Sun *et al. Journal of Cloud Computing: Advances, Systems and Applications* (2020) 9:55
https://doi.org/10.1186/s13677-020-00203-9

Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH Open Access

Lightweight image classifier using dilated and depthwise separable convolutions

Wei Sun[1,2*], Xiaorui Zhang[2,3] and Xiaozheng He[4]

Abstract

The image classification based on cloud computing suffers from difficult deployment as the network depth and data volume increase. Due to the depth of the model and the convolution process of each layer will produce a great amount of calculation, the GPU and storage performance of the device are extremely demanding, and the GPU and storage devices equipped on the embedded and mobile terminals cannot support large models. So it is necessary to compress the model so that the model can be deployed on these devices. Meanwhile, traditional compression based methods often miss many global features during the compression process, resulting in low classification accuracy. To solve the problem, this paper proposes a lightweight neural network model based on dilated convolution and depthwise separable convolution with twenty-nine layers for image classification. The proposed model employs the dilated convolution to expand the receptive field during the convolution process while maintaining the number of convolution parameters, which can extract more high-level global semantic features to improve the classification accuracy. Also, the depthwise separable convolution is applied to reduce the network parameters and computational complexity in convolution operations, which reduces the size of the network. The proposed model introduces three hyperparameters: width multiplier, image resolution, and dilated rate, to compress the network on the premise of ensuring accuracy. The experimental results show that compared with GoogleNet, the network proposed in this paper improves the classification accuracy by nearly 1%, and the number of parameters is reduced by 3.7 million.

Keywords: Classification accuracy, Cloud computing, Depthwise separable convolution, Dilated convolution, Lightweight neural network

Introduction

In recent years, deep networks have made significant progress in many fields, such as image processing, object detection, and semantic segmentation. Krizhevsky, et al. [1] first adopted deep learning algorithm and the AlexNet and won the champion of ImageNet Large Scale Visual Recognition Challenge in 2012, which improved the recognition accuracy by 10% compared to the traditional machine learning algorithm. Since then, various convolutional neural network models have been proposed in the computer vision community, including the VGGNet proposed by the Visual Geometry Group at the University of Oxford [2] in 2014, the GoogLeNet [3, 4] by Google researchers, and the ResNet by He et al. [5, 6] in 2015. These networks are superior to AlexNet [7, 8]. The trend of improvement is using deeper and more complex networks for higher accuracy. With a higher precision for computer vision tasks, the model depth and parameters are also increasing exponentially, making these models dependent more on computationally-powerful GPUs [4, 9]. As a consequence, existing deep neural network models cannot be deployed on resource-constrained devices [10, 11], such as smart-phones and in-vehicle

*Correspondence: sunw0125@163.com
[1]School of Automation, Nanjing University of Information Science and Technology, 210044 Nanjing, China
[2]Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology, 210044 Nanjing, China
Full list of author information is available at the end of the article

devices, due to their limited computing power. The emerging cloud computing has the potential to solve this challenge [12].

Cloud computing technology, which combines the characteristics of distributed computing, parallel computing and grid computing, provides users with scalable computing resources and storage space by using massive computing clusters built by ordinary servers and storage clusters built by a large number of low-cost devices. At present, a large number of enterprises have enterprise-level cloud computing platforms: amazon cloud computing, alibaba cloud computing, baidu cloud computing, and so on. Compared with the traditional application platform, cloud computing platform has many fine characteristics, such as strong computing capacity, infinite storage capacity, convenient and fast virtual service and so on. However, renting the cloud computing servers need extra cost for individuals and small companies. For example,The model training in this article can be run on an NVIDIA P4 cloud server with 8g memory. This server is the most basic server and costs $335 per month. Although the cost is not too expensive for a company, it is a huge expenditure for students without salary. Therefore, there is the need to design a lightweight network to reduce the model's dependence on high-performance devices [13, 14].

To reduce the network's dependence on high performance servers and reduce the cost of cloud computing. various new lightweight networks are proposed for object detection. By compressing the model, the size of neural network is reduced [15, 16]. Typical strategies involve avoiding full connection in the network, reducing the number of channels and the size of convolution kernel, as well as optimizing down-sampled, weight pruning, weight discretization, model representation and coding [17, 18]. For example, GoogleNet [3, 19] increased the width of the network to reduce the network complexity; the subsequent Xception network extended the depthwise separable filter to overcome the shortcomings in the InceptionV3 network [5, 20]. The article MobileNet [21] proposes a deep separable convolution, which shows great potential for decomposing networks. However, the classification accuracy of these models cannot be guaranteed during the compression due to omitting excessive image features for simplified convolution operation [22, 23].

Aimed to address the above issues, this paper proposes a lightweight neural network combining dilated convolution and depthwise separable convolution. Inspired by the MobileNet, this paper adopts a depthwise separable convolution architecture and hyperparameters, width multiplier, and resolution multiplier to obtain a small network model that can be applied to resource-constrained devices such as smartphones [24, 25]. The convolution process is divided into two processes by depthwise separable convolution to reduce network computation. Because the depthwise separation convolution cannot guarantee the classification accuracy of the model [26], the proposed model integrates the dilated convolution into the depthwise separable convolution architecture. The dilated convolution can increase the receptive field of the network in the convolution process without increasing convolution parameters, which can extract more global features and higher-level semantic features, thus improving the classification accuracy of the network [27, 28]. Finally, the proposed model is further compressed by reducing the number of input channels and the resolution of input image using hyperparameter strategy. Compared with other networks, the network proposed in this paper can ensure higher classification accuracy while using fewer resources. In addition, the joint dilated convolution and depthwise separable convolution method proposed in this paper effectively solves the problem that model size and classification accuracy cannot coexist.

Related work

In the current state-of-the-art, deep neural network compression can be conducted in two approaches: i) compressing the trained models by optimizing the network parameters and ii) designing and training small network models directly [29].

For the first approach, Han introduced compression methods such as cropping, weight sharing, quantization, and coding to deep network model in 2015. In general, a complex network has good performance, but its parameters may also be redundant [20]. Therefore, for a network that has already been trained, unimportant hierarchical connections or filters can be tailored to reduce model parameters and redundancy. In the training process, a weight update strategy is introduced to make it sparser, but the commonly used sparse matrix operation is not efficient on the hard-ware platform and is susceptible to hardware devices [30].

The second approach has become popular with the introduction of lightweight models such as SqueezeNet [31], ShuffleNet [32], and MobileNet [21]. The SqueezeNet proposed by Landola et al. applies a convolution kernel to convolve and dimension the upper features and a feature convolution to perform feature stacking, which greatly reduces the number of parameters of convolution layers. SqueezeNet uses a bottleneck method to design a small network that greatly reduces the parameters and computational complexity while maintaining accuracy [19]. Zhang et al. [32] proposed the ShuffleNet, which groups multi-channel feature lines and then performs convolution to avoid unsmooth information flow. ShuffleNet [32] network reduces the amount of network computation through channel shuffling and point-group convolution. Howard et al. [21] proposed the depthwise separable convolution model, named MobileNet, where

the features of each channel convolved separately and then uses 1×1 convolution to splice all features of different channels. These lightweight models reduce the number of network parameters and computational cost. However, during the compression process, the classification accuracy of the model cannot be guaranteed because only local information of the image is utilized [33–35].

Aimed to achieve a lightweight model while ensuring the classification accuracy, this paper combines the above two methods. Firstly, directly design and train a small network model by combining depthwise separable convolution and dilated convolution. The depthwise separable convolution is used to reduce the parameter number and computation burden, and the dilated convolution improves the accuracy of the model. Secondly, inspired by the MobileNet, the proposed model applies the hyperparameters to further compress the trained model, thereby making the model to adapt to source-constrained devices.

Approach

This paper uses dilated convolution as a filter to extract image features. Compared to the traditional filters, the dilated convolution yields more full-image information without increasing the number of network parameters, where the dilated rate δ controls the size of each convolution dilation. Then, we apply depthwise separable convolution instead of traditional convolution to reduce the computational complexity. To compress the model further, we introduce two hyperparameters proposed in MobileNet: width multiplier α and resolution multiplier ρ, to evenly reduce the computational burden of each layer of the network [30, 36]. This paper combines the dilated convolution and the depthwise separable convolution to ensure the classification accuracy while maintaining the model to be lightweight by adjusting hyperparameters. This section first presents the idea of building a joint module of dilated convolution and depthwise separable convolution, which is then used to build the deep convolution network.

Joint module

As shown in Fig. 1, the proposed model dilates each filter to obtain more image information without increasing the computation burden and the number of channels. The dilated filter is then used to convolve each input channel, and the final filter is used filter to combine the output of different convolution channels.

Figure 2 illustrates the dilation process of the 3×3 filter for the dilated convolution process in Fig. 1. The position of the node without the dot mark in Fig. 2 indicates that there is a zero weight, and the node with the dot mark represents non-zero weight to that position. It represents filters having different dilated rates, respectively, in Fig. 2a, b, and c. The parameters of the convolution layer remain

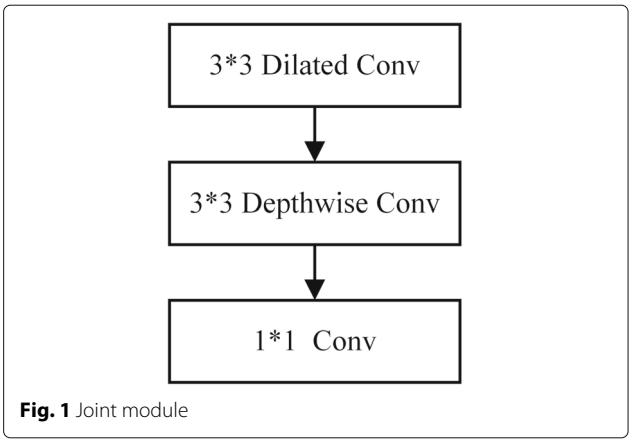


Fig. 1 Joint module

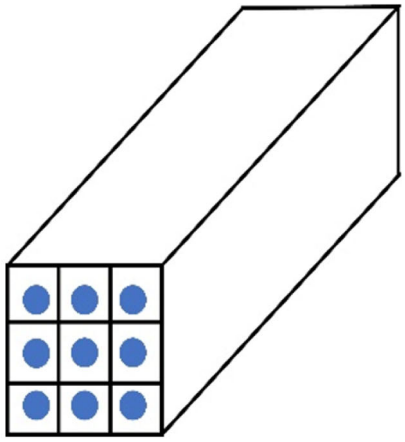
the same, so the amount of convolution process remains the same too. The fields of the filters (a), (b), and (c) are defined as $3\times3 = 9, 7\times7 = 49$, and $11\times11 = 121$, respectively. Filter (c) has the largest receptive field, indicating that each node on the feature map corresponds to more feature information. With the increase of the receptive field, it means that each node contains higher semantic features, which can improve the classification accuracy of the network. To factor the influence of different dilated convolution on model accuracy, we apply hyperparameter δ to control the size of each dilated convolution. As illustrated by Fig. 2, the relationship between the receptive field and the original filter size can be represented as:

$$C = (S \times \delta + (\delta - 1))^2. \tag{1}$$

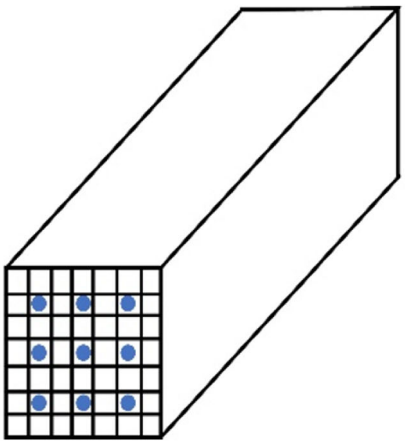
where C denotes the size of the receptive field, S represents the size of the initial filter, and δ represents the dilated rate.

The separable convolution operation is carried out on the obtained dilated convolution filter. The size of the dilation filter is $L_k \times L_k$ with $L_k = \sqrt{C}$. Figure 3 shows the process of constructing a $L_i \times L_i \times H$ feature map and a $L_i \times L_i \times N$ feature map. This process shows how to reduce the number of parameters in the model.

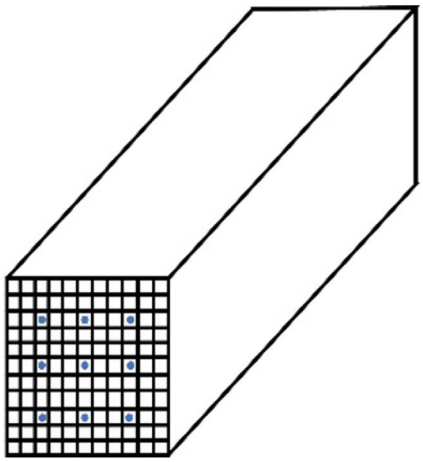
Figure 3a, b, and c represent the traditional convolution filter, depthwise convolution filter, and pointwise convolution filter, respectively. Figure 3b and c together represent a separable convolution process, where $L_i \times L_i$ is the width and height of the input feature map, N is the number of filters, $L_k \times L_k$ is the width and height of the dilated filter, and H is the number of channels. For example, a single dilated filter of $L_k \times L_k$ is firstly used to carry out the convolution operation on each channel. If the number of the feature map channels is H here are H filters with the same size to participate in the convolution operation, and the number of channels of each filter is 1. The image is then convolved by N filters with 1×1 size and convolution channels. Figure 3 shows that the traditional convolution



(a) 3×3 Standard filter $\delta = 1$

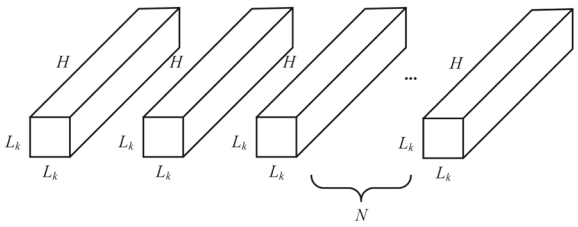


(b) Filter with $\delta = 2$

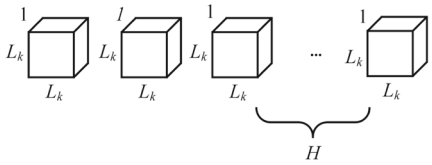


(c) Filter with $\delta = 3$

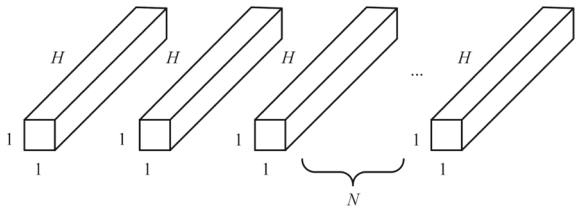
Fig. 2 Dilated convolution process



(a) Traditional convolution filters



(b) Depthwise convolution filters



(c) Pointwise convolution filters

Fig. 3 Dilated convolution process

layer takes a $L_i \times L_i \times H$ feature map as the input and produces a $L_i \times L_i \times N$ feature map, in which $L_i \times L_i$ is the width and height of input feature map, H is the number of input channels, N is the number of output channel, $L_k \times L_k$ is the width and height of the dilated filter. G_t represents the amount of parameters in the traditional convolution process.

$$G_t = L_k \times L_k \times H \times N \times L_i \times L_i. \tag{2}$$

G_d is the number of parameters of the depthwise separable convolution process.

$$G_d = L_k \times L_k \times H \times L_i \times L_i + H \times N \times L_i \times L_i. \tag{3}$$

Therefore, the ratio of separable convolution to the traditional convolution can be represented by:

$$\frac{G_d}{G_t} = \frac{1}{N} + \frac{1}{L_k^2}. \tag{4}$$

Equation (4) shows that the calculation can be reduced to $\frac{1}{N} + \frac{1}{L_k^2}$ compared to the conventional convolution process, which lowers computational complexity.

Network architecture

To avoid the gradient disappearance problem and speed up the network training, we apply the BN layer (Batch Normalization) and the ReLU layer to make the gradient larger [37, 38] after introducing the joint module above. We call the process presented as a basic network structure, shown in Fig. 4.

Using only one basic structure is not enough to form a usable neural network, because we cannot receive deep information about the image if the network is too shallow. Therefore, there is the need to construct a lightweight neural network based on Fig. 4. As shown in Fig. 5, several basic network structures are combined with the average pooling layer, the full connection layer, and the Softmax layer to form the overall network structure. Table 1 shows the entire composition of this lightweight neural network in detail. Class represents the category of the dataset in the table. In total, the model includes one average pooling layer and one fully connected layer, nine dilated convolution layers, nine depthwise separable convolution layers, and nine BN layers.

The model dilates the 3×3 convolution kernel before implementing each depthwise separable convolution. Through the dilated rate to obtain a convolution kernel with a larger receptive field. The obtained 3×3 dilated convolution is applied to each channel of the feature map, and then 1×1 convolution is used to combine the output of the channel convolution. Adding a BN layer and a Relu linear activation function after each 1×1 convolution operation can accelerate training speed and improve the generalization capability of the network [39, 40].

Hyperparameters

This study adjusts the dilated rate δ to change the size of the dilated convolution. The specific experimental results will be introduced in the next section. Different devices require smaller and faster models. Therefore, this paper refers to two hyperparameters, the width multiplier α and resolution multiplier ρ , to obtain a smaller model. The two hyperparameters reduce the computational complexity of the entire network by reducing the computational complexity of the depthwise separable convolution process. The role of the width multiplier is to thin a network uniformly at each layer. The number of input channels changes from H to αH , and the number of output channels becomes αN from N. As a result, the complexity of the depthwise separable convolution is:

$$G_\alpha = L_k \times L_k \times \alpha H \times L_i \times L_i + \alpha H \times \alpha N \times L_i \times L_i. \quad (5)$$

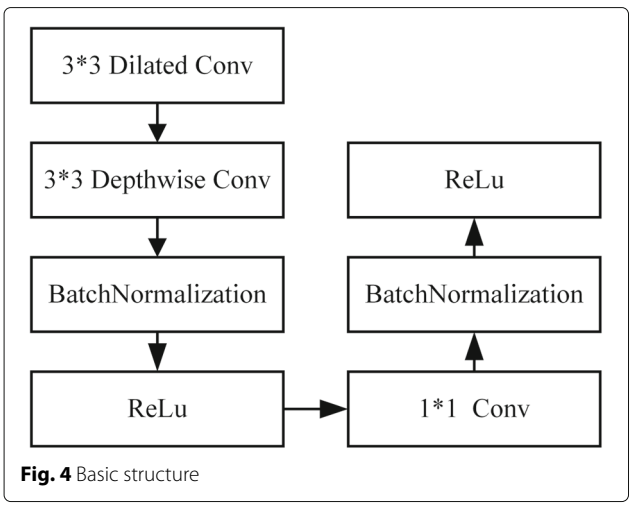


Fig. 4 Basic structure

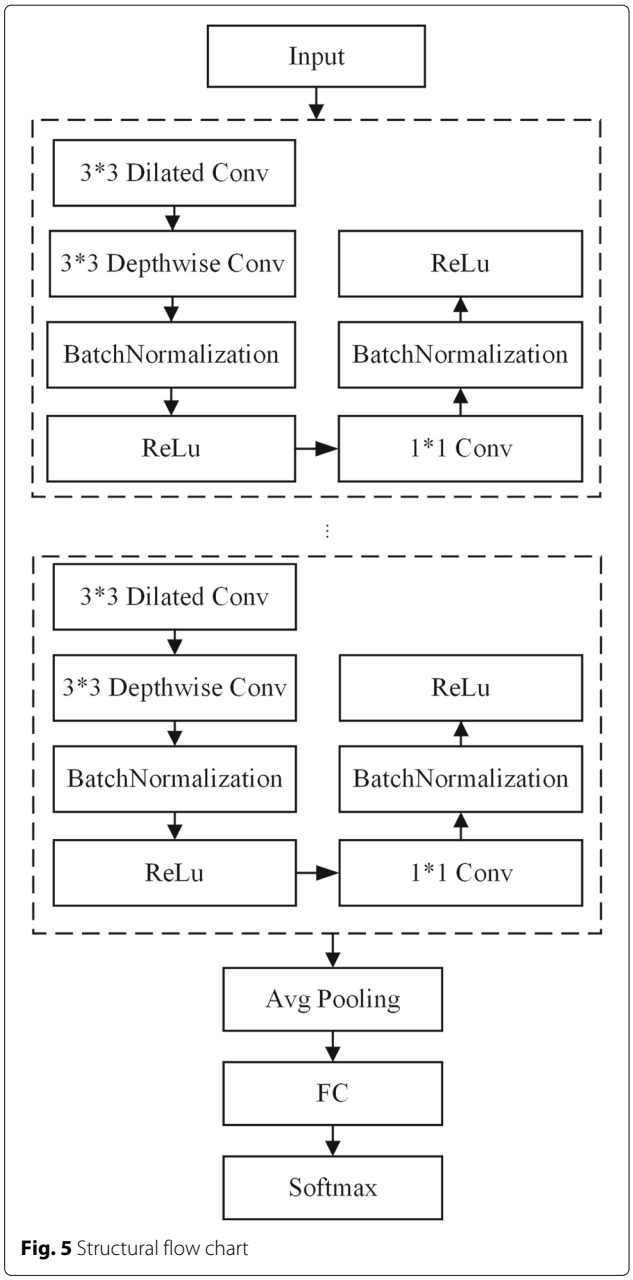


Fig. 5 Structural flow chart

Table 1 Overall architecture

Type	Filter shape	Stride	Input size
Dilated Conv	3 × 3 × 32	1	224 × 224 × 3
Depthwise	3 × 3 × 32	2	224 × 224 × 32
Separable Conv	1 × 1 × 64	1	112 × 112 × 32
Dilated Conv	3 × 3 × 64	1	112 × 112 × 64
Depthwise	3 × 3 × 64	2	56 × 56 × 64
Separable Conv	1 × 1 × 128	1	56 × 56 × 64
Dilated Conv	3 × 3 × 128	1	1 × 1 × 128
Depthwise	3 × 3 × 128	1	56 × 56 × 128
Separable Conv	1 × 1 × 128	1	56 × 56 × 128
Dilated Conv	3 × 3 × 128	1	56 × 56 × 128
Depthwise	3 × 3 × 128	2	28 × 28 × 128
Separable Conv	1 × 1 × 256	1	28 × 28 × 128
Dilated Conv	3 × 3 × 256	1	28 × 28 × 256
Depthwise	3 × 3 × 256	1	28 × 28 × 256
Separable Conv	1 × 1 × 256	1	28 × 28 × 256
Dilated Conv	3 × 3 × 256	1	28 × 28 × 256
Depthwise	3 × 3 × 256	2	14 × 14 × 256
Separable Conv	1 × 1 × 512	1	14 × 14 × 256
Dilated Conv	3 × 3 × 512	1	14 × 14 × 512
Depthwise	3 × 3 × 512	1	14 × 14 × 512
Separable Conv	1 × 1 × 512	1	14 × 14 × 512
Dilated Conv	3 × 3 × 512	1	14 × 14 × 512
Depthwise	3 × 3 × 512	2	14 × 14 × 512
Separable Conv	1 × 1 × 1024	1	7 × 7 × 512
Dilated Conv	3 × 3 × 1024	1	7 × 7 × 1024
Depthwise	3 × 3 × 1024	1	7 × 7 × 1024
Separable Conv	1 × 1 × 1024	1	7 × 7 × 1024
Avg Pool	7 × 7	1	7 × 7 × 1024
FC	1024×class	1	1 × 1 × 1024
softmax	Classifier	1	1 × 1×class

where G_α indicates the amount of calculation, where $\alpha \in (0, 1]$ with a typical value of 1, 0.75, or 0.5 [23]. It represents compression factor. Note that $\alpha < 1$ represents a narrow network. The second hyperparameter ρ is a resolution multiplier. By applying this strategy to the input image, the internal representation of every layer is subsequently reduced. For example, the size of the feature map of each layer of the convolution becomes ρ^2 compared to the original input image. The computational complexity of the depthwise separable convolution is:

$$G_\rho = L_k \times L_k \times H \times \rho L_i \times \rho L_i + H \times N \times \rho L_i \times \rho L_i. \quad (6)$$

where $\rho \in (0, 1]$ which is set implicitly so that the input resolution of the network is 224, 192, or 160 [23]. It represents the size of input images. When $\rho < 1$ it is named the reduced computation network. We use ρ to further compress the trained model. Accordingly, the computational complexity of two hyperparameters is shown as follows:

$$G_{\alpha\rho} = L_k \times L_k \times \alpha H \times \rho L_i \times \rho L_i + \alpha H \times \alpha N \times \rho L_i \times \rho L_i. \quad (7)$$

The computational complexity of the model is reduced by adopting these two hyperparameters, which can be applied to various source-constrained devices. Meanwhile, to ensure the classification accuracy, we need to compromise the hyperparameters α, ρ, δ to get the best model in sections experiments.

Loss function and optimization

We adopt cross-entropy as the loss function of neural network, using Adam as the network optimizer [41]. The formula for cross-entropy is as follows:

$$W(p, q) = \sum_i p(i) * \log\left(\frac{1}{q(i)}\right) \quad (8)$$

where $W(p, q)$ represents cross-entropy, p represents the distribution of the true mark, q is the predicted mark distribution of the trained model, and cross-entropy loss function can measure the similarity between p and q.

Adam is considered to be robust in selecting hyperparameters [11]. Therefore, this paper adopts an adaptive Adam learning rate to optimize the proposed model. In Adam, momentum is incorporated directly as an estimate of the first-order moment (with exponential weighting) of the gradient. Meanwhile, Adam includes bias corrections to the estimates of both the first-order moments (the momentum term) and the (uncentered) second-order moments to account for their initialization at the origin [41]. The optimization steps are presented in Table 2.

Experiments

To verify the effectiveness of the proposed method, we constructed an experimental platform and selected a typical dataset. The proposed network model was compared with other models to verify its effectiveness. Furthermore, we investigated the influence of the dilated convolution size on the classification accuracy of the model and verified the classification accuracy. We also verified the compression effect and accuracy of the proposed model through hyperparameters. All experiments were carried out on a computer with Intel Core i7-7700k CPU, 4.20Ghz×8 frequency, and GTX 1080Ti graphics card. CUDA version 9.0 and cuDNN version 7.3.1 were installed. The proposed model and algorithm were compiled and operated on TensorFlow 1.12.2.

There are many datasets available on the Internet. We select the CIFAR-10 dataset to verify the proposed model,

Table 2 Optimization algorithm

Algorithm: The optimization with the improved loss function

Input: Sample a minibatch of m examples from the training set $\{x^{(1)}, ..., x^{(m)}\}$ with corresponding targets $y^{(i)}$.

Initialization: Step size $\varepsilon = 0.001$, exponential decay rates for moment estimates $\rho_1 = 0.9$, $\rho_2 = 0.999$ and small constant δ used for numerical stabilization $\delta = 10^{-8}$.

Output: Network parameters θ.

1. Initialize: Network parameters θ, 1st and 2nd moment variables $s = 0$, $r = 0$ and time step $t = 0$.

2. While stopping criterion not met do.

Compute gradient: $g \leftarrow \frac{1}{m}\nabla_\theta \sum_i L(f(x^{(i)};\theta), y^{(i)})$.

$t = t + 1$.

Update biased first moment estimate:

$s \leftarrow \rho_1 s + (1 - \rho_1)g$.

Update biased second moment estimate:

$r \leftarrow \rho_2 r + (1 - \rho_2)g \odot g$.

Correct bias in first moment: $\bar{s} \leftarrow \frac{s}{(1-\rho_1^t)}$.

Correct bias in second moment: $\bar{r} \leftarrow \frac{r}{(1-\rho_2^t)}$.

Compute update: $\Delta\theta = -\varepsilon \frac{\bar{s}}{\sqrt{\bar{r}}+\delta}$.

Apply update: $\theta = \theta + \Delta\theta$.

3. end while.

4. Return θ.

because it applies recognition to ubiquitous objects and applies to multiple classifications and the dataset size is also suitable for most classifier training. In addition, according to experimental requirements, the experiment requires different-resolution pictures. CIFAR-10 dataset contains 60,000 color images, all of which are 32 × 32 pixels. The dataset has been divided into 10 categories, each of which includes 6000 images. We selected 50000 images from the dataset as the training set. The training set constitutes five training batches, and each batch includes 10,000 images. Another 10,000 images are used for testing, forming a separate batch. In the test batch, 1000 images are randomly selected from each of the 10 categories, and the rest are randomly arranged to form the training batch again. The number of images with different categories in each training batch is not necessarily the same. Meanwhile, The Tiny ImageNet dataset is used to verify the generalization capability of the proposed net-work. The dataset spans 200 image classes with 500 training examples per class. The dataset also has 50 validation and 50 test examples per class. The images are down-sampled to 64 × 64 pixels.

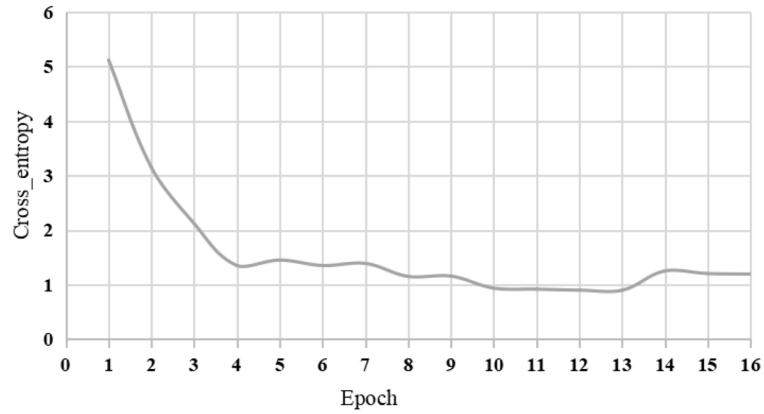
Training results and optimal selection

As shown above, the complete network structure has been set up and the dataset has been selected. Next, we need to train the built model. In the training of the network, the best training result is selected by observing the change of the loss function to test the classification accuracy of the model. The change of the loss function on different datasets is shown in Fig. 6.

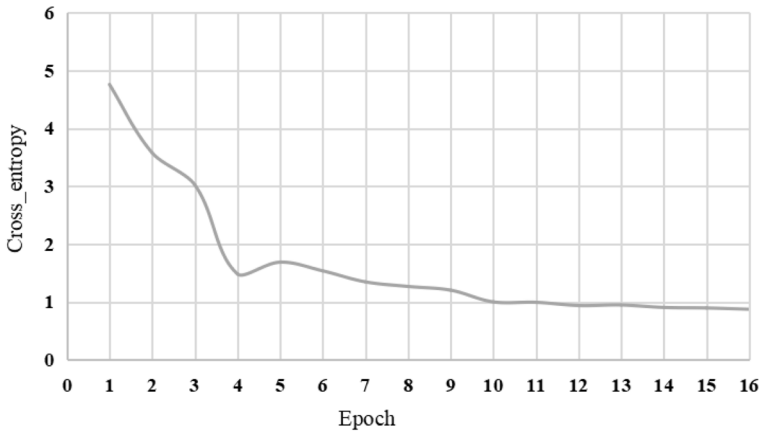
The abscissa in Fig. 6 represents the epoch, and the ordinate represents the cross-entropy, which is regarded as the loss function. The whole picture shows the change in cross-entropy after each epoch training. It can be seen from the Fig. 6a that on the CIFAR-10 dataset, as the training progresses, the value of the loss function continuously decreases. The loss function stabilizes and reaches a minimum at 13 epoch. But epoch is greater than 13, the value of the loss function becomes larger and no longer decreases. This is because the model may be overfitting. In order to get better accuracy, this paper chooses the model parameters when the epoch is 12 for testing. On the Tiny Image dataset, Fig. 6b shows that the loss function decreased steadily in the first few epochs. Although there are some slight fluctuations, the loss function is still converging towards the optimal solution. After the epoch is 10, the loss function is nearly unchanged and does not increase, which indicates that the model has reached the optimal solution. This article chose the training results at epoch 15 as the parameters of the model on the Tiny Image dataset.

Comparison of the proposed network with other networks

To demonstrate the performance of the proposed model in network compression while ensuring accuracy of classification, we compare the proposed network to other mainstream networks and illustrate their classification accuracy based on the dataset CIFAR - 10. The parameters of the proposed network are specified as follows: the dilated rate $\delta = 3$, width multiplier $\alpha = 1.0$, and resolution multiplier $\rho = 1$. The results are shown in Table 3.



(a) Loss function change in cifar-10 dataset



(b) Loss function change in the tiny image dataset

Fig. 6 Loss function change in different dataset

Table 3 shows that, compared to mainstream networks, the proposed network model is more accurate on the CIFAR-10 dataset. With the same width factor and the input image resolution of the MobileNet network, the proposed network retains a high accuracy while reducing the number of network parameters compared to MobileNet and GoogleNet. The SqueezeNet model typical acquires fewer parameters, however, at the cost of low accuracy.

Table 3 The proposed network vs popular networks in CIFAR-10

Model	Classification Accuracy	Parameters (Million)
3 ×1.0 224 this paper	84.25%	4.1
1.0 MobileNet 224	83.91%	4.2
GoogleNet	83.84%	6.8
SqueezeNet	69.83%	1.25
VGG 16	86.17%	138

Although the proposed network requires more parameters than SqueezeNet, it is much better in terms of classification accuracy. Because SqueezeNet sacrifices classification accuracy, it is not suitable for practical applications requiring high accuracy. Therefore, in the compromise of classification accuracy and model size, the proposed network is superior to SqueezeNet model. By contrast, although the VGG16 network has slightly higher classification accuracy than the proposed network, its model size is dozens more than the proposed model, resulting in computational difficulty when computing power is limited. Due to fewer network parameters, the proposed network can be easily transplanted on mobile devices with less storage capacity while having better classification accuracy.

Different dilated rate

This study applies the dilated rate to control the size of the dilated convolution, which affects the size of the receptive field, and the receptive field will lead to the change of

Table 4 Classification accuracy of different dilated rates

Dilated Rate	Width Multiplier	Resolution Multiplier	Classification Accuracy
1	1.0	224	82.04%
2	1.0	224	83.32%
3	1.0	224	84.25%
4	1.0	224	83.56%

classification accuracy. Therefore, we compared the network classification accuracy under different dilated rates, as summarized in Table 4.

Table 4 shows the classification accuracy changes with the dilated rate given the width multiplier $\alpha = 1.0$, and resolution multiplier $\rho = 1$. It shows that the joint dilated convolution and the depthwise separable convolution improve classification accuracy by two percent compared to networks without joint convolutions on the dataset CIFAR-10. It also shows that the maximum classification accuracy is achieved when the dilated rate is 3. Note that the classification accuracy of the network decreases slightly as the dilated rate increases continue. As the dilated rate increases, the receptive field becomes larger, which may contain more global and semantic features. However, blindly expanding its receptive field will lose a lot of local and detailed information during the convolution process, affecting the classification accuracy of small targets and distant objects.

Accuracy after hyperparameter compression

This section is aimed to verify the classification accuracy when applying the width multiplier and the input resolution to compress the model after adding dilated rate. Figure 7 compares the classification accuracy of the proposed model with different width multiplier and input image resolution.

Figure 7 presents the classification accuracy of the proposed model under different dilated rates after further compression with hyperparameters. The triangle label indicates the change of network classification accuracy when $\alpha = 1.0$, $\rho = 1$; the square label indicates the change of the compression network classification accuracy when $\alpha = 1.0$ and $\rho = 0.8571$; the diamond label indicates the network classification accuracy when $\alpha = 0.75$ and $\rho = 1$. Figure 7 shows that the proposed network has improved the classification accuracy with the increasing of the dilated rate and using compression parameters to further compress the model will not affect the effectiveness of the proposed model. Comparing the results with different input resolutions when the width multiplier is constant, we can see that the increasing trend of the classification accuracy is not affected by the resolution of the input image. When the input image resolution is unchanged, the square label polyline and the diamond label polyline are compared. When the dilated rate increases from 1 to 3, we can see that the network reaches the maximum classification accuracy when the dilated rate is 3. In addition, the model accuracy of the width multiplier $\alpha = 1.0$ is increased from 82.04% to 84.25%, and the model accuracy of the width multiplier $\alpha = 0.75$ is improved from 78.75% to 80.35%. When the dilated rate is greater than 3, the network classification accuracy slightly decreases, but it is still better than the original network. Therefore, in order to make the network more effective, we have selected a dilated ratio of 3 in subsequent experiments. The classification accuracy has also improved. In summary, even if the model is further compressed by the width multiplier and the input picture resolution, the proposed method can improve the classification accuracy.

Result on different dataset

The results in previous sections show that the proposed network performs well on the CIFAR-10 dataset. To

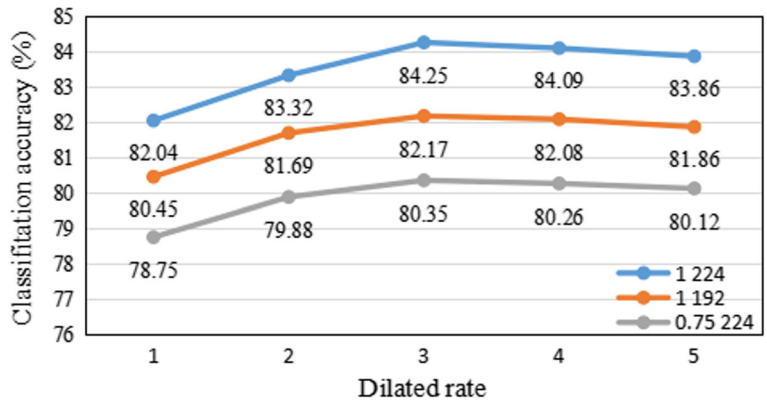


Fig. 7 Accuracy after hyperparameter compression

Table 5 Compare this paper network with popular networks in Tiny ImageNet dataset

Model	Accuracy
3 ×1.0 224 this paper	85.01%
1.0 MobileNet 224	83.81%
GoogleNet	82.94%

investigate the transferability of the proposed model, we conducted training and testing on Tiny ImageNet dataset.

Table 5 shows that the proposed network has good accuracy on Tiny ImageNet Dataset. Compared to the MobileNet with width multiplier $\alpha = 1.0$ and the picture size is 224×224, the proposed network improves the accuracy of both datasets. Compared to GoogleNet, the proposed network enhances the accuracy rate on Tiny ImageNet dataset from 82.94% to 85.01%. These comparisons demonstrate that the proposed network can consistently improve classification accuracy, indicating a good generalization ability. The proposed model also reduces the size under the premise of ensuring accuracy, which makes it possible to achieve better classification accuracy on mobile devices.

Table 6 shows the influence of different dilated rates on the classification accuracy of the model in the Tiny ImageNet dataset. As the dilated rate increases, the model accuracy increases from 81.73% to 85.01%. It shows that the proposed network improve classification accuracy by close to four percent compared to without dilated convolution on the dataset Tiny ImageNet. In addition, the best classification accuracy can be obtained when the dilated rate reaches 3. The results are the same as network in the CIFAR-10 dataset. Therefore, when use the proposed network in this article for testing or training, set the dilated rate to 3 to get the best classification accuracy. What is more, Fig. 7 shows that different dilated rate can effectively increase the robustness of the model. The proposed network in this paper can also improve the classification accuracy of the model on the different dataset and the proposed network has good generalization ability and good accuracy in different datasets.

Table 6 Classification accuracy of different dilated rates in Tiny ImageNet dataset

Dilated Rate	Width Multiplier	Resolution Multiplier	Classification Accuracy
1	1.0	224	81.73%
2	1.0	224	83.12%
3	1.0	224	85.01%
4	1.0	224	84.65%

Discussion

The model proposed is mainly used for image classification, aiming at balancing the network size and classification accuracy for a lightweight and efficient model. The experimental results on different datasets demonstrate that the proposed model has a good generalization ability and classification accuracy. In addition, the network proposed in this paper can be used as the basic network of SSD or YOLO models to realize pedestrian detection, or it can be transplanted to different devices to realize real-time pedestrian detection in portable devices [42, 43]. Applications developed on the basis of this model can convey additional practical values.

Conclusion

This paper proposes a lightweight neural network model combining dilated convolution and depthwise separable convolution. This joint module reduces the computational burden with depthwise separable convolution, making it possible to apply the network model to resources or computationally constrained devices. Meanwhile, the dilated convolution is used to increase the receptive field in the process of convolution without increasing the number of convolution parameters. It extracts global features and higher semantic level features in the convolution process, which improves classification accuracy. The hyperparameters (i.e., width multiplier and resolution multiplier) are used to further compress the model to be lightweight so that the proposed model can be applied to devices with limited computational power. Compared with the previous network, this paper combines dilated convolution and deepthwise separable convolution, which not only solves the problem that the calculation amount is too large to apply to resource-constrained equipment, but also solves the problem that model size and model classification accuracy cannot coexist. Experimental results demonstrate that the proposed model makes a good compromise between the classification accuracy and the model size while maintaining the classification accuracy when the network is compressed. Moreover, it uses hyperparameters and dilated rate to further compress the trained model effectively. The proposed network can greatly reduce the size and computation of the network, making it easier to transplant to devices. For example, the network can be transplanted in Android mobile devices, embedded devices such as MCU or FPGA [44, 45]. In addition, companies or individuals using the network proposed in this paper can reduce the performance of cloud computing servers and reduce the cost of renting cloud computing servers. At the same time, it can be seen from experiments that the amount of calculation and parameters of the lightweight network proposed in this article are quite small, which allows some companies to train on personal servers, which has better security.

Abbreviations
CPU: Central processing unit; CUDA: ComputeUnified device architecture;GPU: GraphicsProcessing unit; 2 ×1.0224 this paper: Width multiplier $\alpha = 2$, resolution multiplier $\rho = 1$ and the input picture size is 224 × 224

Acknowledgements
Authors thank editor and reviewers for their time and consideration.

Authors' contributions
All authors have participated in conception, drafting the article or revising it critically for important intellectual content, approval of the final version.

Authors' information
Not applicable.

Funding
This work is supported in part by the National Nature Science Foundation of China (No. 61304205, 61502240), Natural Science Foundation of Jiangsu Province (BK20191401), and Innovation and Entrepreneurship Training Project of College Students (201910300050Z, 201910300222).

Availability of data and materials
Not applicable.

Competing interests
This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue. The authors declare that they have no competing interests among authors.

Author details
[1]School of Automation, Nanjing University of Information Science and Technology, 210044 Nanjing, China. [2]Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology, 210044 Nanjing, China. [3] Jiangsu Engineering Center of Network Monitoring, 210044 Nanjing, China. [4]Department of Civil and Environmental Engineering, Rensselaer Polytechnic Institute, 12180 Troy, USA.

Received: 15 February 2020 Accepted: 11 September 2020
Published online: 23 September 2020

References

1. krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp 1097–1105
2. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations. IEEE, USA. pp 1–14
3. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 1–12
4. Xu X, He C, Xu Z, Qi L, Wan S, Bhuiyan M (2020) Joint optimization of offloading utility and privacy for edge computing enabled iot. IEEE Internet Things J 7(4):2622–2629. https://doi.org/10.1109/JIOT.2019.2944007
5. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: European Conference on Computer Vision. Springer, German. pp 630–645
6. Xie S, Girshick R, Dollár P, Tu Z, He K (2017) Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 1492–1500
7. Zhou J, Hu X, Ma Y, Sun J, Wei T, Hu S (2019) Improving availability of multicore real-time systems suffering both permanent and transient faults. IEEE Trans Comput 68(12):1785–1801
8. Iandola F, Han S, Moskewicz M, Ashraf K, Dally W, Keutzer K (2017) Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5. In: International Conference on Learning Representations. IEEE, USA. pp 1–13
9. Zhou J, Sun J, Zhou X, Wei T, Chen M, Hu S, Hu X (2018) IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 38(12):2215–2228
10. Xu X, Cai Q, Zhang G, Zhang J, Tian W, Zhang X, Liu A (2018) An incentive mechanism for crowdsourcing markets with social welfare maximization in cloud-edge computing. Concurrency Comput: Pract Experience:4961. https://doi.org/10.1002/cpe.4961
11. Li J, Cai T, Deng K, Wang X, Sellis T, Xia F (2020) Community-diversified influence maximization in social networks. Information Systems 92:1–12
12. Zhou J, Sun J, Cong P, Liu Z, Zhou X, Wei T, Hu S (2020) Security-critical energy-aware task scheduling for heterogeneous real-time mpsocs in iot. IEEE Trans Serv Comput 13(4):745–758. https://doi.org/10.1109/TSC.2019.2963301
13. Guo Y, Wang J, Peeta S, Anastasopoulos P (2020) Personal and societal impacts of motorcycle ban policy on motorcyclists' home-to-work morning commute in china. Travel Behav Soc 19:137–150
14. Guo Y, Peeta S (2020) Impacts of personalized accessibility information on residential location choice and travel behavior. Travel Behav Soc 19:99–111
15. Ramlatchan A, Yang M, Liu Q, Li M, Wang J, Li Y (2018) A survey of matrix completion methods for recommendation systems. Big Data Mining and Analytics 1(4):308–323
16. Zhang C, Yang M, Lv J, Yang W (2018) An improved hybrid collaborative filtering algorithm based on tags and time factor. Big Data Mining and Analytics 1(2):128–136
17. Han S, Mao H, Dally W (2016) Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In: International Conference on Learning Representations. IEEE, USA. pp 1–14
18. Han S, Pool J, Tran J, Dally W (2015) Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems. Springer, German. pp 1135–1143
19. Ghemawat S, Gobioff H, Leung S-T (2003) The google file system. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles. IEEE, USA. pp 29–43
20. Chollet F (2017) Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 1251–1258
21. Howard A, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. In: International Conference on Learning Representations. IEEE, USA. pp 1–9
22. Kumar S, Singh M (2018) Big data analytics for healthcare industry: impact, applications, and tools. Big Data Mining and Analytics 2(1):48–57
23. Chang F, Dean J, Ghemawat S, Hsieh W, Wallach D, Burrows M, Chandra T, Fikes A, Gruber R (2008) Bigtable: A distributed storage system for structured data. ACM Trans Comput Syst (TOCS) 26(2):1–26
24. Liu Y, Wang S, Khan M, He J (2018) A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering. Big Data Mining and Analytics 1(3):211–221
25. Xu X, Mo R, Dai F, Lin W, Wan S, Dou W (2019) Dynamic resource provisioning with fault tolerance for data-intensive meteorological workflows in cloud. IEEE Trans Ind Inform. https://doi.org/10.1109/TII.2019.2959258
26. Dean J, Ghemawat S (2008) Mapreduce: simplified data processing on large clusters. Commun ACM 51(1):107–113
27. Xu X, Liu X, Xu Z, Wang C, Wan S, Yang X (2019) Joint optimization of resource utilization and load balance with privacy preservation for edge services in 5g networks. Mobile Netw Appl:1–12. https://doi.org/10.1007/s11036-019-01448-8
28. Yu F, Koltun V (2016) Multi-scale context aggregation by dilated convolutions. In: International Conference on Learning Representations. IEEE, USA. pp 1–13
29. Wang L, Zhang X, Wang R, Yan C, Kou H, Qi L (2020) Diversified service recommendation with high accuracy and efficiency. Knowledge-Based Systems:106196. https://doi.org/10.1016/j.knosys.2020.106196
30. Xu X, Zhang X, Khan M, Dou W, Xue S, Yu S (2020) A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems. Futur Gener Comput Syst 105:789–799
31. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 7132–7141
32. Zhang X, Zhou X, Lin M, Sun J (2018) Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 6848–6856

33. Guo Y, Wang J, Peeta S, Anastasopoulos P (2018) Impacts of internal migration, household registration system, and family planning policy on travel mode choice in china. Travel Behav Soc 13:128–143
34. Chen Y, Zhang N, Zhang Y, Chen X, Wu W, Shen X (2019) Energy efficient dynamic offloading in mobile edge computing for internet of things. Trans Cloud Comput. https://doi.org/10.1109/TCC.2019.2898657
35. Zhong W, Yin X, Zhang X, Li S, Dou W, Wang R, Qi L (2020) Multi-dimensional quality-driven service recommendation with privacy-preservation in mobile edge environment. Comput Commun 157:116–123. https://doi.org/10.1016/j.comcom.2020.04.018
36. Qi L, He Q, Chen F, Zhang X, Dou W, Ni Q (2020) Data-driven web apis recommendation for building web applications[j]. IEEE Trans Big Data. https://doi.org/10.1109/TBDATA.2020.2975587
37. Han S, Mao H, Dally W (2015) Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149
38. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning. IEEE, USA. pp 448–456
39. Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, et al (2018) Recent advances in convolutional neural networks. Pattern Recog 77:354–377
40. Liu H, Kou H, Yan C, Qi L (2020) Keywords-driven and popularity-aware paper recommendation based on undirected paper citation graph[j]. Complexity:1–15. https://doi.org/10.1155/2020/2085638
41. Kingma D, Ba J (2015) Adam: A method for stochastic optimization. In: International Conference on Learning Representations. IEEE, USA. pp 1–15
42. Liu H, Kou H, Yan C, Qi L (2019) Link prediction in paper citation network to construct paper correlation graph. EURASIP J Wirel Commun Netw 233:1–12. https://doi.org/10.1186/s13638-019-1561-7
43. Hu H, Peng R, Tai Y-W, Tang C-K (2016) Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. In: International Conference on Learning Representations. IEEE, USA. pp 1–9
44. Qiu J, Wang J, Yao S, Guo K, Li B, Zhou E, Yu J, Tang T, Xu N, Song S, et al. (2016) Going deeper with embedded fpga platform for convolutional neural network. In: Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. IEEE, USA. pp 26–35
45. Wu J, Leng C, Wang Y, Hu Q, Cheng J (2016) Quantized convolutional neural networks for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, USA. pp 4820–4828

Publisher's Note