Huang received the B.Sc. degree in information and computing science from University of Electronic Science and Technology of China (UESTC), China in 2012, and the Ph.D. degree in pattern recognition and intelligent systems from University of Chinese Academy of Sciences (UCAS), China in 2017. Since July 2017, He has joined the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), China as associate researcher. He has published more than 50 papers in international journals and conferences in related fields. Related papers have won the CVPR Workshop Best Paper Award, *IEEE Conference on Computer Vision and Pattern Recognition (ICPR)* Best Student Paper Award, etc. He was selected in Beijing Science and Technology Star Program and Microsoft Star Casting Program. He was the Co-chair of multimodal symposiums on *International Conference on Pattern Recognition (CVPR)* and *International Conference on Computer Vision (ICCV)*. He has won the special award of president of Chinese Academy of Sciences, Excellent Doctoral Dissertation Award of Chinese society of artificial intelligence, Baidu scholarship and NVIDIA Innovation Research Award.

His research interests include computer vision and multimodal data analysis.

E-mail: yhuang@nlpr.ia.ac.cn

ORCID iD: 0000-0002-8239-7229



Liang Wang received the B.Eng. degree in radio technology and M.Eng. degree in circuit system from Anhui University, China in 1997 and 2000, respectively, and the Ph.D. degree in pattern recognition and intelligent systems from Institute of Automation, Chinese Academy of Sciences (CASIA), China in 2004. From 2004 to 2010, he was a research assistant at Imperial College London, UK and Monash University, Australia, a research fellow at the University of Melbourne, Australia, and a lecturer at the University of Bath, UK, respectively. Currently, he is a full professor of the Hundred Talents Program at the National Lab of Pattern Recognition, CASIA. He has widely published in highly ranked international journals such as *IEEE Transactions on Pattern Analysis and Machine Intelligence* and *IEEE Transactions on Image Processing*, and leading international conferences such as CVPR, ICCV, and *International Conference on Data Mining (ICDM)*. He is a Senior Member of the IEEE, and an IAPR Fellow.

His research interests include machine learning, pattern recognition, and computer vision.

E-mail: wangliang@nlpr.ia.ac.cn (Corresponding author)

ORCID iD: 0000-0001-5224-8647