# Neural Architecture Search for Convolutional Neural Networks with Attention

Nakai, Kohei

Matsubara, Takashi

Uehara, Kuniaki

PAPER

# Neural Architecture Search for Convolutional Neural Networks with Attention

**Kohei NAKAI**[†a)], ***Nonmember***, **Takashi MATSUBARA**[††b)], ***and*** **Kuniaki UEHARA**[†††c)], ***Members***

**SUMMARY** The recent development of neural architecture search (NAS) has enabled us to automatically discover architectures of neural networks with high performance within a few days. Convolutional neural networks extract fruitful features by repeatedly applying standard operations (convolutions and poolings). However, these operations also extract useless or even disturbing features. Attention mechanisms enable neural networks to discard information of no interest, having achieved the state-of-the-art performance. While a variety of attentions for CNNs have been proposed, current NAS methods have paid a little attention to them. In this study, we propose a novel NAS method that searches attentions as well as operations. We examined several patterns to arrange attentions and operations, and found that attentions work better when they have their own search space and follow operations. We demonstrate the superior performance of our method in experiments on CIFAR-10, CIFAR-100, and ImageNet datasets. The found architecture achieved lower classification error rates and required fewer parameters compared to those found by current NAS methods.

***key words:*** *neural architecture search, object recognition, attention mechanism*

## 1. Introduction

Deep learning has enabled us to solve various tasks with high performance thanks to its flexibility and rich expressive ability. The flexibility and expressive ability come from its architecture designed for targeted tasks. For instance, convolutional neural networks (CNNs) and their extensions are commonly used for computer vision tasks [1]. To build a neural architecture, one has to design it manually by following a trial-and-error approach. Manually designing architectures requires a considerable amount of expertise and time to ensure that they achieve state-of-the-art performances.

The goal of neural architecture search (NAS) is to automate this time-consuming and error-prone process [2]. The recent development of NAS has enabled us to automatically design an architecture that achieves good performance in tasks such as semantic image segmentation [3] and person re-IDentification [4]. A notable difficulty of NAS is ensuring the discreteness of the search space over operations such as convolution and pooling. A selection of one operation among several candidate operations needs to be a discrete and indifferentiable action. Therefore, many of the earliest NAS works used reinforcement learning and evolutionary algorithms [5]–[8].

These methods trained a new candidate architecture from scratch and compared its performance with existing ones; however, these processes are computationally expensive and time-consuming, often requiring thousands of GPU days. ENAS reduced search time to within a GPU day by sharing parameters among all candidate architectures [9]. A breakthrough was led by gradient-based approaches such as DARTS, which builds a neural network as a weighted sum of all candidate operations and trains the relative weights among the candidates and their internal parameters [10]. Compared with ENAS, DARTS achieved better results with slightly longer search times.

Of the three dimensions for NAS: search space, search strategy, and performance estimation strategy [2], most earlier NAS works focus on the search strategy. In this study, our focus is on the search space. For computer vision tasks, we use CNNs, whose main operation is convolution. A neural network can capture the spatial and channel-wise features of an image by applying convolution. However, the neural network considers all spatial and channel-wise features including the ones that are not useful, when using only convolutions. To discard such useless features and focus on important features, attentions have been proposed [11]–[17]. Studies that employed CNNs with attentions reported better performances with a comparable number of parameters. We consider that attentions are promising candidates for neural architectures and can be employed in the search space.

In this work, we propose a novel search method called Att-DARTS that can find architectures using attentions. Att-DARTS assumes a CNN composed of repeatedly stacked cells similar to most existing NAS works; however, it inserts an attention after each operation. During the search process, Att-DARTS optimizes the relative weights of both operations and attentions in addition to internal parameters using gradient descent like DARTS [10]. After the search process, Att-DARTS chooses the operations and attentions with the strongest relative weights and builds a final cell architecture.

We searched the architecture with attentions on an image dataset, CIFAR-10, and evaluated the identified cells on CIFAR-10, CIFAR-100, and ImageNet [18], [19]. In

Manuscript received May 21, 2020.
Manuscript revised September 15, 2020.
Manuscript publicized October 26, 2020.

[†]The author is with Graduate School of System Informatics, Kobe University, Kobe-shi, 657–8501 Japan.
[††]The author is with Graduate School of Engineering Science, Osaka University, Toyonaka-shi, 560–8531 Japan.
[†††]The author is with Faculty of Business Administration, Osaka Gakuin University, Suita-shi, 564–8511 Japan.
a) E-mail: k.nakai@ai.cs.kobe-u.ac.jp
b) E-mail: matsubara@sys.es.osaka-u.ac.jp
c) E-mail: kuniaki.uehara@ogu.ac.jp

DOI: 10.1587/transinf.2020EDP7111

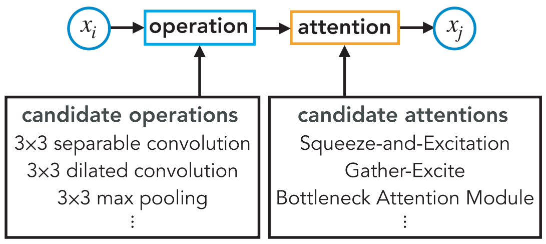


**Fig. 1** An overview of Att-DARTS. We propose applying not only a standard operation (convolution or pooling) but also an attention after the operation is applied. Att-DARTS chooses both operations and attentions from the respective candidate space.

our experiments, we found that Att-DARTS found a cell that achieves a 2.54% test error on CIFAR-10 (around 10% lower than that of DARTS) and a 16.54% test error rate on CIFAR-100. On ImageNet, the cell achieved a 26.0% top-1 error rate and an 8.5% top-5 error rate, both are better than that of DARTS. Simultaneously, Att-DARTS reduces the number of parameters by 0.1M compared to DARTS.

The remainder of this paper is organized as follows. Section 2 is devoted to introducing related works on neural architecture search and attentions. Section 3 elaborates on our proposed method, Att-DARTS. Section 4 provides details on the experimental results on CIFAR-10, CIFAR-100 and ImageNet results. In Sect. 5, our experiments demonstrate that Att-DARTS outperforms DARTS and several other comparative methods. Additional experiments show that attentions work best when they are put after every operation. Finally, Sect. 6 provides the conclusion of the paper. Limited and preliminary results can be found in conference proceedings [20]. Compared to the conference proceedings, we additionally evaluate the searched cells with an additional dataset, ImageNet, in order to confirm the transferability of Att-DARTS in Sect. 5.2. Further, we evaluate possible alternatives to find the best location where attentions are inserted in Sect. 5.3.

The source code to reproduce the results of this study can be found at `https://github.com/chomin/Att-DARTS`.

## 2. Related Work

### 2.1 Neural Architecture Search (NAS)

NAS aims to automate the design of neural network architectures and to reduce experts' efforts. Early NAS works are based on reinforcement learning or evolutionary algorithms [5]–[8]. Zoph *et al.* [5] generated optimal hyperparameters of each convolutional layer (e.g., filter size, and stride) and the number of layers using a controller recurrent neural network (RNN). The RNN is trained with reinforcement learning to maximize the accuracy of a child network. Other studies employed evolutionary algorithms such as AmoebaNet [7].

These methods train a new candidate architecture from scratch and compare it with the previously generated ones; these processes are computationally expensive and time-consuming. Some works reduced the search time by using a one-shot model, which is a single model composed of all candidate operations [9], [21], [22]. The model facilitated sharing weight parameters among all candidate architectures. At the search stage, we only need to train this single model instead of all candidate models from scratch. Instead of designing an entire neural network architecture, NASNet [6] designed an architecture composed of repeated small networks, each of which is called a cell, and this helped to drastically reduce the search space.

Many of these methods used reinforcement learning or evolutionary algorithms to design a neural network architecture topology, whereas they train a designed neural network using the gradient descent method. Hence, architecture search and parameter optimization can be performed alternatively, which increases computational time. To bridge this gap, gradient-based methods such as DARTS [10] have been proposed.

Gradient-based methods make the operation search space continuous by implementing an operation as a weighted sum of all candidate operations. DARTS jointly trains the relative weights of the operations and their internal parameters using the gradient descent method, and it chooses the cells with the strongest relative weights. The best cells found by DARTS achieved results comparable with those found by methods based on reinforcement learning or evolutionary algorithms.

Various methods inspired by DARTS have been proposed. To reduce memory consumption during the search space, DARTS employed a shallower network architecture at the search stage than at the evaluation stage. P-DARTS [23] bridged this depth gap by reducing the pattern of candidate operations step-by-step. PC-DARTS [24] proposed a partial channel connection and improved memory efficiency. Some works focus on the issue that DARTS tends to choose many skip connections [23], [25], [26]. However, these studies focused on the search strategy. In contrast, our study focuses on the search space itself.

### 2.2 Attention Modules

Convolutional neural networks have achieved significant results in image classification tasks. The expressive power of CNNs increases with the network size, and many studies have been conducted to investigate a trainable deeper and wider architecture [1]. The main operation of CNNs is convolution, and by repeating convolutions, CNNs can extract a larger amount of spatial and channel-wise features, some of which are often useless or conflict with important features. Attentions have been proposed to discard such features and focus on the important one. Moreover, attentions help CNNs achieve better performance and make them robust against disturbances [11]–[17].

*Wang et al.* [11] proposed a residual attention network, which is composed of trunk and mask branches. The trunk branch extracts features similar to ordinary convolutions.

The mask branch builds an attention mask and multiplies the output of the trunk branch. Many following studies proposed attentions that focus on the mask and are used after a convolution operation. Squeeze-and-excitation (SE) [12] proposed a lighter attention, which has channel attention that enables the network to focus on important channels. Gather-excite (GE) [13] added a depth-wise convolution to SE and increased its flexibility. BAM [14] and CBAM [15] have spatial attention and channel attention. BAM builds a mask as a sum of its channel and spatial attentions, whereas CBAM sequentially applies channel and spatial attentions. $A^2$-Net [16] has a bilinear pooling operation to reflect the different needs in different locations.

Previous studies demonstrated that the same attention contributes to a variety of image processing tasks and a variety of data domains [12], [15], [17], [27]. For example, SE was confirmed to contribute to image classification tasks on ImageNet, CIFAR-10, and CIFAR-100 datasets, to scene classification task on Places365-Challenge dataset, and to object detection task on the MS COCO dataset [12]. SE is also applicable to medical image segmentation [27]. The task dependency of the attention modules is at a similar level to other operations, and we have confirmed that the cell found with the CIFAR-10 dataset is transferable to the CIFAR-100 and ImageNet datasets.

## 3. Method

### 3.1 Architecture Search with Attention Modules

In this section, we propose Att-DARTS, a method that searches for a neural architecture using attentions, as illustrated in Fig. 2. As with many NAS works, the entire network consists of repeated cells, and Att-DARTS searches the good cells. Each cell $c_k$ is expressed as a directed acyclic graph with $N$ nodes, two of which are inputs and one is the output. The remaining $N-3$ nodes are intermediate nodes. Each node $x_i$ is a feature map. The input nodes are obtained from the output nodes of the previous two cells $c_{k-1}$ and $c_{k-2}$. The output node is defined as the depth-wise concatenation of all intermediate nodes in the cell.

Att-DARTS searches the cell including attentions as

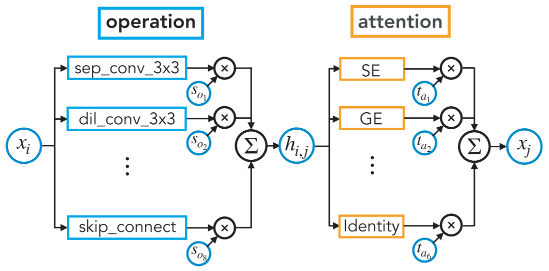


**Fig. 2** Illustration of Att-DARTS. This figure shows the edge from node $x_i$ to node $x_j$ via intermediate node $h_{i,j}$. Node $h_{i,j}$ is computed as the weighted sum of all candidate operations from $x_i$ to $x_j$, and node $x_j$ is computed as the weighted sum of all candidate attentions from node $h_{i,j}$. We provide the details in Sect. 3.

well as operations. Original studies on attentions proposed several locations to insert attentions; however, the most popular location is immediately after a convolution operation (e.g., see [12], [15]). Following this approach, Att-DARTS assumes an edge composed of an operation preceding an attention.

At the search stage, Att-DARTS identifies an operation in the operation space $\mathcal{O}$, and an attention in the attention space $\mathcal{A}$. When the focus is on the edge from node $x_i$ to node $x_j$, each candidate operation $o(\cdot) \in \mathcal{O}$ has a relative weight $s_o^{(i,j)}$, which is defined by the softmax operation over the operation parameters $\alpha_o^{(i,j)}$ of all candidate operations:

$$s_o^{(i,j)} = \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in \mathcal{O}} \exp(\alpha_{o'}^{(i,j)})}. \tag{1}$$

Each candidate attention $a(\cdot) \in \mathcal{A}$ has a relative weight $t_a^{(i,j)}$, which is also defined by the softmax operation over the attention parameters $\beta_a^{(i,j)}$ of all candidate attentions:

$$t_a^{(i,j)} = \frac{\exp(\beta_a^{(i,j)})}{\sum_{a' \in \mathcal{A}} \exp(\beta_{a'}^{(i,j)})}. \tag{2}$$

We introduce a new node $h_{i,j}$ on the edge from node $x_i$ to node $x_j$. Similarly to DARTS, the information flow from node $x_i$ to node $h_{i,j}$ is computed as the weighted sum of all candidate operations:

$$h_{i,j}(x_i) = \sum_{o \in \mathcal{O}} s_o^{(i,j)} o(x_i). \tag{3}$$

The information flow from $h_{i,j}$ to $x_j$ is computed as the weighted sum of all candidate attentions:

$$\bar{a}^{(i,j)}(h_{i,j}) = \sum_{a \in \mathcal{A}} t_a^{(i,j)} a(h_{i,j}) \tag{4}$$

Node $x_j$ is computed as the sum of all flows from source nodes $h_{i,j}$ in the cell:

$$x_j = \sum_{i<j} \bar{a}^{(i,j)}(h_{i,j}) \tag{5}$$

Then, we learn the operation parameters $\alpha$, attention parameters $\beta$, and weight parameters $w$ (e.g., convolution kernels) simultaneously, using the gradient descent by solving the bilevel optimization problem, where the operation parameters $\alpha$ and the attention parameters $\beta$ are the upper-level variables and the weight parameters $w$ are the lower-level variables.

$$\min_{\alpha,\beta} \mathcal{L}_{val}(w^*(\alpha,\beta),\alpha,\beta) \tag{6}$$

$$s.t.\ w^*(\alpha,\beta) = \operatorname*{argmin}_{w} \mathcal{L}_{train}(w,\alpha,\beta), \tag{7}$$

with $\mathcal{L}_{train}$ and $\mathcal{L}_{val}$ denoting the training and validation losses, respectively.

After the search stage, Att-DARTS chooses the operation $o(\cdot)$ for each edge from node $x_i$ to $x_j$ with the strongest relative weight $s_o^{(i,j)}$. Furthermore, Att-DARTS chooses attention $a(\cdot)$ for each edge with the strongest relative weight $t_a^{(i,j)}$. Then, Att-DARTS chooses two edges with the strongest operation weights $s_o^{(i,j)}$ from all edges connected to node $x_j$. Att-DARTS maintains $2 \times (N-3)$ operations and attentions in a cell. The information flow to node $x_j$ is defined as

$$x_j = \sum_{(o,a,x_i)\in C_j} a^{(i,j)}(o^{(i,j)}(x_i)), \tag{8}$$

where $C_j$ is the set of chosen edges connected to node $x_j$ including their operations and attentions.

## 4. Experiments

### 4.1 Experimental Details

We conducted the neural architecture search on CIFAR-10 and evaluated it on CIFAR-10, CIFAR-100 and ImageNet [18], [19]. CIFAR-10 consists of 10 class RGB images, and CIFAR-100 consists of 100 class RGB images. Both datasets contain 50,000 training images and 10,000 test images, with every image having a resolution of $32 \times 32$ pixels. We used ILSVRC2012 [28], a subset of ImageNet [19]. ILSVRC2012 consists of 1,000 class RGB images. ILSVRC2012 has 1.2 million training images and 50,000 validation images. We fixed the input image size to $224 \times 224$ pixels.

At the search stage, we searched for cells. We followed the same experimental setting as that of the original study on DARTS unless otherwise stated [10]. We independently executed Att-DARTS 4 times with different random seeds for 50 epochs with batch size 64. We randomly split 50,000 training images into training and validation sets of equal sizes. We set the number of cells $L = 8$; two reduction cells and six normal cells. The reduction cells were inserted into the 1/3 and 2/3 locations of the entire network. Operations were with stride 1 in the normal cells and with stride 2 in the reduction cells; hence, the image size was halved at the reduction cells. Each cell consisted of $N = 7$ nodes. The initial number of channel was set to 16, and it was doubled at the reduction cells.

The original study on DARTS proposed the first- and second-order optimizations [10]. We employed the second-order optimization to pursue a better performance. In the same way as the operation parameters $\alpha$ of DARTS, Att-DARTS updates the attention parameters $\beta$ by minimizing the objective function;

$$\nabla_{\alpha,\beta}\mathcal{L}_{val}(w^*(\alpha,\beta),\alpha,\beta) \tag{9}$$
$$\approx \nabla_{\alpha,\beta}\mathcal{L}_{val}(w - \xi\nabla_w\mathcal{L}_{train}(w,\alpha,\beta),\alpha,\beta). \tag{10}$$

We used zero initialization for both architecture and attention parameters. We used momentum SGD to optimize weight parameters $w$ with an initial learning rate 0.025, momentum 0.9, and weight decay $3.0 \times 10^{-4}$. The learning rate was annealed down to zero following a cosine schedule [29]. We used the Adam optimizer [30] for the operation parameters $\alpha$ and attention parameters $\beta$ with an initial learning rate $3.0 \times 10^{-4}$, momentum $(0.5, 0.999)$, and weight decay $1.0 \times 10^{-3}$.

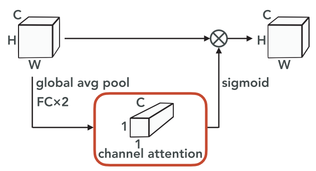
After the search stage, we obtained candidate cells for each iteration by choosing operations and attentions according to the operation parameters $\alpha$ and attention parameters $\beta$. We built a neural network composed of candidate cells from scratch and trained it with the same experimental setting as the evaluation stage. Then, we chose the best cells among all four runs based on the best validation accuracy within the 600 epochs, while DARTS chose the best cells based on the validation accuracy at the 100th epoch.

The evaluation is performed as follows. For CIFAR-10, we set the number of cells $L = 20$; 2 reduction cells and 18 normal cells. The initial number of channel was set to 36, and doubled at the reduction cells. We then applied cutout [31], path dropout [32] of probability 0.2, and auxiliary towers [33] of weight 0.4. We independently trained the best cells 5 times for 600 epochs with batch size 96, and then reported the mean accuracy and standard deviation. For ImageNet, we set the number of cells $L = 14$; 2 reduction cells and 12 normal cells. The initial number of channel was set to 48, and doubled at the reduction cells. We then applied auxiliary towers [33] of weight 0.4. We trained the best cells for 250 epochs with batch size 128.

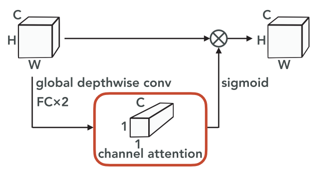
The experimental design we employed was the same as that of the DARTS paper [10]. Namely, the best cells were searched over the CIFAR-10 dataset. Then, the obtained cells were evaluated on the CIFAR-10, CIFAR-100, and ImageNet datasets, where the parameters of the found cell were trained from scratch, i.e., the architecture is transferred to other datasets but the parameters are not. Of course, the best cell can depend on a given task; a NAS method can find a cell that works better for the CIFAR-100 dataset when searching with the CIFAR-100 dataset. The experimental design aims not to find the universally best cell but to confirm the transferability of the found cell. The transferability is important for a classification task of a small-sized dataset and for an unsupervised clustering task where the cell is hard to search. Many NAS studies have also evaluated their transferability [6], [8].

### 4.2 Architecture Search Space

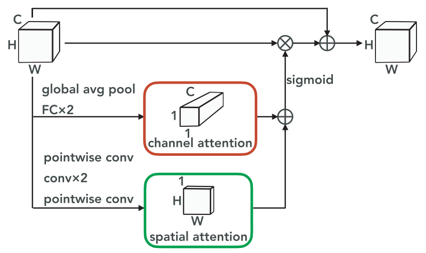
The operation space $\mathcal{O}$ was the same as that of DARTS: Identity, $3 \times 3$ and $5 \times 5$ separable convolutions, $3 \times 3$ and $5 \times 5$ dilated separable convolutions, $3 \times 3$ max pooling, $3 \times 3$ average pooling, and *zero*. *Zero* indicates the absence of a connection and it is excluded from the chosen operations. A depth-wise separable convolution [34], [35] is the computational- and memory-efficient version of a convolution. It splits the convolution into two layers: depth-wise convolution and point-wise convolution. The depth-wise convolution is applied to each channel independently; the pointwise convolution is a convolution with a kernel size of 1. A dilated convolution [36] is a convolution whose kernel



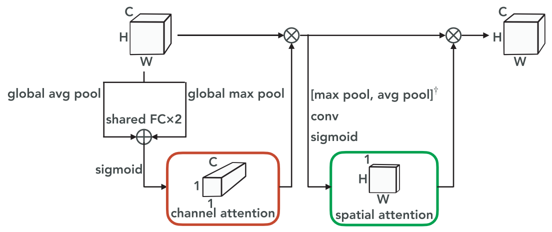
(a) Squeeze-and-Excitation (SE).



(b) Gather-Excite (GE-$\theta^+$).

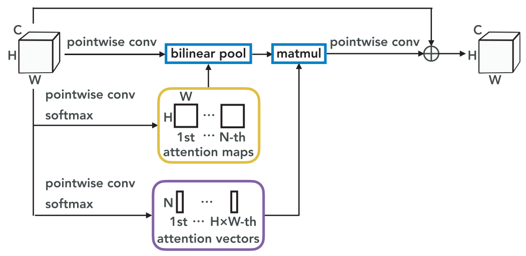


(c) Bottleneck Attention Module (BAM).



†[, ] means concatenation.
Both pooling operations and the concatenation are along the channel axis.

(d) Convolutional Block Attention Module (CBAM).



(e) Double-attention block ($A^2$-block).

**Fig. 3** Overview of each attention. ⊗ and ⊕ mean element-wise multiplication and element-wise summation, respectively.

is applied to every $l$ space points ($l$ is a dilation factor).

The attention space $\mathcal{A}$ includes Identity, SE [12], GE-$\theta^+$ [13], BAM [14], CBAM [15], and double-attention block ($A^2$-block) [16]. We show overviews of each attention in Fig. 3.

As shown in Fig. 3 (a), SE has a channel attention that enables the network to focus on important channels. The SE gathers spatial features of the input feature map by using a global averaging pooling, applies two fully connected layers, and employs the sigmoid function, thereby obtaining an attention mask with a spatial size of 1. The fully connected layers reduce the number of channel from $C$ to $C/r$, where $r$ is a reduction ratio. GE-$\theta^+$ applies a global depth-wise convolution instead of the global averaging pooling as shown in Fig. 3 (b).

BAM is a combination of channel and spatial attentions (see Fig. 3 (c)). The spatial attention mask has the same size as that of the input, while its number of channel is 1; the spatial attention enables a network to focus on important spatial positions. In particular, spatial attention is composed of four layers. The first point-wise convolution reduces the number of channel from $C$ to $C/r$, two dilated convolutions with a dilation value $d$ are applied, and the second one reduces it from $C/r$ to 1. The channel attention of BAM is the same as that of SE. The attention mask of BAM is the sum of the outputs of the spatial and channel attentions. BAM multiplies the input tensor using the attention mask element, and adds the resulting output to the input tensor.

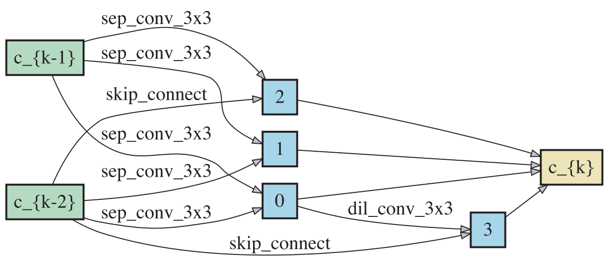
CBAM is also a combination of channel and spatial attentions as shown in Fig. 3 (d). CBAM gathers spatial features of the input feature map by max pooling in addition to global average pooling, applies a shared multi-layer perceptron (MLP), and finds the sum of the outputs. The shared MLP has the same size as the one used in the channel attention in SE. CBAM applies the sigmoid function to the output, and it obtains the channel attention mask. After channel attention, CBAM applies spatial attention, which applies the max and average poolings along with the channel axis, concatenates the outputs, applies a convolution and the sigmoid function, and obtains the spatial attention mask.

Figure 3 (e) shows a double-attention block ($A^2$-block). The double-attention block uses two groups of attentions: attention maps and attention vectors. Each attention is obtained by the softmax function following a pointwise convolution that reduces the number of channel from $C$ to $N$. The softmax function is applied over the spatial direction for the attention maps, and over the channel direction for the attention vectors. Furthermore, double-attention block applies a pointwise convolution to the input to reduce the number of channel from $C$ to $M$ before applying a bilinear pooling using the attention maps. Then, double-attention block applies the output matrix multiplication using the attention vectors. Finally, pointwise convolution restores the number of channel from $M$ to $C$. Note that the double-attention block adds the attention mask to the input, whereas the other attentions multiply the input by the attention mask. We use the same hyperparameter set for each attention as the one in the original papers. For example, the reduction ratio $r$ of SE was set to 16. Dilation value $d$ of a convolution in the spatial attention of BAM was set to 4, and the reduced channel numbers $M$ and $N$ in the double-attention block were set to $C/4$.

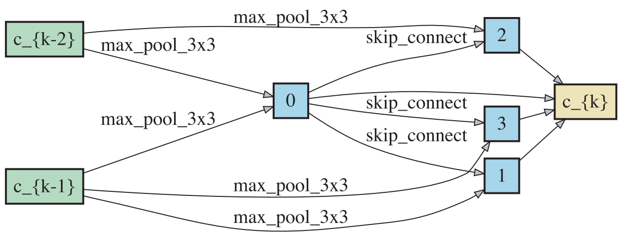
## 5. Results and Discussion

### 5.1 Chosen Architecture

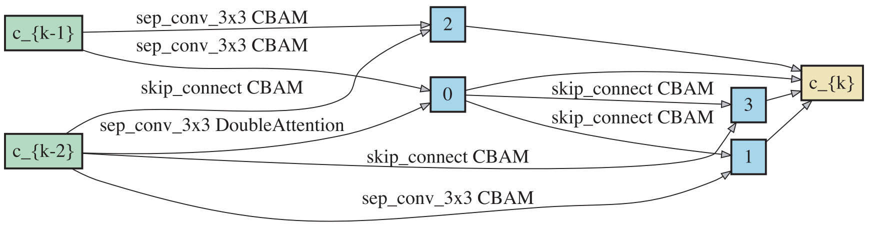
We illustrate the cells in Fig. 4. The skip connection is chosen as an operation more frequently by Att-DARTS than by



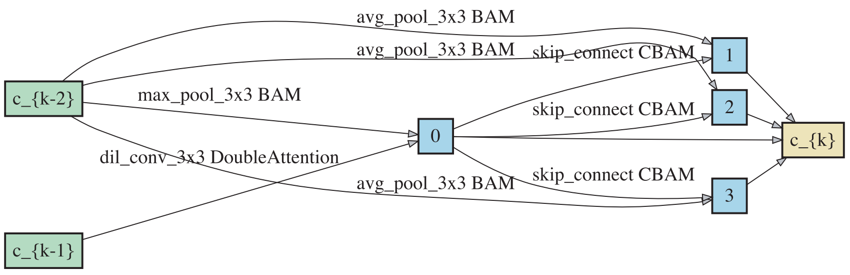
(a) Normal cell by DARTS.



(b) Reduction cell by DARTS.



(c) Normal cell by Att-DARTS.



(d) Reduction cell by Att-DARTS.

**Fig. 4** Illustrations of cells found by DARTS and Att-DARTS. Att-DARTS can find cells wherein the attention is inserted after each operation. sep_conv_3x3 and sep_conv_5x5 denote 3×3 and 5×5 separable convolutions respectively. dil_conv_3x3 and dil_conv_5x5 denote 3×3 and 5×5 dilated separable convolutions respectively. max_pool_3x3 denotes 3×3 max pooling. avg_pool_3x3 denotes 3×3 average pooling.

**Table 1** Comparison with DARTS on CIFAR-10 and CIFAR-100

| Architecture | Test Err. on CIFAR-10 (%) | Test Err. on CIFAR-100 (%) | Params (M) |
|---|---|---|---|
| DARTS [10] | 2.76 ± 0.09 | 16.69 ± 0.28[†] | 3.3 |
| Att-DARTS | **2.54** ± 0.10 | **16.54** ± 0.40 | **3.2** |

[†]Results by the best cells for CIFAR-10 reported in the original paper [10] as is the case with our Att-DARTS. Every network was trained with cutout [31].

**Table 2** Comparison with DARTS on ImageNet

| Architecture | Test Err. (%) | | Params (M) |
|---|---|---|---|
| | top-1 | top-5 | |
| DARTS [10] | 26.7 | 8.7 | 4.7 |
| Att-DARTS | **26.0** | **8.5** | **4.6** |

DARTS, which means that attentions are repeatedly applied without convolution operations (e.g., from node $c_{k-1}$ to node 1 via node 0). These attentions have different parameters, and the network can focus on more specific features. In contrast, Identity is never chosen as an attention despite it corresponding to a skip connection as an operation. Hence, one can say that Att-DARTS focuses on the attentions.

CBAM and BAM are the most and second-most frequently selected attentions, while SE and GE-$\theta^+$ are never selected. SE and GE-$\theta^+$ have only a channel attention, and their contributions are limited compared to CBAM and BAM, which have both channel and spatial attentions. This result emphasizes the importance of spatial attention. Despite CBAM and BAM having many convolution layers, Att-DARTS reduces the number of weight parameters; the entire network requires 3.2M weight parameters with Att-DARTS and 3.3M with DARTS. This is because the number of channels and spatial resolution are reduced in the attentions.

DARTS and Att-DARTS chose separable convolutions as operations most frequently. A separable convolution can be considered as a convolution whose kernel is

**Table 3** Comparison with state-of-the-art architectures on CIFAR-10

| Architecture | Test Err. (%) | Params (M) | Search Method |
|---|---|---|---|
| DenseNet-BC [37] | 3.46 | 25.6 | manual |
| NASNet-A + cutout [6] | 2.65 | 3.3 | RL[†] |
| AmoebaNet-B + cutout [7] | 2.55 ± 0.05 | 2.8 | evolution |
| Hierarchical Evolution [8] | 3.75 ± 0.12 | 15.7 | evolution |
| PNAS [38] | 3.41 ± 0.09 | 3.2 | SMBO[†] |
| ENAS + cutout [9] | 2.89 | 4.6 | RL[†] |
| DARTS + cutout [10] | 2.76 ± 0.09 | 3.3 | gradient |
| SNAS (moderate) + cutout [39] | 2.85 ± 0.02 | 2.8 | gradient |
| BayesNAS + cutout [40] | 2.81 ± 0.04 | 3.4 | gradient |
| PC-DARTS + cutout [24] | 2.57 ± 0.07 | 3.6 | gradient |
| Att-DARTS + cutout | **2.54** ± 0.10 | 3.2 | gradient |

[†]RL and SMBO mean reinforcement learning and sequential model-based optimization, respectively.

factorized and requires less parameters. DARTS chose only max pooling among pooling operations in the reduction cells, whereas Att-DARTS chose average pooling. Max pooling selects the most significant pixel in its kernel, as it is similar to an attention. Att-DARTS learns to select features by the attentions and does not need max pooling. The same tendency was found in the three remaining examples; the skip-connection operation and separable convolutions were the most frequently chosen operations, the CBAM and BAM were the most frequently chosen attentions, and the identity, SE and GE were never chosen as attentions.

## 5.2 Architecture Evaluation

We summarize the evaluation results on CIFAR-10 in Table 1. We report the average and standard deviations of five independent runs for CIFAR-10. Att-DARTS achieved a 2.54% test error with 3.2M parameters; Att-DARTS improved the error rate by 0.22% and reduced the number of parameters by 0.1M from DARTS. We also evaluated the best cells on CIFAR-100 and ImageNet as summarized in Table 1 and Table 2, respectively. We report the average and standard deviations of five independent runs for CIFAR-100. Att-DARTS also improved the error rate by 0.15% for CIFAR-100, 0.2% for ImageNet top-5 and 0.7% for ImageNet top-1.

We summarize the comparison with state-of-the-art architectures on CIFAR-10 in Table 3. Att-DARTS achieved the state-of-the-art results among NAS methods. AmoebaNet-B and PC-DARTS achieved comparable results, while the former requires thousands of GPU days and the latter requires 13% more parameters. Moreover, Att-DARTS's approach to search attentions can be combined with any gradient-based NAS methods such as PC-DARTS and potentially improves their performances.

Note that the architecture search space $\mathcal{O} \times \mathcal{A}$ of Att-DARTS is the product of the operation space $\mathcal{O}$ and the attention space $\mathcal{A}$. Hence, architecture search based on reinforcement learning or evolutionary algorithm encounters difficulty in finding a good architecture compared with the case only in the operation space $\mathcal{O}$.

## 5.3 Comparison with Alternatives

In Sect. 4, we have inserted an attention just after each operation on each edge. This approach is common among many studies on attentions [12], [13], [15]. However, few other studies inserted an attention at other locations; for example, the original study on BAM put it at each bottleneck of a network (e.g., after each ResBlock in ResNet) [14]. When following this study, an attention is inserted at the end of each cell because a cell in an NAS architecture corresponds to a ResBlock. Other studies used an attention module in a parallel with an operation [17]. From this perspective, one connection in a cell is chosen from all attention modules and operations. To investigate the appropriate location where an attention is inserted, we have examined several other candidate positions.

**after every:** We put an attention after every operation (as proposed above); the search space is $\mathcal{O} \times \mathcal{A}$ per edge. This is the most typical approach.

**end:** We put an attention at the end of each cell. This approach is a similar to that of BAM [14].

**after every + end:** We put an attention after every operation and at the end of each cell. This is an combination of the two above approaches.

**mixed with operation:** We put an attention or an operation on each edge exclusively; there is only one search space $\mathcal{O} \cup \mathcal{A}$ per edge. This approach is inspired by [17].

**double mixed:** We put two of attention and operation candidates on each edge; the search space is $(\mathcal{O} \cup \mathcal{A})^2$ per edge. After search stage, we chose the edges that have the first operations (or attentions) with the strongest relative weights. This approach is an extension of the above.

We report the results in Table 4. One can see that Att-DARTS worked the best with the "after every" position. Att-DARTS (end) improved the error rate by 0.19% from DARTS, and achieved a comparable performance with Att-DARTS (after every), but require much more parameters. Att-DARTS (after every + end), Att-DARTS (mixed

**Table 4** Comparison of Attention Positions in Att-DARTS on CIFAR-10.

| **Architecture** | **Test Err. (%)** | **Params (M)** |
|---|---|---|
| DARTS [10] | 2.76 ± 0.09 | 3.3 |
| Att-DARTS (after every) | **2.54** ± 0.10 | 3.2 |
| Att-DARTS (end) | 2.57 ± 0.05 | 5.9 |
| Att-DARTS (after every+ end) | 2.86 ± 0.13 | 5.5 |
| Att-DARTS (mixed with operation) | 3.23 ± 0.06 | 3.5 |
| Att-DARTS (double mixed) | 2.81 ± 0.13 | **2.8** |

Each network was trained with cutout [31].

with operation), and Att-DARTS (double mixed) performed poorly. The search space for Att-DARTS (after every + end) is $\mathcal{A}$ for each cell in addition to $\mathcal{O} \cup \mathcal{A}$ for each edge, and its size is larger than the size of search space $\mathcal{O} \times \mathcal{A}$ for Att-DARTS (after every). The search space for Att-DARTS (double mixed) is $(\mathcal{O} \cup \mathcal{A})^2$, which is a superset of the search space $\mathcal{O} \times \mathcal{A}$ for Att-DARTS (after every). Therefore, none of the additional methods could mitigate the increase in the search space. As described in Sect. 2, DARTS tends to choose many skip connections, and this is also the case with Att-DARTS. This is because an operation with more parameters needs a longer training, and DARTS underestimates operations with many parameters even if they have the potential for further improvements. Chu et al. [26] have pointed out a similar problem, and combination with their proposal is included in future works.

### 5.4 Other Possible Alternatives

The required attention might depend on the position in layers. Indeed, a previous study demonstrated that attention modules work more effectively at the latter part of a CNN than at the former part [17]. This is unsurprising because operations should extract features before attention discard useless features. A limited number of NAS studies considered "macro-search" strategy, where each cell has its own structure, and the found CNN structure can be used directly. With the macro-search strategy, the employed attentions are changeable depending on the position in a CNN. However, the gradient-based NAS methods train all candidate operations at the search stage, and hence they require large memory cost; the macro-search strategy is hardly applicable. Most NAS studies considered a CNN as a chain-like structure that repeats the same cells; this strategy is called "micro-search" [2], [9]. With the micro-search strategy, the same attentions are used at every cell of a deep CNN, but the number of the cells is flexible. The number of cells is reduced to surpass the memory cost at the search stage, and then it is increased to pursue a better performance. This is why we employed the micro-search strategy and used the same attentions independently from the position in a CNN. The macro-search strategy for attentions will be a future work.

The search space of Att-DARTS is transferable to other gradient-based NAS methods such as BayesNAS [40] and PC-DARTS [24], and possibly improves their performance. Since the gradient-based NAS methods evaluate the sum of all candidate operations and attentions, the computational cost is simply increased from $O(|\mathcal{O}|)$ to $O(|\mathcal{O}| + |\mathcal{A}|)$. The NAS methods based on reinforcement learning and evolutionary algorithms evaluate a cell with one of all the possible combinations of operations and attentions at once, and the number of the possible combinations per edge is increased from $|\mathcal{O}|$ to $|\mathcal{O}| \times |\mathcal{A}|$. Hence, the methods encounter a difficulty in searching the best combination. In this study, Att-DARTS chose operations and attentions from limited subsets. We recommend the methods to search a cell from the subsets to reduce the search cost.

## 6. Conclusion

In this work, we proposed Att-DARTS, a differentiable architecture search that finds cells with attentions. Att-DARTS assumes a CNN composed of repeatedly stacked cells similar to most existing NAS works; however, it inserts an attention after each operation, unlike the previous works. The experimental results on CIFAR-10, CIFAR-100 and ImageNet demonstrated that Att-DARTS found an architecture that achieves better classification error rates and requires less parameters than that found by its non-attentive counterpart, DARTS. Future works will include evaluations on a larger CNN and a more flexible cell topology, as well as developing a more appropriate search space for CNN.

## Acknowledgments

This study was partially supported by the MIC/SCOPE #172107101 and JSPS KAKENHI (19H04172, 19K20344).

### References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.770–778, 2016.

[2] T. Elsken, J.H. Metzen, and F. Hutter, "Neural architecture search: A survey," Journal of Machine Learning Research, vol.20, pp.1–21, 2019.

[3] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. Yuille, and L. Fei-Fei, "Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.82–92, 2019.

[4] R. Quan, X. Dong, Y. Wu, L. Zhu, and Y. Yang, "Auto-ReID: Searching for a Part-aware ConvNet for Person Re-Identification," The IEEE International Conference on Computer Vision (ICCV), pp.3750–3759, 2019.

[5] B. Zoph and Q.V. Le, "Neural architecture search with reinforcement learning," International Conference on Learning Representations (ICLR), 2017.

[6] B. Zoph, V. Vasudevan, J. Shlens, and Q.V. Le, "Learning Transferable Architectures for Scalable Image Recognition," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.8697–8710, 2018.

[7] E. Real, A. Aggarwal, Y. Huang, and Q.V. Le, "Regularized Evolution for Image Classifier Architecture Search," The AAAI Conference on Artificial Intelligence (AAAI), vol.33, pp.4780–4789, 2019.

[8] H. Liu, et al., "Hierarchical representations for efficient architecture search," International Conference on Learning Representations

(ICLR), 2018.
[9] H. Pham, et al., "Efficient neural architecture search via parameter sharing," International Conference on Machine Learning (ICML), vol.9, pp.6522–6531, 2018.
[10] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable Architecture Search," International Conference on Learning Representations (ICLR), 2019.
[11] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.6450–6458, 2017.
[12] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation Networks," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.7132–7141, 2018.
[13] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," Neural Information Processing Systems (NeurIPS), pp.9401–9411, 2018.
[14] J. Park, S. Woo, J.Y. Lee, and I.S. Kweon, "BAM: Bottleneck Attention Module," British Machine Vision Conference (BMVC), 2018.
[15] S. Woo, J. Park, J.-Y. Lee, and I.S. Kweon, "CBAM: Convolutional block attention module," The European Conference on Computer Vision (ECCV), 2018.
[16] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "$A^2$-Nets: Double Attention Networks," Neural Information Processing Systems (NeurIPS), pp.352–361, 2018.
[17] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q.V. Le, "Attention Augmented Convolutional Networks," The IEEE International Conference on Computer Vision (ICCV), pp.3286–3295, 2019.
[18] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," Technical report, University of Toronto, 2012.
[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: a Large-Scale Hierarchical Image Database," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.248–255, 2009.
[20] K. Nakai, T. Matsubara, and K. Uehara, "Att-DARTS: Differentiable Neural Architecture Search for Attention," The International Joint Conference on Neural Networks (IJCNN), 2020.
[21] A. Brock, T. Lim, J.M. Ritchie, and N. Weston, "SmaSH: One-shot model architecture search through hypernetworks," International Conference on Learning Representations (ICLR), 2018.
[22] G. Bender, P.J. Kindermans, B. Zoph, V. Vasudevan, and Q. Le, "Understanding and simplifying one-shot architecture search," International Conference on Machine Learning (ICML), vol.2, pp.883–893, 2018.
[23] X. Chen, L. Xie, J. Wu, and Q. Tian, "Progressive Differentiable Architecture Search: Bridging the Depth Gap between Search and Evaluation," The IEEE International Conference on Computer Vision (ICCV), pp.1294–1303, 2019.
[24] Y. Xu, L. Xie, X. Zhang, X. Chen, G.-J. Qi, Q. Tian, and H. Xiong, "PC-DARTS: Partial Channel Connections for Memory-Efficient Differentiable Architecture Search," International Conference on Learning Representations (ICLR), 2020.
[25] H. Liang, S. Zhang, J. Sun, X. He, W. Huang, K. Zhuang, and Z. Li, "DARTS+: Improved Differentiable Architecture Search with Early Stopping," 2019.
[26] X. Chu, T. Zhou, B. Zhang, and J. Li, "Fair DARTS: Eliminating Unfair Advantages in Differentiable Architecture Search," 2019.
[27] S. Pereira, A. Pinto, J. Amorim, A. Ribeiro, V. Alves, and C.A. Silva, "Adaptive Feature Recombination and Recalibration for Semantic Segmentation with Fully Convolutional Networks," IEEE Trans. Med. Imag., vol.38, no.12, pp.2914–2925, 2019.
[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision, vol.115, no.3, pp.211–252, 2015.
[29] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," International Conference on Learning Representations (ICLR), 2017.
[30] D.P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," International Conference on Learning Representations (ICLR), 2015.
[31] T. DeVries and G.W. Taylor, "Improved Regularization of Convolutional Neural Networks with Cutout," 2017.
[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Journal of Machine Learning Research, vol.15, pp.1929–1958, 2014.
[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1–9, 2015.
[34] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017.
[35] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1800–1807, 2017.
[36] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," International Conference on Learning Representations (ICLR), 2016.
[37] G. Huang, Z. Liu, L. Van Der Maaten, and K.Q. Weinberger, "Densely connected convolutional networks," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.2261–2269, 2017.
[38] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive Neural Architecture Search," The European Conference on Computer Vision (ECCV), 2018.
[39] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: Stochastic neural architecture search," International Conference on Learning Representations (ICLR), 2019.
[40] H. Zhou, M. Yang, J. Wang, and W. Pan, "BayesNAS: A Bayesian Approach for Neural Architecture Search," International Conference on Machine Learning (ICML), vol.97, pp.7603–7613, 2019.
[41] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, "FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.10726–10734, 2019.

## Appendix A: Computational Cost

### A.1 Search Stage

Our method expands NAS search space from $\mathcal{O}$ to $\mathcal{O} \times \mathcal{A}$. For gradient-based methods such as DARTS, the computational cost is increased from $O(|\mathcal{O}|)$ to $O(|\mathcal{O}| + |\mathcal{A}|)$ because gradient-based NAS methods compute all candidate operations and attentions concurrently. We summarized the computational cost in Appendix A.2 for reference. An attention typically requires a smaller computational cost than an operation; the computational cost of gradient-based NAS $O(|\mathcal{O}| + |\mathcal{A}|)$ is smaller than $O(2|\mathcal{O}|)$. However, an attention module is composed of several substeps and it was inserted after an operation. This fact implies that the cell found by Att-DARTS is not well parallelized on graphics processing units (GPUs) and may require a longer real-time computation. Using 2 NVIDIA GeForce GTX 1080Ti, Att-DARTS requires 3 days to search a cell, whereas DARTS requires 1

**Table A· 1** Computational costs of each operation and attention.

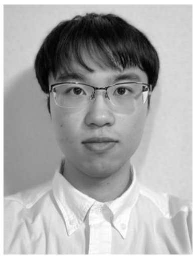
| Operation | Computational cost |
|---|---|
| Convolution | $O(KSC^2)$ |
| Separable convolution | $O(KSC + SC^2)$ |
| Dilated convolution | $O(d^{-2}KSC^2)$ |
| Max/Average pooling | $O(KSC)$ |
| BAM | $O(KSC^2)$ |
| CBAM | $O(SC + C^2 + KS)$ |
| Double Attention | $O(SC^2)$ |

$K$ denotes the square of kernel size. $S$ denotes the spatial size $WH$. $d$ denotes the dilation number. Note that the output is assumed to have the same channel size $C$ as the input.

day.

## A.2 Operation, Attention, and Cell

After the search stage, only a limited subset of attentions and operations were retained as shown in Fig. 4. The Att-DARTS mainly employed the CBAM as attention modules and the skip connection as operations, while the DARTS employed the separable convolution. The CBAM requires the computational cost, which is of lower-order than that of separable convolution: the former is $O(SC + C^2 + KS)$ and the latter is $O(KSC + SC^2)$. Hence, the cell found by Att-DARTS requires a lower theoretical computational cost and, as mentioned above, a longer real-time computation than DARTS. Searching a cell with a lower latency can be a future work [41].



**Kohei Nakai** is a graduate student at the Graduate School of System Informatics, Kobe University, Hyogo, Japan. He received his B.E. degree from Kobe University in 2020. He is interested in neural architecture search.

**Takashi Matsubara** received his B.E., M.E., and Ph.D. in engineering from Osaka University, Osaka, Japan, in 2011, 2013, and 2015, respectively. From 2015 to 2020, he was an Assistant Professor at the Graduate School of System Informatics, Kobe University, Hyogo, Japan. He is currently an Associate Professor at the Graduate School of Engineering Science, Osaka University, Osaka, Japan. His research interests are in computational intelligence and computational neuroscience.

**Kuniaki Uehara** received his B.E., M.E., and D.E. degrees in information and computer sciences from Osaka University, Osaka, Japan, in 1978, 1980 and 1984, respectively. From 1984 to 1990, he was with the Institute of Scientific and Industrial Research, Osaka University as an Assistant Professor. From 1990 to 1997, he was an Associate Professor with Department of Computer and Systems Engineering of Kobe University. From 1997 to 2002, he was a Professor with the Research Center for Urban Safety and Security of Kobe University. From 2002 to 2020, he was a Professor with Graduate School of System Informatics of Kobe University. Currently he is a Professor with Faculty of Business Administration of Osaka Gakuin University.