IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 12, 2020, accepted September 7, 2020, date of publication September 14, 2020,
date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023746

Human Segmentation Based on Compressed Deep Convolutional Neural Network

JUN MIAO[1,2,3], KEQIANG SUN[3], XUAN LIAO[3], LU LENG[4,5], (Member, IEEE), AND JUN CHU[4]

[1]Key Laboratory of Nondestructive Testing (Ministry of Education), Nanchang Hangkong University, Nanchang 330063, China
[2]Key Laboratory of Lunar and Deep Space Exploration, National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100101, China
[3]School of Aeronautical Manufacturing Engineering, Nanchang Hangkong University, Nanchang 330063, China
[4]School of Software, Nanchang Hangkong University, Nanchang 330063, China
[5]School of Electrical and Electronic Engineering, College of Engineering, Yonsei University, Seoul 120749, South Korea

Corresponding author: Lu Leng (leng@nchu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61661036, Grant 61866028, and Grant 61663031; in part by the Open Foundation of Key Laboratory of Nondestructive Testing (Ministry of Education), Nanchang Hangkong University, under Grant ZD201529003; in part by the Open Foundation of Key Laboratory of Lunar and Deep Space Exploration, Chinese Academy of Sciences (CAS), under Grant LDSE201705; in part by the Key Program Project of Research and Development, Jiangxi Provincial Department of Science and Technology, under Grant 20171ACE50024 and Grant 20192BBE50073; and in part by the Foundation of China Scholarship Council under Grant CSC201808360294 and Grant CSC201908360075.

ABSTRACT Most semantic segmentation models based on deep convolutional neural network (CNN) typically require a large number of weight parameters, high hardware resources for storage and computation. Moreover, redesigning a compact network suffers from some training problems, such as under-fitting. A human segmentation algorithm is proposed based on compressed deep CNN to optimize the convolution layers and filters. PSPNet-50 is fine-tuned on the human segmentation dataset to obtain the human segmentation model with higher accuracy. Then the convolutional-layer level pruning and corresponding structure optimization are performed so that the parameters of the model are substantially reduced. Finally, the two-stage global filter-level pruning strategy is used. Compared with the method of layer by layer pruning and retraining, our strategy not only reduces parameters of the model and saves the time of retraining, but also keeps the high IoU (Intersection over Union) accuracy. In addition, by adding auxiliary losses in the network during training CNN, the supervised training of the network is improved, and IoU is further increased. Compared to the model before compression, the sufficient experiments show that the parameter number, computation cost, memory consumption, and parameter storage are decreased by 1/7.5, 5.6/6.6, 0.7/1, 6.5/7.5, respectively, while the segmentation speed is accelerated by 2.4 times, and IoU on test set reaches 93.2%.

INDEX TERMS Compressed deep convolutional neural network, human segmentation, convolutional-layer level pruning, filter level pruning.

I. INTRODUCTION

Human segmentation refers to segmenting human regions from the background. As the foundation of analysis and understanding of human behavior, segmentation results are important to the subsequent works, such as 3D reconstruction, recognition, detection, and tracking.

The traditional methods of image segmentation based on digital image processing and mathematical morphology are not robust enough to resist the noise, and the accurate segmentation typically requires a lot of human-computer interaction. With the rapid development of deep learning, especially convolutional neural network, image segmentation methods based on deep learning outperform traditional methods in many aspects. The models trained with these segmentation algorithms can be used for portrait images; however, it is difficult to obtain high IoU because they are not specifically trained for human body images. Moreover, deep learning network models need to calculate convolution on the feature maps with the large size. The number of weight parameters of the segmentation model is substantially large, accordingly the segmentation speed is slow, and requirements

The associate editor coordinating the review of this manuscript and approving it for publication was Yongjie Li.

of computation and storage are high. Human photograph has become the main application of mobile devices such as mobile phones, whereas human segmentation requires high accuracy and real-time performance.

In this article, a human segmentation algorithm is proposed based on a compressed deep CNN, which reduces the resources for computation and storage while yields high IoU. Firstly, PSPNet-50 [1] is specifically fine-tuned for human segmentation, which is not only for preserving the compactness of the original network structure, but also for avoiding the underfitting. Then, the model is pruned and simplified at convolutional-layer-level and filter-level. The convolutional layers and filters that have little impact on accuracy are pruned, so the amount of parameters and calculation is reduced while preserving high IoU. The sufficient experiments on the human segmentation dataset demonstrate that our algorithm outperforms the commonly used models in terms of model size, segmentation speed and accuracy.

The remainder of this article is organized as follows: Section 2 presents a review of related works. Section 3 describes fine-tune of PSPNet-50 for human segmentation. Section 4 introduces the compression and acceleration of our method. Section 5 describes the compressed DCNN training for human segmentation. Experiments are displayed in Section 6, followed by a conclusion in Section 7.

II. RELATED WORKS

There are many traditional human segmentation methods, which can be briefly divided into brush-based, graffiti-based, and border-based methods. Brush-based methods are typically based on graph cut [2], [3], geodesic distance [4], random walk [5], and fully connected conditional random field [6]. They require users to specify some foreground and background as the boundary conditions by using a brush, and then optimize the solution. In contrast to the brush-based method, the graffiti-based method [7] only requires users to simply paint on the object. The boundary-based method [8], [9] calculates the boundary of the object by tracking the rough boundary input by users. Although the applications of these methods in image processing software are very popular, the tedious and complex interaction limits their application in automatic image processing.

With the rapid development of deep learning, many semantic segmentation methods based on deep CNN have emerged, such as FCN [10], DeepLab [11], PSPNet [1], MDCCNets [12], etc., which can automatically learn the deep features of object representation, and yield better results than the traditional segmentation methods. These deep learning models can be used for human segmentation; however, the accuracies are not high since they are not specifically trained for human images. Wang *et al.* [13] proposed a human segmentation method based on FCN structure combining with deconvolution.

Based on FCN, Shen *et al.* [14] added the position and shape information of portraits into the network training to improve human segmentation. However, deep network models typical require more hardware for the substantial computational complexity and storage. For example, the widely used VGG-16 [15] requires 138 million parameters to be trained. It requires more than 15G (10^9) of floating operations (FLOPs) to classify a 224×224 image and storage space of 500 MB. Thus it is difficult to deploy on the devices with limited resources, such as mobile phones.

Therefore, many methods were proposed for the compression and acceleration of classification network, which can be divided into quantization-based methods, pruning-based methods, and the methods for designing compact networks. Quantization-based methods are to quantify each weight of the network into an element in a finite set. Zhou *et al.* [16] proposed an incremental network quantization, which gradually converted a pre-trained full-precision floating-point neural network into a lossless low-bit binary model. Gong *et al.* [17] directly performed k-means clustering on the weights of the network. Each weight was represented by a cluster center, which can realize a high compression ratio. These quantified methods can reduce the storage consumption, but the speed of the neural networks has not been greatly accelerated.

Pruning-based methods [18]–[22] can reduce the storage and computation by removing some connections in the neural network. Through linear discriminant analysis, Tian *et al.* [23] found that a lot of filters are highly uncorrelated for facial gender classification in the last convolution layer of VGG-16 model [15], so this can be used to remove those filters with high within-class variance and low between-class variance. Some filter-level pruning strategies have been developed by evaluating neuron importance. Li *et al.* [24] measured the importance of each filter by calculating its absolute weight sum. The whole filters with the small sum values are removed from the network together with their connecting feature maps. This approach does not result in sparse connectivity patterns, and the computation costs are reduced significantly. ThiNet [25] pruned the filters of the network based on the statistical information of the next layer, and achieved 16.63 times of the parameter compression on VGG-16. To evaluate the importance of each filter in the network, Liu *et al.* [26] employed the scaling factor parameter of the Batch Normalization (BN) layer [27]. The filters with the scaling factors close to zero were considered unimportant and could be deleted. Chen *et al.* [28] combined a weight-based pruning with an adaptive architecture squeezing to obtain a high compression ratio in CNN, and utilized pruning to find an appropriate squeezing ratio. In [29], a lossless lightweight CNN design strategy is explored for the SAR target recognition by using the structured pruning and the knowledge distillation. Pruning methods can reduce the amount of parameters while reducing the computational complexity, but it is possible that the accuracy is degraded due to the retraining of each network layer iteratively.

Designing compact network architecture can achieve low storage cost, low computational complexity and high network performance. Since the classical deep networks, including Alex-Net [30] and VGG16, require a large amount of

computation resources, some researchers proposed more efficient neural network architectures, such as residual network (ResNet) [31], which reduce model parameters but maintain or even improve the performance of network models. SqueezeNet [32] further simplified the weight parameters by replacing the 3×3 filter with the 1×1 filter and reducing the number of channels of the 3×3 filter. Unfortunately, redesign a compact network still encounters with the underfitting problem.

III. HUMAN SEGMENTATION MODEL BASED ON PSPNet-50

The first 50 layers of PSPNet-50 [1] use a 50-layer residual network [32] as the feature extractor to generate the feature maps in the various levels by pyramid pooling with various pooling kernel sizes. Since fusion can improve the performance [33]–[38], these features are fused under four different pyramid scales as a pyramid module, and then the different levels of features obtained by the pooling are fused. An upsampling and convolutions are performed to form the final feature representation, which achieve end-to-end output of both local and global context information. In this article, we first fine-tune PSPnet-50 neural network for human segmentation.

A. TRAINING AND TESTING DATASETS

The segmentation dataset is derived from Baidu dataset [36], which contains 5,387 human images of various poses and their segmentation labels in various scenes. According to our observation, a few segmentation labels in the dataset are incorrect or inaccurate, so we remove these segmentation labels and the corresponding images to avoid degrading the training, and finally 4,512 human images and their segmentation labels are selected. 4,112 images are training samples, and the remaining 400 images are test samples. Each of image size is 473×473.

B. MODEL TRAINING

We assign the number of output feature maps of the last convolutional layer in PSPNet-50 [1] to 2, which correspond to the background and foreground (human), respectively, and then use the training set to fine-tune the human segmentation model.

When the network performs forward propagation, the average value of the prediction error sum corresponding to all pixels is calculated as the training error, and the weight parameters are updated according to the minimizing training error/loss. The loss in training:

$$\mathcal{L} = \frac{1}{n}\sum_{i} \ln[p(x_i = k)], \quad i = 0, 1, \ldots, n-1 \tag{1}$$

$$p(x_i = k) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}, \quad k = 0, 1, \ldots, K-1 \tag{2}$$

In Eq. (1) and (2), $\mathcal{L}$ denotes the model training loss. $p(x_i = k)$ is the probability that the i-th pixel x_i belongs to the k-th category. n is the number of pixels in the current training image. z_k is the eigenvalue of the k-th category. The bigger the value of z_k is, the higher the probability that the pixel x belongs to the k-th category. z_j the j-th element of the eigenvector z, which is turned into probabilities by softmax function Eq. (2). K is the total number of categories. In this article, there are only two categories, namely human and background, so $K = 2$.

Stochastic gradient descent (SGD)[37] is used to update the weight parameters by a linear combination of the negative gradient and the last weight update value:

$$\boldsymbol{V}_{t+1} = \mu \boldsymbol{V}_t - \alpha \nabla L(\boldsymbol{W}_t) \tag{3}$$

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + \boldsymbol{V}_{t+1} \tag{4}$$

In Eq. (3) and (4), $\boldsymbol{W}_t$ is the weight matrix in the t-th iteration calculation. $\boldsymbol{V}_t$ is the weight update value matrix in the t-th iteration calculation. α is the basic learning rate of the negative gradient. μ is the weight of $\boldsymbol{V}_t$, which is used to weight the effect of the previous gradient direction on the current gradient descent direction. Here $\mu = 0.9$.

During iterative calculating, in order to accelerate the convergence speed of model loss, the learning rate is adjusted as:

$$\mathcal{LR} = base\_lr \cdot \left(1 - \frac{iter}{max\_iter}\right)^{power} \tag{5}$$

In Eq. (5), $\mathcal{LR}$ denotes the actual learning rate. $base\_lr$ is the basic learning rate, max_iter is the maximum number of iteration. $iter$ is the current number of iterations. $power$ is the learning rate parameter. Here $base\_lr = 0.0001$, and $power = 0.9$. In addition, to realize data enhancement, training images are horizontally mirrored and flipped during training to improve the generalization ability.

IV. COMPRESSION AND ACCELERATION

The strategies for compression and acceleration are as follows. The hierarchical structure pruning is applied on the model that is initially trained in Section 3, and then this model is retrained on the human dataset. Finally, filter-level pruning is used on the retrained model.

A. HIERARCHICAL STRUCTURE PRUNING

As shown in Figure 1(a), there are 4 parallel network paths in the initial pyramid pooling module obtained by fine-tuning PSPNet-50, which contains a complicated structure and requires a large amount of computation. We prune the convolutional layer structure by removing 4 parallel convolutional layers, and then merge the parallel output features of the pyramid pooling module by using Eltwise instead of Concat (as shown in Figure 1(b)). In Figure 1(a), the first upsample interpolates the feature maps to the 1/4 size of the original image, and the second one interpolates the feature maps to the size of the original image. In Figure 1(b), the first, the second, and the third upsamples interpolate the feature maps to the 1/4 size, the 1/2 size, and the same size of the original image respectively. It means that the

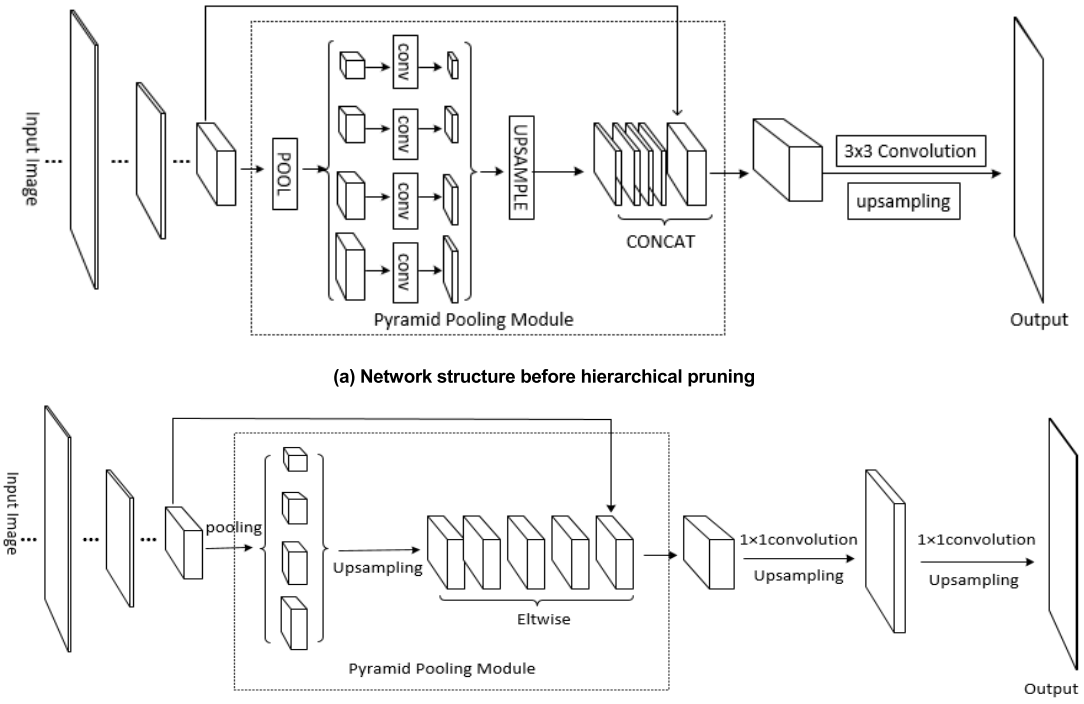

FIGURE 1. Network structure before and after hierarchical pruning.

concatenating operation of the 4 parallel output features is replaced with the merging operation, i.e. the sum of the eigenvalues. The reason is that the numbers of the filter parameters of these 4 parallel convolutional layers are the largest among all the convolutional layers except the penultimate convolutional layer. Figure 2 shows that the number of parameters of each convolutional layer in the PSPNet-50 network. The number of parameters for the 4 parallel convolution layers reached 4.2M (million). The removal of the 4 parallel convolutional layers can also ensure that the numbers of input channels of multiple bottoms in the Eltwise layer are equal so that the sum operation can be performed. This process not only reduces the number of channels in the output feature maps of the pyramid pool module by half and speeds up the computation, but also substantially decreases the number of weight parameters. In addition, because the number of weight parameters of the penultimate convolution layer is the largest among all the convolution layers in the model, the filter size of the penultimate convolution layer is changed from 3×3 to 1×1 so that the number of parameters in this layer is decreased from 18.9M to 2.1M.

Table 1 shows comparisons of the segmentation speed, parameter storage, and IoU before and after the pruning of the hierarchical structure. IoU calculates the intersection ratio of the target human region and the human region predicted by the segmentation network. Results show that the segmentation speed is accelerated after the hierarchical structure pruning, while the parameter storage is decreased by 47% and IoU is still high.

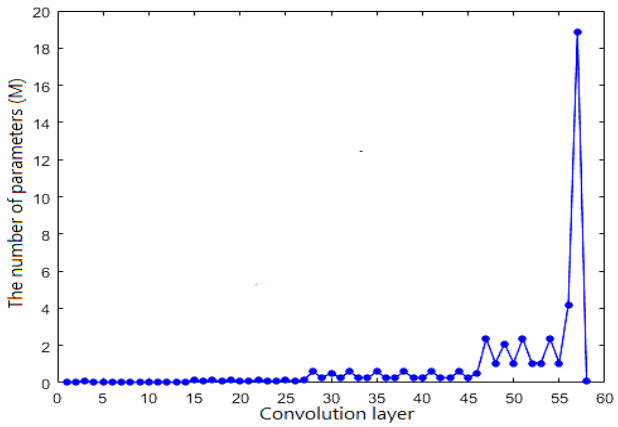

FIGURE 2. Parameter number of each convolutional layer of human segmentation network.

TABLE 1. Comparison of the performance of the hierarchical structure before and after pruning.

Method	Split speed (fps/s)	Parameter storage (MB)	IoU (%)
Before pruning	12.4	187.1	94.8
After pruning	14.7	99.0	94.8

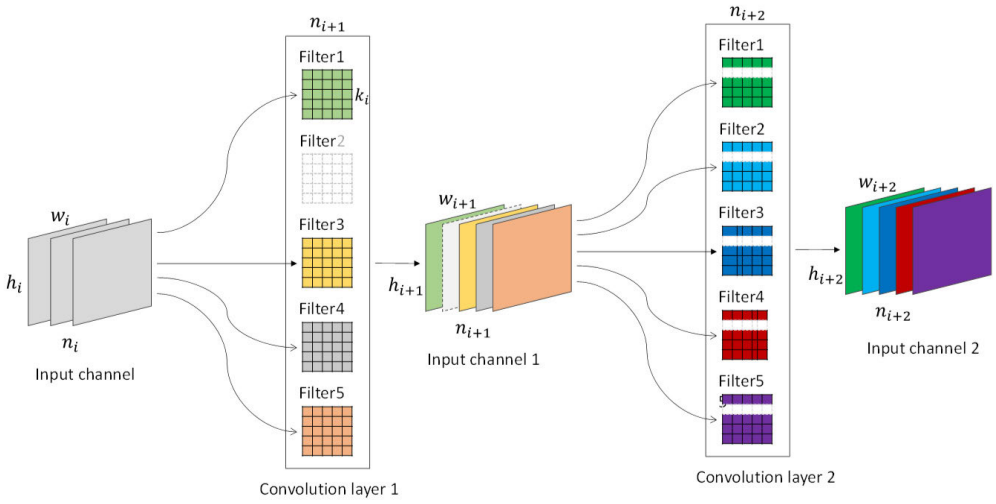

FIGURE 3. The impact of a filter removed.

B. FILTER-LEVEL PRUNING

Generally, the smaller the absolute value of the weight, the lower the importance, and the smaller the impact on the convolution result [38], [39]. To compress the network with high efficiency, reduce the number of weight parameters, and speed up the computation, we do not prune a single weight [38] because this strategy typically results in the sparse connection. However, graphics processor (GPU) is not good at dealing with sparse matrix. In other words, the number of the valid parameters is reduced, while the network speed is not accelerated, and the storage cost is not reduced. Therefore, the filter pruning approach in [24] is applied for further compression. The relative importance of a filter in each layer is measured by calculating the sum of its absolute weights. Those unimportant filters are removed directly. The filter level pruning belongs to structural pruning, which avoids sparse matrix, substantially reduces number of parameters, and accelerates speed.

Figure 3 shows the effect of pruning/removing a filter. Suppose that the number of input channels of the i-th convolution layer is n_i, and the size of the input feature map width is $h_i \times w_i$. We use n_{i+1}3D filters $F_{i,j} \in R^{n_i \times k_i \times k_i}$ ($k_i \times k_i$ is the 2D filter size) to perform convolution on the input n_i channel feature maps $x_i \in R^{n_i \times h_i \times w_i}$ to get n_{i+1} channel feature maps $x_{i+1} \in R^{n_{i+1} \times h_{i+1} \times w_{i+1}}$. The number of multiplications performed by this convolution is $n_{i+1} \times n_i \times k_i^2 \times h_{i+1} \times w_{i+1}$. Therefore, removing a 3D filter in the convolution layer reduces a feature map, reduces $n_i \times k_i \times k_i$ weight parameters that need to be trained and stored, and reduces $n_i \times k_i^2 \times h_{i+1} \times w_{i+1}$ operations refers to the number of multiplication operations between numbers. Moreover, the number of channels in input feature maps of the next convolution layer is also reduced by one, and $n_{i+2} \times k_{i+1} \times k_{i+1}$ weight parameters that need to be trained and stored are reduced, and $n_{i+2} \times k_{i+1}^2 \times h_{i+2} \times w_{i+2}$ operations are canceled. Consequently, the total reduced number of weight parameter is $n_i \times k_i \times k_i + n_{i+2} \times k_{i+1} \times k_{i+1}$, which is larger than $1/n_i$ of the total parameter number of the convolutional layers. The total number of reduced operations is $n_i \times k_i^2 \times h_{i+1} \times w_{i+1} + n_{i+2} \times k_{i+1}^2 \times h_{i+2} \times w_{i+2}$, which is larger than $1/n_{i+1}$ of the total number of calculations for this convolutional layer. Therefore, removing the filters in convolutional layers not only largely decreases the number of weight parameters and the storage consumption, but also reduces computation costs and increases computation speed.

When pruning the filters, it is necessary to measure the importance of each filter in a convolutional layer, and remove the less important filters in each convolutional layer. For each filter of a convolution layer, the L1 norm of the filter weights is computed, which is the sum of the absolute values of the filter weights. The L1 norms of all filters of a convolution layer are sorted in descending order. Generally, if the L1 norm of the filter is small, its weight parameters are small, and accordingly this filter relatively weakly impacts on the final performance in practice. Thus the filters with small L1 norms can be removed. The process of pruning the m-th filter in the i-th convolution layer is as follows:

Step 1. Calculate L1 norm $s_j = \sum_{l=1}^{n_i} \sum |k_l|$ for each filter $F_{i,j}$, and sort the filters in descending order according to s_j.

Step 2. The m filters with small L1 norms and their corresponding feature maps are pruned, and the kernels corresponding to the pruned feature map in the next convolutional layer are also removed.

Step 3. Build a new kernel matrix for the i-th and $i+1$-th layers, and copy the remaining kernel weights to the new model.

Convolutional layers of our model are followed by batch normalized (BN) layers. BN layers regularize the input features, and the features in each channel correspond to a trainable parameter. When a filter of the convolution layer is

deleted, its corresponding output feature channel disappears. Therefore, when the convolution layers are pruned at filter levels, the parameters of the corresponding channels of BN layers should also be removed.

Our model is based on the residual network (ResNet) [31], which is more complicated than FCN, and require to consider the pruning of the residual module. As shown in Figure 4, since the sum operation is performed on the output feature maps of the convolutional layer after pruning in the residual block and the output feature maps of the corresponding convolutional layer, the pruning of the convolutional layer on one side must be taken as the benchmark to conduct the corresponding pruning on the other side. Based on the pruning of the convolutional layer on one side, the corresponding pruning is performed on the other side. Since the feature maps output from the shortcut layer of ResNet are the master features learned by the network, which are more important than the added residual maps, the feature maps to be pruned should be determined by the pruning results of the shortcut layer, and the residual block is pruned with the same filter index as selected by the pruning of the shortcut layer.

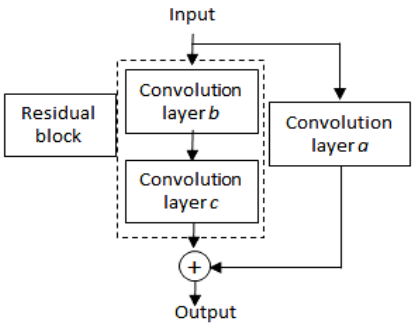

FIGURE 4. Residual module.

C. TWO-STAGE FILTER PRUNING STRATEGY

We do not use layer-by-layer pruning and fine-tuning retraining [25], [38] to compress the model, because it is too time-consuming and labor-intensive to train the network of deep architecture layer by layer. Thus we prune all the weight parameters at one time, and then train the model.

To quickly reduce the complexity of the network structure without degrading the accuracy, a two-stage pruning method is performed. We do not directly compress the model to a specified size because the model would jump too much and be underfitted. As shown in Table 2, if the model is directly compressed by 1/2, the accuracy decreases to 87.7%, and if the remaining ratio is assigned a smaller value, the accuracy decreases more sharply.

To balance the remaining ratio and the accuracy, the number of filters in all convolutional layers is compressed by 1/4 in the first pruning stage, and then the model is retrained on the dataset. In the second stage, the retained model is further compressed by a half, and then retrained to ensure the stable compression. It can be seen from Table 3 that the two-stage compression is a simple but highly effective method, and it yields higher IoU than the compression model obtained by direct compression.

TABLE 2. Remaining ratio and model accuracy during filter pruning.

Remaining ratio	1	0.9	0.8	0.7	0.6
Model accuracy (%)	94.8	94.3	93.4	91.9	89.8
Remaining ratio	0.5	0.4	0.3	0.2	0.1
Model accuracy (%)	87.7	79.3	74.1	65.9	53.6

TABLE 3. Accuracy comparison of different compression schemes.

Method	IoU
Direct compress by a half	87.7%
Two-stage compression by a half	92.6%

V. TRAINING OF COMPRESSED HUMAN SEGMENTATION NETWORK

The training of the compressed human segmentation network is similar to the training before compression. Moreover, we add two auxiliary losses to the loss function as the extra supervision during training to improve the accuracy.

A. AUXILIARY LOSSES

Many researches show that neural network structure with deeper layers can achieve better performance. However, as the network deepens, it also encounters with optimization difficulties [40], [41]. ResNet effectively releases this problem by residual learning. Therefore, our network training is further optimized based on ResNet framework. An initial segmentation is generated in training process, and the supervised learning is conducted by an auxiliary loss, the output loss of the final network is used to supervise the network training. Therefore, the learning optimization of deep network is decomposed into two parts that are easier to solve. Auxiliary losses1 and losses2 in Figure 5 are added to improve the training.

Figure 5 shows our supervised learning approach of the deep network. In addition to the Softmax cross entropy loss of the output of the last network layer, the feature maps output from the 4th and 5th parts in resnet-50 which is used in ours network are respectively passed through a convolution layer and an upsampling layer to obtain the corresponding classification output (score maps), which are used to calculate the auxiliary Softmax cross entropy loss. Each of these three losses acts on the resnet-50 network. The auxiliary losses help optimize the learning process, and the three losses are weighted respectively to obtain a more reasonable total loss.

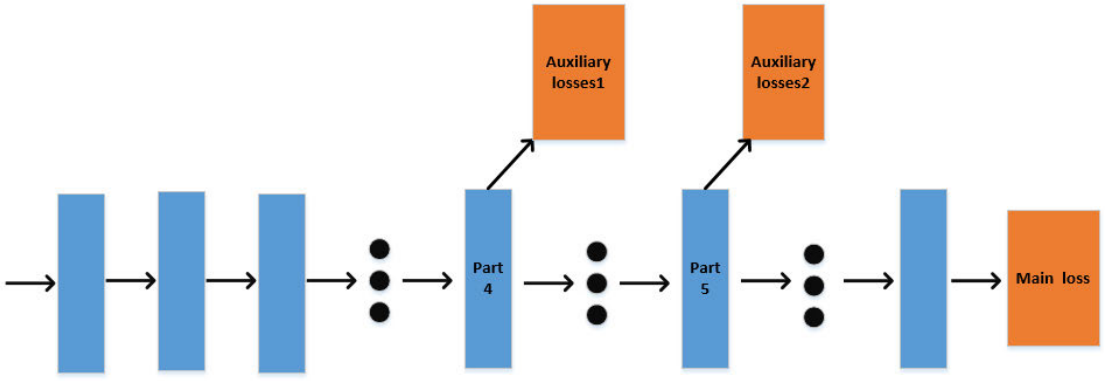

FIGURE 5. Supervised learning of the deep network.

The loss function of the whole network is:

$$\mathcal{L}(x) = \alpha\mathcal{L}_1(x) + \beta\mathcal{L}_2(x) + \gamma\mathcal{L}_3(x) \tag{6}$$

In Eq. (6), $\mathcal{L}_1(x)$, $\mathcal{L}_2(x)$, and $\mathcal{L}_3(x)$ are the master loss, auxiliary losses1, and auxiliary losses2. Each loss is calculated by Eq. (1) using the network's output score map. The normalized score of pixels belonging to the correct classification is found by the score graph, and the mean square error with 1 is the loss of the pixel. α, β, and γ denote the weights of the three loss. x denote output score map from each part of the network. We take four sets of values of α, β and γ for experiment, respectively $\alpha = 0.8$, $\beta = 0.1$, $\gamma = 0.1$; $\alpha = 0.6$, $\beta = 0.2$, $\gamma = 0.2$; $\alpha = 0.4$, $\beta = 0.3$, $\gamma = 0.3$; $\alpha = 0.2$, $\beta = 0.4$, $\gamma = 0.4$. It is found that the model achieves a higher accuracy at $\alpha = 0.4$, $\beta = 0.3$, $\gamma = 0.3$.

Part 4 and 5 above refer to part 4 and 5 of Resnet-50. The loss function is then used to train the network to minimize the total loss function $\mathcal{L}(x)$. Table 4 shows a comparison of the accuracy with and without the auxiliary losses. It is noted that the accuracy is improved by 0.6% after the auxiliary losses are added.

TABLE 4. Accuracy comparison between adding and not adding auxiliary losses.

Method	IoU
Without auxiliary losses	92.6%
With auxiliary losses	93.2%

VI. EXPERIMENTAL RESULTS AND ANALYSIS

A. SETUP

Our experiments are evaluated on Ubuntu 16.04, a 64-bit operating system. The deep learning framework for training is the open source framework Caffe (Convolutional Architecture for Fast Feature Embedding) of the Berkeley Visual Learning Center (BVLC). The hardware configuration includes Intel Xeon E5-2603 v3@1.60GHz 6 (CPU), NVIDIA GeForce GTX 1080Ti, memory: 11GB (GPU).

The evaluation is performed on Baidu dataset [36] and Flickr dataset [14]. There are 1800 portrait images in Flickr dataset, some of which have large variations in color, background, clothing, accessories, etc.

Our method is compared with some state-of-the-art methods in terms of model size, speed, and IoU accuracy. The evaluation includes three parts. Since our compression model is built form PSPNet-50, we compare fine-tuned PSPNet-50 before and after compression at Baidu dataset. We also compare our model with some segmentation approaches based on deep learning on Baidu dataset, including FCN-8S and DeepLab-v2. In addition, we compare our model with some compression models both on Baidu dataset and Flickr dataset.

B. COMPARISON BEFORE AND AFTER COMPRESSION

The comparisons of the human segmentation network based on fine-tuned PSPNet-50 before and after compression are shown in Table 5. The accuracy is slightly decreased from 94.8% to 93.2% due to three reasons. Firstly, in order to improve efficiency and reduce the training complexity, we directly retrain all the convolutional layers of the network after pruning rather than iterative training after pruning each convolutional layer. Training after one-time pruning only needs to be trained once, while layer-by-layer pruning requires training once again for each pruned convolutional layer. Although layer-by-layer pruning can attain better IoU accuracy, it has to consume much more time. Secondly, L1 norm is a little bit rough to measure of the importance of the filters. Thirdly, the importance of the filters in various convolutional layers are not the equal. Although it is convenient to simply compress them by the same proportion, it is possible that some relatively important filters are removed.

C. COMPARISON WITH UNCOMPRESSED SEGMENTATION NETWORKS

Our model is compared with some popular segmentation networks based on deep learning, which are also trained on human image dataset, for effective evaluation. Results are shown in Table 6. FCN-8S-VOC and PSPNet-50-ADE are

TABLE 5. Comparison of various indices before and after compression.

Indices	Before compression	After compression
Parameter amount (M)	46.7	6.2
Parameter ratio	7.5	1
Calculation (Gflops)	161.3	24.3
Calculated amount ratio	6.6	1
Speed (fps/s)	13.2	31.3
Speed ratio	1	2.4
RAM (GB)	2.5	1.5
Memory ratio	1.7	1
Storage (MB)	187.1	24.9
Storage space ratio	7.5	1
IoU Accuracy (%)	94.8	93.2

TABLE 6. Comparison with uncompressed segmentation networks.

Meth	Parm Number (M)	Comput Cost (Gflops)	Speed (fps/s)	Mem (GB)	Parm Storage (MB)	Accuracy IoU (%)
1[10]	134.5	165.0	21.9	1.8	538.0	70.8
2[1]	46.8	161.5	10.2	2.6	197.0	84.9
3[10]	134.3	73.3	25.4	1.6	537.1	90.3
4[11]	37.8	171.7	25.2	**1.5**	151.2	91.5
5[11]	127.9	257.2	4.9	9.6	513.2	**93.8**
6	**6.2**	**24.3**	**31.3**	**1.5**	**24.9**	93.2

Note: bold font indicates the best result. 'Meth', 'Parm', 'Comput' and 'Mem' are abbreviations of 'Method', 'Parameter', 'Computation', and 'Memory'. 'M' and 'G' denote means10^6 and 10^9, respectively. Computation cost is the calculation amount on the convolution layer. 'flops' is the number of floating-point operations. 'fps/s' is the number of frames processed per second. 1[10]: FCN-8S-VOC, 2[1]: PSPNet-50-ADE, 3[10]: FCN-8S_person, 4[11]: DeepLab-v2 _VGG-16, 5[11]: DeepLab-v2 _ResNet-101, 6: Our model.

the FCN [10] and PSPNet-50 [1] trained by PASCAL VOC dataset [42] and ADE20K dataset [43], respectively. Since FCN-8S-VOC and PSPNet-50-ADE are not trained on human images, their accuracies are not good. FCN-8S is a powerful architecture of the variations of FCN [10]. We train FCN-8S on the human image dataset to build FCN-8S-Person. We also use the human image dataset to train DeepLab-v2_VGG-16 [11] and DeepLab-v2_resnet-101 [11], which are the improved segmentation models based on VGG-16 [15] and resnet-101 [31], respectively. FCN-8S-VOC and PSPNet-50-ADE yield lowest and the second lowest accuracies because they are not trained on the human image dataset. Our model is the best compression architecture in terms of computational cost, parameter number, speed, memory consumption, and parameter storage. Although DeepLab-v2_resnet-101 yields 0.6% higher accuracy than our model, our model outperforms DeepLab-v2_resnet-101 in terms of the other 5 indices. For example, DeepLab-v2_resnet-101 requires more than 11GB memory for training with 473 × 473 images, which is run out of memory on our computer. In fact, we have to change the image size to 352 × 352, it still yields the lowest segmentation speed (4.9fps/s), which is difficult to run in real-time mode.

Figure 6 shows example the segmentation results on Baidu dataset. The models that are trained on the human image dataset perform much better than those that are not. In many cases, although FCN-8S-person is trained on the human image dataset, its edges are not accurate (shown in the rectangle boxes) because FCN model is not good at detail extraction. The segmentation performances of PSPNet-50 and DeepLab-v2 are improved with the fusion of various scale features. Despite our model is a compression model of PSPNet-50, it remains the pyramid pooling structure of PSPNet-50 to obtain global context information so that it has almost the same performance of the original PSPNet-50.

D. COMPARISON WITH COMPRESSION NETWORKS

Our model is compared with several state-of-the-art compression networks, including NISP [20], VGGNet-pruning [26], PSPNet-50-pruning [38], MBNet_DPLab [44] both on Baidu dataset and Flickr dataset. NISP prunes CNN by removing neurons with least importance, and then is fine-tuned to recover CNN predictive power. VGGNet-pruning is built from a VGGNet model, which is fine-tuned in [26] and trained on human image dataset. MBNet_DPLab is a lightweight semantic segmentation model that combines MobileNet-v2 and DeepLab-v3 [45] frameworks to simplify the design of mobile terminals. PSPNet-50-pruning is built from a PSPNet-50 model, which is fine-tuned in [38] and trained on human image dataset.

As shown in Table 7, MBNet_DPLab requires the lowest memory consumption and parameter storage, and has the highest speed, but the lowest accuracy. The reason is that DeepLab-v3 only compresses the weight parameters of the full connection layer. However, for the segmentation network without the full connection layer, DeepLab-v3 only prunes a small part of the weights of the convolutional layer to ensure the accuracy because the weights of the convolutional layer are more sensitive than those of the full connection layer. In our method, the convolution kernels of the filter are pruned to compress and accelerate. Filter level pruning belongs to structural pruning, which avoids the sparse connection caused by pruning a single weight, accordingly the graphics processor (GPU) does not need to process the calculation of sparse matrix, so our method can not only greatly reduce parameters, but also accelerate the running speed. Since our method only uses the L1 norm for filter pruning, it is not sufficient to fully

TABLE 7. Comparison the compression networks on Baidu dataset.

Method	Memory (GB)	Parameter storage (MB)	Speed (fps/s)	Accuracy IoU(%)
[20]	2.3	172.2	13.2	91.1
[26]	1.7	127.3	9.1	90.3
[38]	2.5	187.1	13.3	**93.8**
[44]	**1.5**	**23.0**	**52.1**	87.2
Ours	**1.5**	24.9	31.3	93.2

Note: [20]: NISP, [26]: VGGNet-pruning, [38]: PSPNet-50-pruning, [44]: MobileNet-V2_DeepLab-v3-VOC.

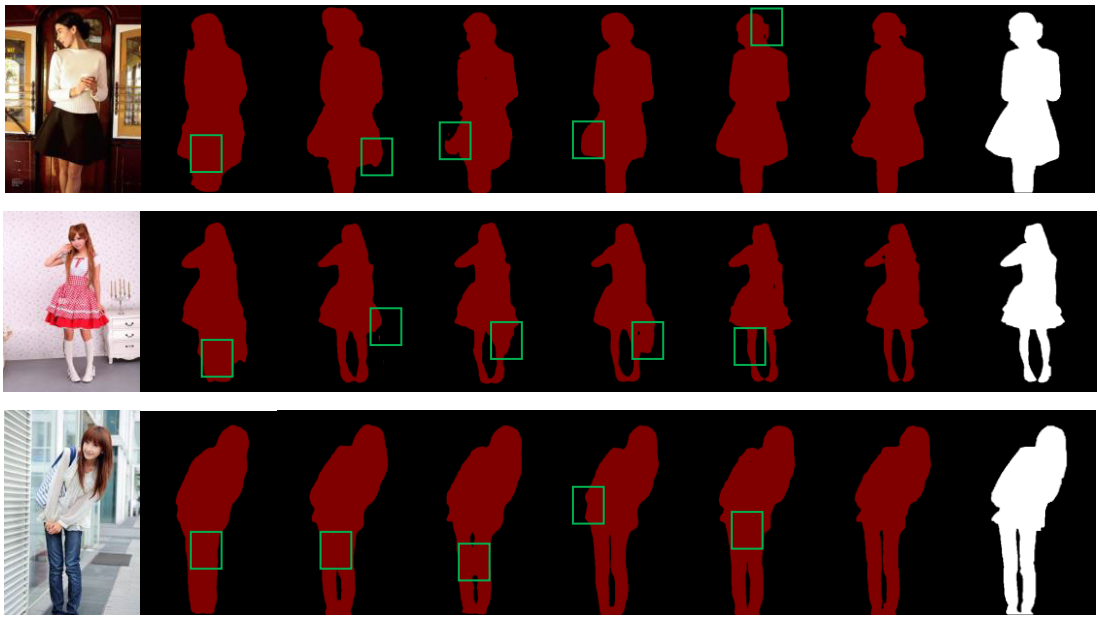

FIGURE 6. Segmentation results on Baidu dataset. (a): Original image, (b): FCN-8S _VOC, (c): PSPNet-50_ADE, (d): FCN-8S, (e): DeepLab-v2_VGG-16, (f): DeepLab-v2_ResNet-101, (g): Our model, (h): Ground Truth.

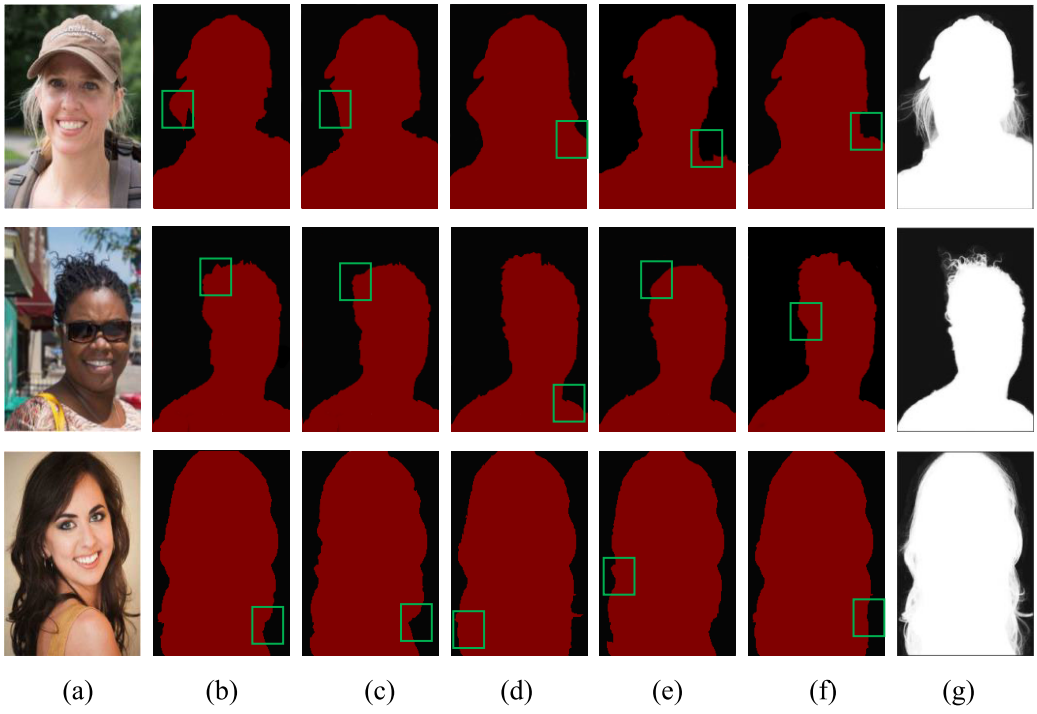

FIGURE 7. Segmentation results of different models on the Flickr dataset. (a): Original image, (b): NISP, (c): VGGNET-pruning, (d): PSPNet-50-pruning, (e): MobileNetV2_ DeepLab-v3-VOC, (f): Our model, (g): Ground Truth.

consider the importance of the filter. Although our model is not the best in all aspects, its performances are comprehensively optimized.

Figure 7 shows an example of the segmentation results on Flickr dataset. The failure details are displayed in rectangle boxes. Table 8 further reports a quantitative evaluation. In this experiment, all models are not retrained with portrait images, but tested directly with these images. Because of the strong generalization ability, our method still achieves competitive performance as shown on Baidu dataset. It can be seen from

TABLE 8. Comparison the compression networks on Flickr dataset.

Method	Memory (GB)	Parameter storage (MB)	Speed (fps/s)	Accuracy IoU(%)
[20]	1.8	134.8	16.9	91.9
[26]	1.3	97.3	11.9	90.9
[38]	1.9	142.1	17.5	**94.3**
[44]	**1.2**	**18.4**	**65.1**	87.9
Ours	**1.2**	19.9	39.1	93.6

Note: [20]: NISP, [26]: VGGNet-pruning, [38]: PSPNet-50-pruning, [44]: MobileNet-V2_DeepLab-v3-VOC.

Table 7 and Table 8 that all models attain better performance on Flickr dataset. It can explain that face and the upper part of the human body occupy the majority features in the portrait image, so it easier to segment face and the upper part than whole body.

VII. CONCLUSION

This article proposes a method for human segmentation based on compressed DCNNs. First, PSPNet-50 is fine-tuned on the human image dataset to obtain the initial segmentation model with a high accuracy, and then the convolutional-layer level pruning and structural optimization are performed on the initial model, which substantially reduces the number of parameter. Finally, a two-stage filter level pruning strategy is employed. Compared with the methods for pruning and retraining layer by layer, our method does not only decrease the retraining time and the parameters, but also guarantees a high accuracy. In addition, during the training of the network, by adding two auxiliary losses in the network, the supervised training of the network is better realized, and IoU is improved.

The segmentation model in this article is more conducive to the applications on mobile devices. However, the algorithm in this article can be further improved. IoU is degraded due to the compression. The further works will focus on the following two points: (1) Besides L1 norm, other indices will be used to judge the importance of the filter, (2) The number of the reserved filters of the convolutional layers can be different according to the importances of the convolutional layers.

REFERENCES

[1] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.
[2] Y. Y. Boykov and M. P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images," in *Proc. ICCV*, Jul. 2001, pp. 105–112.
[3] C. Rother, V. Kolmogorov, and A. Blake, "'GrabCut': Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.
[4] X. Bai and G. Sapiro, "Geodesic matting: A framework for fast interactive image and video segmentation and matting," *Int. J. Comput. Vis.*, vol. 82, no. 2, pp. 113–132, Apr. 2009.
[5] L. Grady, "Random walks for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1768–1783, Nov. 2006.
[6] K. Philipp and K. Vladien, "Efficient inference in fully connected crfs with Gaussian edge potentials," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 109–117.
[7] J. Y. Liu, J. Sun, and H. Shum, "Paint selection," *ACM Trans. Graph. (ToG)*, vol. 28, no. 5, p. 69, 2009.
[8] G. Zhu, "Boundary-based image segmentation using binary level set method," *Opt. Eng.*, vol. 46, no. 5, May 2007, Art. no. 050501.
[9] Y. Peng, S. Ruan, G. Cao, S. Huang, N. Kwok, and S. Zhou, "Automated product boundary defect detection based on image moment feature anomaly," *IEEE Access*, vol. 7, pp. 52731–52742, 2019.
[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.
[12] Q. Zhou, W. Yang, G. Gao, W. Ou, H. Lu, J. Chen, and L. J. Latecki, "Multi-scale deep context convolutional neural networks for semantic segmentation," *World Wide Web*, vol. 22, no. 2, pp. 555–570, Mar. 2019.
[13] P. Wang, Z. Fang, X. Zhao, and B. Huang, "Human body image segmentation algorithm based on deep learning," *J. Wuhan Univ. (Natural Sci. Ed.)*, vol. 2017, vol. 63, no. 3, pp. 466–470.
[14] X. Shen, A. Hertzmann, J. Jia, S. Paris, B. Price, E. Shechtman, and I. Sachs, "Automatic portrait segmentation for image stylization," *Comput. Graph. Forum*, vol. 35, no. 2, pp. 93–102, May 2016.
[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[16] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen, "Incremental network quantization: Towards lossless CNNs with low-precision weights," 2017, *arXiv:1702.03044*. [Online]. Available: http://arxiv.org/abs/1702.03044
[17] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," 2014, *arXiv:1412.6115*. [Online]. Available: http://arxiv.org/abs/1412.6115
[18] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1389–1397.
[19] F. Tung and G. Mori, "CLIP-Q: Deep network compression learning by in-parallel pruning-quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7873–7882.
[20] R. Yu, A. Li, C.-F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, and L. S. Davis, "NISP: Pruning networks using neuron importance score propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9194–9203.
[21] K. D. Rajab, "New associative classification method based on rule pruning for classification of datasets," *IEEE Access*, vol. 7, pp. 157783–157795, 2019.
[22] R. Reed, "Pruning algorithms—A survey," *IEEE Trans. Neural Netw.*, vol. 4, no. 5, pp. 740–747, Sep. 1993.
[23] Q. Tian, T. Arbel, and J. J. Clark, "Deep LDA-pruned nets for efficient facial gender classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 10–19.
[24] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*. [Online]. Available: http://arxiv.org/abs/1608.08710
[25] J.-H. Luo, J. Wu, and W. Lin, "ThiNet: A filter level pruning method for deep neural network compression," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5058–5066.
[26] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2736–2744.
[27] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167
[28] H. Chen, F. Zhang, B. Tang, Q. Yin, and X. Sun, "Slim and efficient neural network design for resource-constrained SAR target recognition," *Remote Sens.*, vol. 10, no. 10, p. 1618, Oct. 2018.
[29] F. Zhang, Y. Liu, Y. Zhou, Q. Yin, and H.-C. Li, "A lossless lightweight CNN design for SAR target recognition," *Remote Sens. Lett.*, vol. 11, no. 5, pp. 485–494, May 2020.
[30] K. Alex, S. Ilya, and E. H. Geoffrey, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 6, 2012, pp. 1097–1105.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[32] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and & < 0.5 MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360
[33] L. Leng, M. Li, C. Kim, and X. Bi, "Dual-source discrimination power analysis for multi-instance contactless palmprint recognition," *Multimedia Tools Appl.*, vol. 76, no. 1, pp. 333–354, Jan. 2017.
[34] L. Leng and A. B. J. Teoh, "Alignment-free row-co-occurrence cancelable palmprint fuzzy vault," *Pattern Recognit.*, vol. 48, no. 7, pp. 2290–2303, Jul. 2015.
[35] L. Leng and J. Zhang, "PalmHash code vs. PalmPhasor code," *Neurocomputing*, vol. 108, pp. 1–12, May 2013.
[36] Z. Wu, Y. Huang, Y. Yu, L. Wang, and T. Tan, "Early hierarchical contexts learned by convolutional networks for image segmentation," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 1538–1543.
[37] L. Yann, B. Leon, Y. Bengio, and H. Patrick, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[38] S. Han, P. Jeff, T. John, and D. William, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.
[39] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*. [Online]. Available: http://arxiv.org/abs/1510.00149
[40] L. Shen, Z. C. Lin, and Q. M. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks. European conference on computer vision," in *Proc. ECCV*, 2016, pp. 467–482.
[41] C. Y. Lee, S. N. Xie, P. Gallagher, Z. Y. Zhang, and Z. W. Tu, "Deeply-supervised nets," *Artif. Intell. Statist.*, pp. 562–570, Feb. 2015.
[42] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015.
[43] B. Z. Zhou, H. Zhao, P. Xavier, F. Sanja, B. Adela, and T. Antonio, "Semantic understanding of scenes through the ade20k dataset," *Int. J. Comput. Vis.*, vol. 2016, vol. 127, no. 3, pp. 1–20, 2016.
[44] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[45] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: http://arxiv.org/abs/1706.05587

JUN MIAO received the Ph.D. degree from Nanchang University, Nanchang, China, in 2015. He was a Visiting Scholar with the University of California at Merced, Merced, CA, USA. He is currently a Researcher with the Key Laboratory of Jiangxi Province for Image Processing and Pattern Recognition, and an Associate Professor with the School of Aeronautical Manufacturing Engineering, Nanchang Hangkong University. His research interests include computer vision, 3-D reconstruction, and pattern recognition.

KEQIANG SUN received the bachelor's degree from West Anhui University, Lu'an, China. He is currently pursuing the master's degree with Nanchang Hangkong University. His research interests include computer vision and image processing.

XUAN LIAO received the bachelor's and master's degrees from Nanchang Hangkong University, Nanchang, China, in 2016 and 2019, respectively. His research interests include computer vision and image processing.

LU LENG (Member, IEEE) received the Ph.D. degree from Southwest Jiaotong University, Chengdu, China, in 2012.

He performed his postdoctoral research at Yonsei University, Seoul, South Korea, and the Nanjing University of Aeronautics and Astronautics, Nanjing, China, from 2012 to 2015. He was a Visiting Scholar with West Virginia University, USA, from 2015 to 2016. He is currently an Associate Professor with Nanchang Hangkong University, and also a Visiting Scholar with Yonsei University. He has published more than 70 international journal and conference papers. He has been granted several scholarships and funding projects for his academic research. He is also a reviewer for several international journals and conferences. His research interests include computer vision, biometric template protection, and biometric recognition.

Dr. Leng is a member of the Association for Computing Machinery (ACM), the China Society of Image and Graphics (CSIG), and the China Computer Federation (CCF).

JUN CHU received the Ph.D. degree from Northwestern Polytechnic University, Xi'an, China, in 2005.

She was a Postdoctoral Researcher with the Exploration Center of Lunar and Deep Space, National Astronomical Observatories, Chinese Academy of Sciences, from 2005 to 2008. She was a Visiting Scholar with the University of California at Merced, Merced, CA, USA. She was also the Director of the Jiangxi Institute of Computer Science. She is currently the Director of the Key Laboratory of Jiangxi Province for Image Processing and Pattern Recognition, the Vice President, and a Full Professor with the School of Software, Nanchang Hangkong University. Her research interests include computer vision and pattern recognition. She was also a member of the Computer Vision Special Committee and the China Computer Federation.