Received November 9, 2020, accepted November 23, 2020, date of publication November 26, 2020,
date of current version December 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3040823

Two-Stage Model Compression and Acceleration: Optimal Student Network for Better Performance

JIALIANG TANG[1], **(Graduate Student Member, IEEE), NING JIANG**[1], **(Member, IEEE), WENXIN YU**[1], **(Member, IEEE), JINJIA ZHOU**[2,3], **(Member, IEEE), AND LIUWEI MAI**[4]
[1]School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China
[2]School of Science and Engineering, Hosei University, Tokyo 184-0002, Japan
[3]JST, PRESTO, Tokyo 332-0012, Japan
[4]Sichuan Languang Development Company Ltd., Chengdu 611731, China

Corresponding author: Ning Jiang (jiangning@swust.edu.cn)

This work was supported in part by the Sichuan Science and Technology Program under Grant 2020YFS0307, JST PRESTO, Japan, under Grant JPMJPR1757, SWUST Doctoral Foundation under Grant 19zx7102.

ABSTRACT Convolutional neural networks(CNNs) have demonstrated its advanced ability in many fields. However, the calculations and parameters of the advanced CNNs are unaffordable for exiting intelligence devices. This problem mostly hinders the practical application of CNNs. In this paper, we propose a two-stage model compression and acceleration(abbreviated as STCA) method to solve this problem. The STCA is composed of supernet and subnet, the supernet is a large pre-trained neural network with superior performance, and the subnet is obtained by pruning the supernet. More specifically, the overall process of STCA includes the search and train stage. In the search stage, we first search and remove the unnecessary channels of the supernet based on channel importance pruning to get the pruned network. Then the weights in the pruned network are initialized to get the subnet. During the training stage, the subnet will learn from the training data and the supernet together. We will extract the knowledge from the supernet and transfer it to the subnet to improve the performance of the subnet. We have proved the effectiveness of STCA by implementing extensive experiments on several advanced CNNs (VGGNet, ResNet, and DenseNet). All subnet trained by STCA achieve significant performance, especially when selecting the VGGNet-19 as the supernet, the subnet only with about 1/10 parameters and 1/2 calculations achieves 94.37% and 74.76% accuracy on the CIFAR-10 and CIFAR-100 dataset, which are 0.84% and 2.31% higher than the accuracy of the supernet.

INDEX TERMS Model compression and acceleration, stagewise search and train, convolutional neural networks, network pruning, knowledge transform.

I. INTRODUCTION

In recent years, convolutional neural networks(CNNs) have achieved excellent results in many computer vision tasks, such as image classification [3], [9], [24], object detection [6], [22], and semantic segmentation [2], [19], which have greatly promoted the development of artificial intelligence. Overall, with the continuous improvement of the CNNs performance, its depth and width are also increasing. In 2012, AlexNet [13] contained only eight convolutional layers, and there are most 256 channels in a layer, achieved top-1 and top-5 error rates of 37.5% and 17% respectively on ImageNet dataset [46]. In 2016, the deeper ResNet-152 [9] with 152 layers and has most 512 channels in a layer had a performance superior to AlexNet, with top-1 and top-5 error rates 21.43% and 5.71%, respectively. However, these large-scale CNNs with superior performance often with huge memory and computational burdens. For example, when inferencing an image with a size of 224×224, the ResNet-152 has approximately 60 million parameters and requires more than 20 Giga float-point-operations(FLOPs). The great memory and computing resource consumption of CNNs are unaffordable for these resource-limited intelligent devices such as autonomous vehicles, smart speakers, and smartphones, which hinders the further development of artificial intelligence.

Many approaches have been proposed to reduce the number of parameters and calculations of CNNs to promote the practical application of artificial intelligence. These approchs mainly include network pruning [16], [20], knowledge distillation [1], [11], lightweight network [27], [29] and automatic model compression methods [4], [48].

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

Network pruning first selects and prunes unimportant parts of the CNNs by specific criteria, and then fine-tunes the pruned network on the dataset to recover the ability. Network pruning can reduce the size of neural network greatly, but the performance of the pruned neural network is unsatisfactory when the pruning rate is higher. The knowledge distillation framework mainly includes the teacher network and student network. During the training process, the knowledge extracted from the teacher network will transfer to a small student network. After knowledge transforms, the student can achieve better performance than the network only trained on the dataset. After knowledge distillation, the student network with fewer parameters and calculations can achieve the ability to meet practical applications. But knowledge distillation requires a network with a specific optimized structure as a student network, and the knowledge distillation is difficult to extend to other neural network structures. For example, when use ResNet-family [9] neural network to implement knowledge distillation, The 50-layers ResNet50 or the 34-layers ResNet34 are often select as the teacher network, the 18-layers ResNet18 is often choosing as the student network. However, the student network is usually not small enough. For example, ResNet18 still has 11 million parameters and 557 million calculations, which always has a large memory and calculation burden for smart devices. Lightweight networks are composed of efficient convolutional structures, which can achieve high performance with few parameters. However, the lightweight networks require design artificial, which is very time-consuming and requires many experiments to verify the effectiveness of the network. The automatic model compression method uses neural architecture search [43], [44] or reinforcement learning [51], [52] to automatically realize model compression. These methods may solve the problems of the aforementioned model compression methods, but the existing automated model compression method is very dependent on computing resources. Cheng *et al.* [49], Choudhary *et al.* [50] summarizes the existing model compression and acceleration methods to propose that how to effectively use the limited available resources to design specific compression methods for these resource-limited devices remains a challenge.

In this paper, we propose a two-stage model compression and acceleration framework(STCA) to solve the problems of the existing model compression methods. Our proposed method can compress the model to any size and improve the accuracy of the model to meet the needs of practical applications. Our proposed STCA includes the search stage and the training stage. Figure 2 is the overall framework of our proposed approach STCA. An optimized structure is crucial for the CNNs with fewer parameters and calculations to achieve excellent performance. We will first search for an optimized small network (called subnet) from a big network (called supernet). The input image will be input to the supernet and subnet to obtain the features, classification results, and soft-targets of the supernet and subnet. Then the consistency loss, cross-entropy loss, and soft-target loss are calculated in turn. In the end, the optimized subnet will achieve significant accuracy. In [18], Liu *et al.* proposed the main effect of network pruning is to find an optimized pruned network, and they train the pruned network from scratch to get a considerable performance with the pruned network fine-tuned on the dataset. Therefore, in the search stage, network pruning is used to find an optimized subnet from a pre-trained supernet, and we will initialize the weights of the subnet. The process of STCA searches for subnets from the supernet very fast. When training on CIFAR-10 dataset, searching for a subnet with about half calculations from the ResNet32 [9] only takes about 20 seconds on a single V100 Graphics Processing Unit(GPU). STCA is significantly faster than the automatic model compression and acceleration method TAS [4], which uses the neural architecture search to spend more time searching for the structure of an optimal network. When ResNet32 is also used for training on the CIFAR-10 dataset, TAS finish the searching procedure of ResNet-32 in about 3.8 hours on a single V100 GPU and reduce about 50% calculations of ResNet32.

In the stage of training, we will fix the weights of the supernet, and the supernet is only used to implement knowledge transform. The training data will be input into the supernet and the subnet to get the outputs of the supernet and subnet, respectively. In the first, the outputs of supernet and subnet will input to the softmax function to get the soft-targets. By minimizing the difference of the soft-target of the supernet and subnet, the knowledge will be able to transfer from the supernet to the subnet to improve the performance of the subnet. Then, the output of the subnet is compared to the ground-truth label, and the performance of the subnet is future improved. Extensive experiments on the benchmark dataset demonstrate that the subnet with fewer parameters and calculations searched and trained by our proposed STCA can achieve higher accuracy than the supernet. Figure 1 shows the

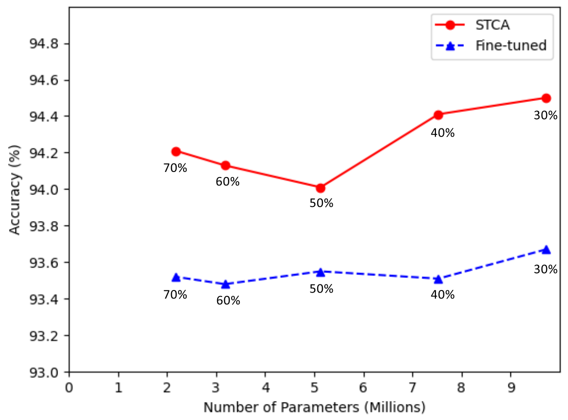


FIGURE 1. All experiments are performed on the CIFAR-10 dataset. The circular dots represent the accuracy of the subnet trained by STCA under different pruning rates. The triangle points represent the accuracy of fine-tuning by the pruning network under different pruning rates. In the same model size, the performance of the subnet trained by STCA is superior to the performance of fine-tuning the pruned network on the dataset.

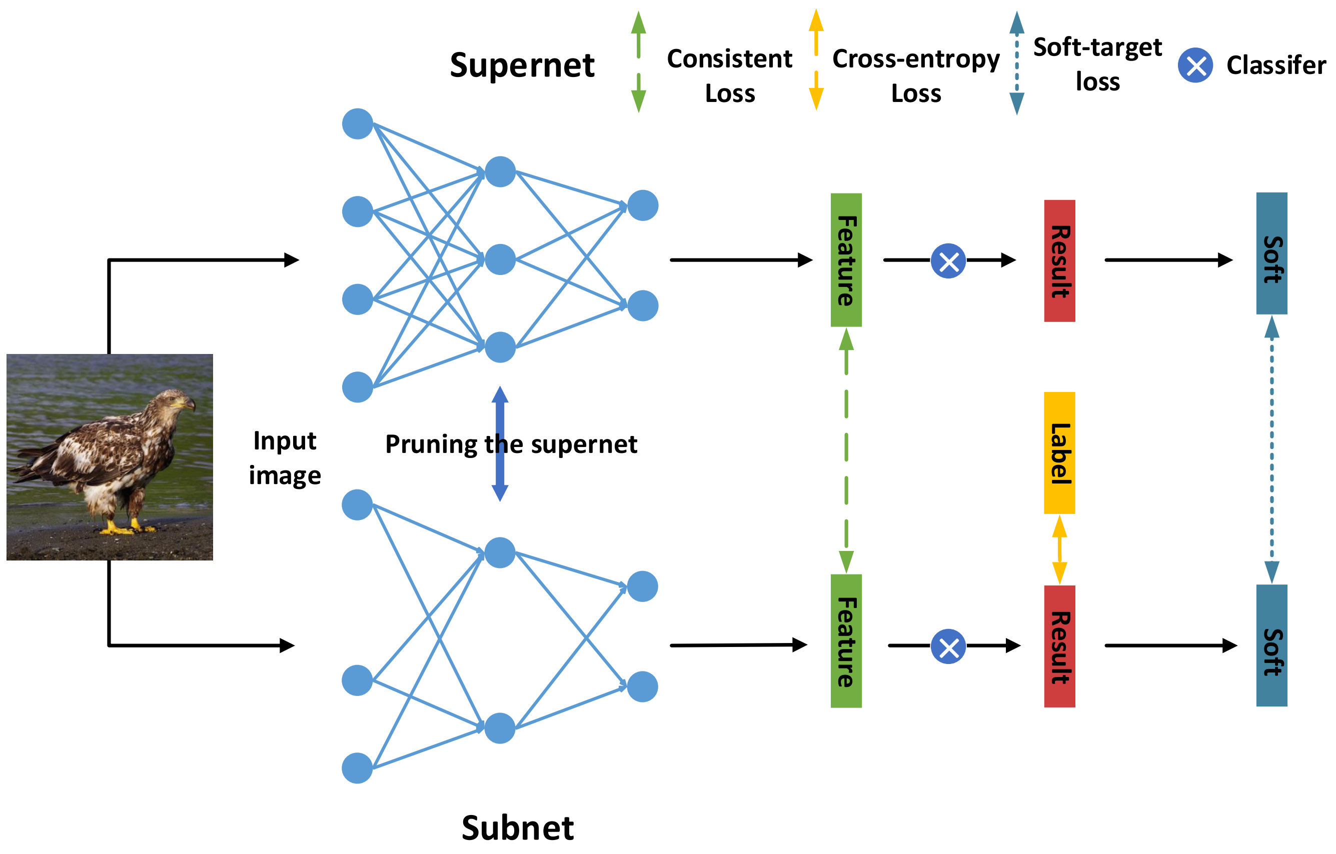


FIGURE 2. **The entire structure of our proposed method. The subnet is obtained by pruning the supernet. The image will input into the supernet and subnet to get the features, classification results, and soft-targets. Then, the subnet will learn under the supervision of the truth labels and the knowledge from the supernet.**

accuracy of the subnet trained by STCA and fine-tune on the data, the accuracy of the subnet trained by STCA all higher than the accuracy fine-tunes on the data at various prune ratio.

Our contributions are summarized as follows:

1. We propose a two-stage model compression and acceleration framework composed of supernet and subnet, abbreviated as STCA. STCA can obtain an optimal subnet by pruning the supernet, and the subnet trained under the supervision of the supernet. After training, the subnet trained by STCA achieves state-of-the-art performance.

2. Compared with the supernet, the subnet trained by STCA has only a few parameters and calculations, but the subnet has achieved better performance than the supernet.

3. We conducted a large number of experiments on a range of benchmark datasets(CIFAR-10, CIFAR-100, STL-10), and extensively verified the effectiveness of STCA.

The rest of the paper is summarized as follows: Section II describes the related works of the paper. Section III elaborates the details about how to implement the STCA. Section IV shows the experiment results of STCA on benchmark datasets. Section V concludes the paper. Section V is the supplementary materials of the paper.

II. RELATED WORK

Based on different algorithms and applications, existing model compression and acceleration methods main include three categories, i.e., network pruning, knowledge distillation, and lightweight network. There are also some methods are proposed to search for an optimal structure automatic.

A. NETWORK PRUNING

In the search stage of the STCA, we will select and pruned the unimportant part of the supernet to get the subnet based on network pruning. The network pruning remove the connections and corresponding parts of a neural network in a structured or unstructured manner according to specific criteria and preserve the network's performance as more as possible. In the early work of network pruning, optimal brain surgeon [8] and optimal brain damage [14] measure the importance of weights in the network based on the second derivative of the loss function and then crop the unimportant weights to get a compact network. Han *et al.* [7] proposed to find important connections by training the entire network and prune the unimportant connections to obtains a sparse network. Finally, the remaining parameters will fine-tune on data to recover the capacity of the pruned network. Peng *et al.* [20] proposed calculating the channel importance evaluation based on Taylor expansion to find and remove unimportant channels in a neural network. Network slimming [17] adds a scale factor to each channel to control the network's output and regularizes these scale factors by

$L1$-*norm* during the training process. After training, the channels with low scale factor will be pruned. Li *et al.* [15] proposed a structured network pruning framework, which presets the number of filters to keep in each layer and uses the sum of absolute weights of the filters to measure this filter's effect. The filters in each layer will sort accord to the sum of absolute weights, and finally, these filters will be pruned according to the preset threshold. Lottery [5] search a relatively optimized sparse network (called winning tickets) in the complex deep neural network. The parameter amount and complexity of the winning tickets are much lower, but it achieves an inference accuracy similar to the original network. Long *et al.* [18] proposed that the main contribution of neural network pruning is to obtain an optimized network structure, and to initialize the weights of the pruned network to train the network from scratch, finally achieves the same accuracy as the fine-tuning of the pruned network that retains the weights on the dataset. Dong *et al.* [4] proposed the automatic network pruning method based on neural architecture search, and similar to us, the original network is used to transfer knowledge to the pruned network. But their search phase is time-consuming, and the accuracy of the pruned network is also reduced to some extent.

B. KNOWLEDGE DISTILLATION

In the training stage of STCA, we transfer knowledge from the supernet to the subnet effective based Knowledge distillation. Knowledge distillation utilizes a large network with more parameters and calculations to transfer knowledge to a small network with fewer parameters and calculations to improving the performance of the small network. Ba and Caruana [1] proposed that a shallow net with fewer parameters can achieve similar results with a deep net, and uses a large network to transfer knowledge to a small network. The concept of knowledge distillation was first proposed by Hinton *et al.* [11]. First, they input the outputs of the teacher network and student network to the softmax function to get the soft-target, which contains more information than the outputs. Then, the student network learns from the soft-target and realize knowledge transfer. FitNets [23] extracts the middle layer features of the teacher network to obtain more knowledge for knowledge transfer and successfully mimics a deeper student network. Zagoruyko and Komodakis [25] proposed that only use the feature map of the neural network to transfer knowledge from teacher network to student network is inefficient and uses the attention mechanism to extract the attention map from the feature map to implement knowledge transfer more efficient. Yang *et al.* [24] proposed to add the noise to the teacher network. It makes the teacher network contains more soften information and reduces the performance of the teacher network to a certain extent. But the soften knowledge from the teacher network improves the generalized ability of the student network. Knowledge distillation based on generative adversarial network(KDGAN) [32] introduce to train the student network to improve its performance by adversarial learning loss greatly. During the training, the conditional generative adversarial networks is used to discriminate the outputs of the teacher network and the student network. Furlanello *et al.* [33] proposed the Born Again Neural Networks(BornNet), which train a small network in generations under the supervision of a large network. After multiple generations of training, the small network finally obtains performance beyond the large network. Correlation congruence knowledge distillation(CCKD) [21] introduced the correlation congruence as the knowledge, in the progress of knowledge transfer, the instance-level information and the correlation information between instances are transferred together. Positive-Unlabeled knowledge distillation (PUKD) [35] uses a positive-unable classifier to select training data from the cloud based on a small amount of data and realizes knowledge distillation with only a few labeled data. Date-Free learning(DFL) [34] combines generative adversarial networks and knowledge distillation to construct a data-free knowledge distillation structure, which can train high-performance student networks without data. Wang *et al.* [47] proposed a one-shot automatic pruning method based on the online ensemble distillation, which pruning removes the redundant structures of CNNs at once in a global way to obtain compact ones without any iterative pruning and retraining.

C. LIGHTWEIGHT NETWORK

Different from our proposed STCA to search a subnet with an optimized structure, the lightweight network designs an optimized structure with small memory and computing requirements artificially. Squeezenet [27] uses the 1×1 convolution layer to replace the 3×3 convolution layer to compose the fire module. By stacking fire modules, it achieves AlexNet-level accuracy with about 50x fewer parameters and 0.5MB model size. Chollet *et al.* [28] proposes to separate the channels correlation and the spatial correlation to get the separable convolution. Then, they use the separable convolution to improve the Inception to get the Xception, which obtains higher accuracy when the number of parameters is less than the Inception network. MobileNet [29] uses depth-wise separable convolution to build a lightweight neural network and introduces width multiplier and resolution multiplier to balance the accuracy and the inference speed. Zhang *et al.* [30] proposed to use group convolution and channel shuffle to construct a shuffle unit, and to construct ShuffleNet by stacking shuffle units. Chen *et al.* [31] proposed that in the CNNs, the multiplication operation has higher computational complexity than the simple addition operation. By using addition operation to replace the multiplication operation in CNNs, they hugely reduce the size of the CNNs.

D. NEURAL ARCHITECTURE SEARCHING

An optimized structure is essential for the neural network to achieve superior performance, but the artificially designed state-of-the-art structure [40]–[42] is very complicated and time-consuming. There are also some researchers proposed the neural architecture search(NAS) methods to search for

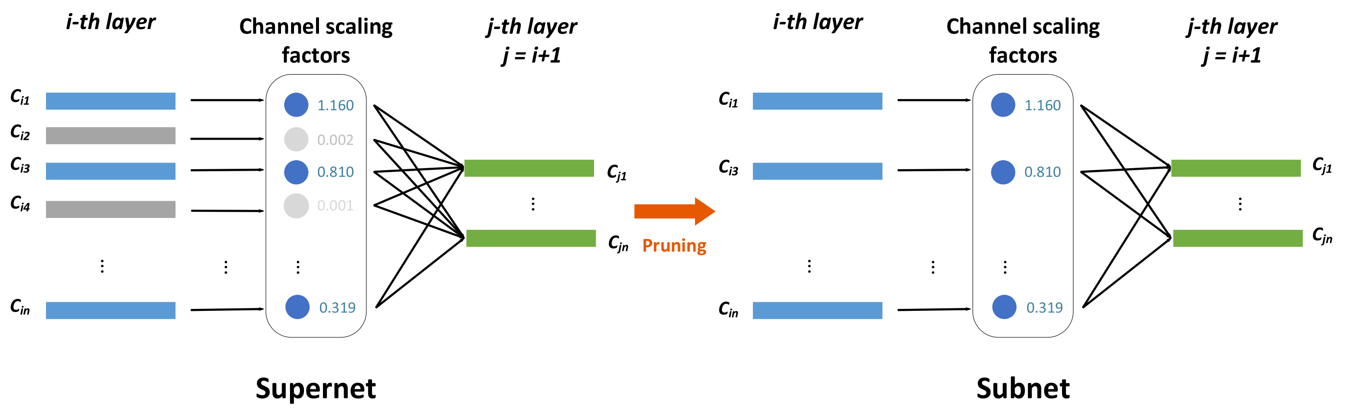


FIGURE 3. The diagram of how to pruning the neural network. After pruning, the channel with a lower scaling factor in the supernet will be pruned to get the subnet.

an optimized network structure through automatic learning search. Different from with the STCA use the network pruning to search the neural architecture. Zoph *et al.* [43] proposed to automatically generate a CNN structure based on transfer learning and promote the searched model to achieve SOTA effects on large-scale image classification and object detection. ENAS [44] effective reduce the time-consuming computing process of NAS, which shortens the GPU computing time of NAS by more than 1000 times. Liu *et al.* [45] proposed a differentiable architecture search(DARTS), DARTS uses gradient descent to efficiently search neural network architectures based on the continuous relaxation of the architecture representation.

III. APPROACH

In this section, we will explain how to implement our proposed method STCA and analysis why STCA is effective. The purpose of our proposed method is first to find an optimized subnet with a few parameters and calculations. Then, we will transfer knowledge from the supernet to the subnet to promote the subnet to achieve the performance beyond the supernet.

A. SEARCH THE OPTIMAL SUBNET

The existing advanced neural network can obtain satisfactory performance after specific iterations of training. However, in a neural network, each neuron contributes differently to the final classification result of the network. There are many inefficient neurons in the neural network, which leads to a large amount of waste of computing and memory resources. So, it is essential to search for an optimal structure for the neural network to use a small number of parameters to obtain superior performance. The fundamental purpose of network pruning is to find and remove the part that contributes less to the final result of a CNNs. After pruning, the pruned network has fewer parameters and calculations while retaining as much accuracy as possible. Recent research [18] shows that the network pruning can search for an optimal structure for a small network with superior performance. Therefore, in this paper, we use the network pruning to search for a suitable network structure and take the network slimming [16] as an example to elaborate on how to search the optimal subnet.

The figure 3 shows the process of how to pruning a layer of a neural network. More specifically, we will add a trainable scale factor γ to each channel of the neural network. During the training process, we will train the weights of the network and these scale factors of channels jointly. After training, each scale factor of the channel will represent their final contribution to the network. These channels with small scale factors mean their contribution lower to the neural network, and these channels can be removed almost without down the ability of the neural network. But adding a scale factor to each channel will increase the parameters of the network. In the search stage, we use batch normalization(BN) [36] layer to act as a scale layer. BN is a basic unit of CNNs and is widely applied in existing neural networks. The transformation progress can be written as follows:

$$\hat{x} = \frac{x_{in} - \mu}{\sqrt{\sigma^2 + \epsilon}}; \quad x_{out} = \gamma \hat{x} + \beta \tag{1}$$

The x_{in} and x_{out} are the input and output of BN, the $\hat{x}$ is the temp features. $\mu = \frac{1}{m}\sum_{i=1}^{m} x_{in}^{i}$ is the mean of x_{in} and $\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_{in}^{i} - \mu\right)^2$ is the variance of the x_{in}, m is the number of mini-batch, ϵ is a minimum to prevent division by zero errors. The γ and β are the adaptive learning parameters, which are used to improve the representability of the network. The γ is range from 0 to 1 and multiplied with the $\hat{x}$, which is equivalent to the scale factor of the $\hat{x}$. So, the γ in BN is used as the scale factor without any more additional parameters, and the $L1$-*norm* is applied to promote the γ of the inefficient channel to zero. After training, these channels in each layer will sort by the γ, and the pruning threshold will be calculated according to the preset ratio. Then we will prune the channels with a γ small than the threshold. For example, when using the 19-layers VGGNet19 [23] as a supernet, we will prune 70% of the channels and initialize the weight of the remaining channels to get a supernet.

Algorithm 1 Training Process of STCA

```
Input: A set of RGB images $x = \{x^1, x^2, \cdots, x^n\}$, the preset ratio $r$.
Model: The pre-trained supernet $\mathcal{N}_p$
Parameter: Hyper-parameter $\alpha$ $\beta$ $\gamma$ of the overall objective function.
1:  Part One: Searching the optimal subnet:
2:  Sort the channels of supernet by the corresponding scale factor;
3:  Get the sort list $S = \{s^1, s^2, \cdots, s^N\}$, $N$ is the total channel number of the supernet;
4:  Get the threshold $Thre$ by sorting list $S$ and preset pruning rate $r$;
5:  Set $i$=0;
6:  repeat
7:      Compare the scale factor $s^i$ with the $Thre$;
8:      if $s^i < Thre$;
9:      Pruned the corresponding channel of $s^i$;
10:     $i = i + 1$;
11: until $i < N$
12: Get the pruned network $\mathcal{N}_b$;
13: initialized the weights of $\mathcal{N}_p$ to get the subnet $\mathcal{N}_b$;
14: Part Two: Training the subnet:
15: repeat
16:     Input the images $x$ to the supernet $\mathcal{N}_p$ and subnet $\mathcal{N}_b$;
17:     Get the feature $F$, classification results $Y$, soft-target $S$ of the $\mathcal{N}_p$ and $\mathcal{N}_b$;
18:     Calculation the loss for consistency $L_{cs}$ by $F$;
19:     Calculation the loss for soft-target $L_{st}$ by $S$;
20:     Calculation the loss for classification $L_{label}$ by $Y$;
21:     Update weights in the subnet $\mathcal{N}_b$ through back-propagation;
22: until convergence
    Output: The compact subnet $\mathcal{N}_b$.
```

The table 1 shows the configuration of the subnet obtained by pruning the VGGNet19 according to various pruned ratios.

Long *et al.* [18] proposed that the pruned network can be initialized and trained from scratch to achieve the performance similar to the retained weights pruned network, which fine-tuned in the dataset. In our STCA, we hope that the subnet can learn from the dataset and the supernet together. The subnet retains the original weight will be able to prevent the subnet from learning knowledge from the supernet. So, in our method, the weights of the subnet will be initialized, and the subnet will be trained from scratch.

B. TRAIN THE SUBNET

Unlike the convention training process of network pruning, which first pruned the CNNs and then fine-tuned the pruned network on the data. In the paper, the subnet will first be initialized; then, the subnet will training under the supervision of the supernet and data. Figure 2 shows the train process of the subnet. The input image x is first input into

TABLE 1. ConvNet configurations. The leftmost is the network configuration of the supernet, and the right is the network configuration of the subnet obtained by pruning the 30%, 50%, and 70% channels of the supernet. The conv3 × 3-64 represents that the neural network layer is composed of a 64-channel convolution kernel with a size of 3 × 3. The conv3 × 3-64 represents that all the channel in the layer are pruned. Each convolutional network block consists of several convolutional layers followed by a maxpool layer. At the end of the convolutional layer, it contains a 512-channel fully connected layer(FC-512) and a soft-max layer.

ConvNet Configuration			
Supernet(VGGNet19)	Subnet		
Pruned Ratio	30%	50%	70%
input(32×32 RGB image)			
conv3×3-64 conv3×3-64	conv3×3-62 conv3×3-64	conv3×3-61 conv3×3-64	conv3×3-53 conv3×3-64
maxpool			
conv3×3-128 conv3×3-128	conv3×3-128 conv3×3-128	conv3×3-128 conv3×3-128	conv3×3-128 conv3×3-128
maxpool			
conv3×3-256 conv3×3-256 conv3×3-256 conv3×3-256	conv3×3-256 conv3×3-256 conv3×3-256 conv3×3-256	conv3×3-256 conv3×3-256 conv3×3-256 conv3×3-256	conv3×3-256 conv3×3-256 conv3×3-255 conv3×3-210
maxpool			
conv3×3-512 conv3×3-512 conv3×3-512 conv3×3-512	conv3×3-510 conv3×3-473 conv3×3-356 conv3×3-191	conv3×3-482 conv3×3-221 conv3×3-62 conv3×3-31	conv3×3-34 conv3×3-2 conv3×3-0 conv3×3-0
maxpool			
conv3×3-512 conv3×3-512 conv3×3-512 conv3×3-512	conv3×3-109 conv3×3-82 conv3×3-214 conv3×3-511	conv3×3-4 conv3×3-3 conv3×3-44 conv3×3-499	conv3×3-0 conv3×3-0 conv3×3-1 conv3x3-264
maxpool			
FC-512			
soft-max			

the supernet and subnet to obtain the feature of the supernet $\boldsymbol{F_p} = \{f_p{}^1, f_p{}^2, \cdots, f_p{}^n\}$ and the feature of subnet $\boldsymbol{F_b} = \{f_b{}^1, f_b{}^2, \cdots, f_b{}^n\}$, n is the number of the feature. Because the supernet is a pre-trained neural network with excellent performance, we assume that if the feature of the subnet is more consistent with those of the supernet, then the subnet also has a similar performance. Euclidean distance is a distance metric that can measure the absolute distance between two points in a multidimensional space. In the training of subnet, the Euclidean distance is used to calculate the consistency loss of their feature. By minimizing the Euclidean distance, the feature of subnet will improve to more consistency with the supernet. The consistency loss function is formula as:

$$\mathcal{L}_{cs} = \frac{1}{n}\sum_{i=1}^{n}\left\|F_b{}^i - F_p{}^i\right\|_2 \tag{2}$$

Then, the feature of the supernet and subnet will input into the classification layer to obtain the classification result of the supernet $\boldsymbol{Y_p} = \{y_p{}^1, y_p{}^2, \cdots, y_p{}^n\}$ and the classification result of the subnet $\boldsymbol{Y_b} = \{y_b{}^1, y_b{}^2, \cdots, y_b{}^n\}$. Specifically, the classification result of CNNs is the one-hot vector, such as $o = [0, 1, 0 \ldots 0]$. There is only one value is one and others are zero. The one-hot vector o only contains the information of the class that the corresponding value is 1. However, there is more similar information between pictures in the dataset. For example, a dataset contains pictures of cats, dogs, and cars. The pictures of cats contain a lot of information similar to dogs and contain little information similar to cars. This similar information is very useful for improving CNNs. Hinton *et al.* [11] proposed to input the classification result to the softmax function to get the soft-target example as $s = [0.1, 0.8, \ldots, 0.05]$ to transfer knowledge. There are many non-zero values in the soft-target, which means the soft-target s contains more similar information than the one-hot vector o. So, in our proposed approach, the soft-target is used to transfer knowledge from the supernet to the subnet. In details, for the classification result $\boldsymbol{Y} = \{y^1, y^2, \cdots, y^n\}$ of neural network, the soft-target can be get as follows:

$$s^i = \frac{\exp\left(y^i/T\right)}{\sum_j \exp\left(y^j/T\right)} \tag{3}$$

The T is the temperature to soften the output neural network. A more significant T can get a more softened soft-target, more soften soft-target contains more non-zero values. By utilizing the eq.3, the soft-target of supernet and subnet can be get as $\boldsymbol{S_p} = \{s_p{}^1, s_p{}^2, \cdots, s_p{}^n\}$ and $\boldsymbol{S_b} = \{s_b{}^1, s_b{}^2, \cdots, s_b{}^n\}$ respectively. The loss function of soft-target is elaborated as:

$$\mathcal{L}_{st} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{H}_{KLdiv}\left(\mathrm{s}_p^i, \mathrm{s}_b^i\right) \cdot T^2 \tag{4}$$

The $\mathcal{H}_{KLdiv}$ is the KL divergence, if the S_p and S_b more similar, the KL divergence more smaller.

In the train process of CNNs, the network will be trained under the supervision of the ground-truth label $\boldsymbol{L} = \{l^1, l^2, \cdots, l^n\}$. We will supervise the subnet learning by the ground-truth label to promote the subnet classification more accurately. The classification result of the subnet $\boldsymbol{y_b} = \{y_b{}^1, y_b{}^2, \cdots, y_b{}^n\}$ will calculate the cross-entropy loss with the ground-truth label through the following formula:

$$\mathcal{L}_{\text{label}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{H}_{\text{cross}}\left(\mathrm{y}_b^i, l^i\right) \tag{5}$$

The $\mathcal{H}_{cross}$ is the cross-entropy loss function. By minimizing the cross entropy loss of the subnet, the classification of the subnet can be promoted more accurately.

C. OVERALL OBJECTIVE FUNCTION

By combining these loss functions proposed above, our final objective function is:

$$\mathcal{L}_{\text{Total}} = \alpha \cdot \mathcal{L}_{cs} + \beta \cdot \mathcal{L}_{st} + \gamma \cdot \mathcal{L}_{label} \tag{6}$$

The α, β and γ are the hyperparameter to adjust the three term in the overall objective function. Through the final loss function, the subnet can learn more information from the supernet and can absorb the information not exiting in the supernet from the dataset. In the end, the subnet with few parameters and calculations will achieve a very significant performance than the supernet. The algorithm list of our STCA is shows at algorithm.1.

TABLE 2. ConvNet configurations. The configuration of the Predefined Structured Pruning, the firsy layer of supernet and the last three convolution blocks will pruned half number of channels.

Convnet Configuration	
Predefined Structured Pruning	
Supernet(VGGNet16)	Subnet
Model type	VGGNet16-H
input(32×32 RGB image)	
conv3×3-64 conv3×3-64	conv3×3-32 conv3×3-64
maxpool	
conv3×3-128 conv3×3-128	conv3×3-128 conv3×3-128
maxpool	
conv3×3-256 conv3×3-256 conv3×3-256	conv3×3-128 conv3×3-128 conv3×3-128
maxpool	
conv3×3-512 conv3×3-512 conv3×3-512	conv3×3-256 conv3×3-256 conv3×3-256
maxpool	
conv3×3-512 conv3×3-512 conv3×3-512	conv3×3-256 conv3×3-256 conv3×3-256
maxpool	
FC-512	
soft-max	

D. ANALYZE OUR PROPOSED METHOD

The memory and computational costume of the subnet are less than that of the supernet, but the subnet trained by our proposed method have achieved better performance than the supernet. There are mainly two reasons why the subnet achieves such performance:

1. The subnet has a more optimized network structure: A more optimized structure means that this network contains fewer inefficient neurons. In the process of back-propagation, without the interference of these invalid neurons, the subnet can more easily find the optimal point, thereby obtaining higher performance.

2. The subnet can learn more effective knowledge: The Subnet can gain knowledge from the supernet and dataset together. At one aspect, the supernet has a larger network structure than the subnet, which means that the supernet can learn more information from the dataset and includes more useful knowledge. Therefore, in knowledge transfer, the subnet can learn a lot of valid information that cannot be learned from the dataset. On the other hand, the structure of the subnet and the supernet is different, which means the knowledge that

the subnet can learn from the dataset is also different from the supernet. It leads the subnet contains a lot of useful information that the supernet does not. After training, the subnet with effective knowledge from the supernet and the dataset so that the subnet can achieve a better performance than the supernet.

IV. EXPERIMENTS

In this section, we will chose three pruning algorithms to search the network model for extensive experiments. There are non-structural pruning methods [7], predefined structured pruning methods [15] and automatic structured pruning method [16]. We will use various popular image classification CNNs as our experimental model, include the VGGNet-family [23] CNNs, ResNet-family [9] CNNs, and DenseNet-family [12] CNNs. In all experiments, the parameters in the total objective function eq.6 are set to: α=0.1, β=0.9 and γ=0.1.

We use the CIFAR-10 [37], CIFAR-100 and STL-10 [38] datasets for training. For the CIFAR datasets, the training set has 50,000 pictures, and the test set has 10,000 pictures. The CIFAR datasets consist of 32×32 pixel images in multiple categories, CIFAR-10 contains 10 categories, and CIFAR-100 contains 100 categories. The STL-10 dataset is an image recognition dataset similar to CIFAR-10, which contains 10 categories of pictures. Compared with CIFAR-10, the STL-10 [38] dataset is more challenging. Each category contains only 500 tagged training images, and STL-10 consists of higher resolution (96×96) images.

A. UNSTRUCTURED PRUNING

The unstructured pruning method named weight-level [7] which turns the dense network into a sparse network. It first trains the entire network to find out the crucial connections based magnitude. Then the low-weight connections will be removed base on the predefined threshold. Finally, the remaining parameters will fine-tune on the dataset to boost the performance of pruned network. The details of weight-level can be seen at appendixV-A. In this experiment, we only prune the weights of the convolutional layer, without changing the structure of the fully connected layer. We first use the weight-level to get a pruned network and initialize the pruned network to get the subnet by the same initialization method as [10]. The CIFAR and STL-10 datasets are used as the training data. When the experiment on the CIFAR datasets are used as data, the details of the training sets are set as follows, the batch size is 64, the parameter T in eq.3 is set as 8. For the optimization function of the neural network, we select the Stochastic Gradient Descent(SGD) [26] as the optimizer, the learning rate of the optimizer is 0.1 and divide by 10 every 80 times epoch. In SGD, weight-decay is used to adjust the influence of model complexity on the loss function to prevent the overfitting of the neural network model during the training process, in the experiments, we set weight-decay as 5e-4. The momentum is a commonly used acceleration technology in SGD, we set the momentum as 0.9 during these experiments. When the STL-10 is used as data, the learning rate and weight-decay of SGD are all set as 0.01, others training settings are same as the CIFAR datasets. To prove the effectiveness of our proposed method more extensively, the 16-layers VGGNet-16 [23], 110-layers pre-activation PreResNet-110 [9] and 40-layers DenseNet-40 [12] are used as the supernet. And these CNNs will training on the CIFAR and STL-10 datasets.

Table 3 is the results of the subnet which uses the unstructured magnitude-based weight pruning to search a network architecture. Each supernet will be cut by 30% channel to get a subnet. When fine-tuning the pruned network on the data, due to the reduction in the number of parameters, when the subnet is fine-tuned on the CIFAR-100 dataset, its effect is still lower than that of the supernet. When the VGGNet-19 is used as a supernet, the accuracy of the subnet the fine-tuning on the CIFAR-100 dataset reduced by 0.67% to 71.96%. But when the subnets are trained by the STCA, which enable the subnets learning from the knowledge of the supernet, all the subnets have achieved better performance than the supernet. When training on the CIFAR-10 dataset, the accuracy of the subnets obtained by pruning VGGNet-19, PreResNet, and DenseNet-40 is increased by 0.55%, 0.51%, and 0.21% respectively compared with the supernet. When training on the CIFAR-100 dataset, all the subnets still achieve satisfactory performance. Especially when VGGNet-19 is used as a supernet to train the corresponding subnet, STCA has achieved the greatest improvement, the subnet has achieved 73.58% accuracy, which is 0.95% higher than the supernet. When training the subnet on the STL-10 dataset, the VGGNet-19 is used as the supernet, the subnet trained by STCA still get the best accuracy of 86.04%. This experiment results show that our proposed STCA can effectively improve the performance of small networks under the supervision of supernet.

TABLE 3. Results for using weight-level to search the subnet. The PR represent the preset prune ratio.

Dataset	Model	Original	PR	Fine-tuned	ST
CIFAR-10	VGGNet-19	93.50%	30%	93.51%	**94.05%**
	PreResNet-110	95.04%	30%	95.06%	**95.55%**
	DenseNet-40	94.40%	30%	93.86%	**94.61%**
CIFAR-100	VGGNet-19	72.63%	30%	71.96%	**73.58%**
	PreResNet-110	76.96%	30%	76.88%	**77.31%**
	DenseNet-40	73.82%	30%	73.65%	**74.64%**
STL-10	VGGNet-19	85.41%	30%	85.27%	**86.04%**

B. PREDEFINED STRUCTURED PRUNING

The predefined structured pruning method [15] pre-sets the structure of the pruned network and uses the $L1$-*norm* to find the unimportant parts of the network to cut them out. The training settings is same as section IV-A. The 16-layers VGGNet-16, 56-layers ResNet-56 and 110-layers ResNet-110 are used as supernets. We use $L1$-*norm* to find half channels of each convolutional layer in the supernet to remove. These channels contributed lower to the classification result of the supernet, so that can be pruned without hurt

TABLE 4. Results for using *l1-norm* to search subnet.

Dataset	Model	Original	Pruned Model	Fine-tuned	ST
CIFAR-10	VGGNet-16	93.63%	VGGNet-16-H	93.41%	**94.21%**
	ResNet-56	93.14%	ResNet-56-H	92.97%	**94.19%**
	ResNet-110	93.14%	PreResNet-110-H	93.14%	**94.75%**
CIFAR-100	VGGNet-16	72.41%	VGGNet-16-H	72.37%	**74.61%**
	ResNet-56	72.20%	ResNet-56-H	72.14%	**73.07%**
	ResNet-110	73.89%	PreResNet-110-H	72.87%	**75.58%**
STL-10	VGGNet-16	84.93%	VGGNet-16-H	84.64%	**86.90%**

TABLE 5. Results for using network slimming to search subnet.

Dataset	Model	Original	Prune Ratio	Fine-tuned	Accuracy
CIFAR-10	VGGNet-19	93.53%	70%	93.60%	**94.37%**
	PreResNet-164	95.04%	40%	94.77%	**95.69%**
	DenseNet-40	94.10%	60%	94.00%	**94.33%**
CIFAR-100	VGGNet-19	72.63%	70%	72.32%	**74.76%**
	PreResNet-164	76.80%	40%	76.22%	**77.24%**
	DenseNet-40	73.82%	60%	73.35%	**73.98%**
STL-10	VGGNet-19	85.41%	30%	85.36%	**86.21%**
	VGGNet-19	85.41%	50%	85.25%	**85.69%**
	VGGNet-19	85.41%	70%	85.13%	85.24%

the ability of the supernet and get the optimal subnet. The pruning procee of predefined structured pruning can be seen at appendixV-B.

Table 4 displays the experiment results of the subnets pruned by the predefined structured pruning method. The table shows the accuracy of the supernet, the precision achieved by the subnet through fine-tuning, and the accuracy achieved by the subnet trained by STCA. It can be seen from the table that thees subnets trained by our STCA all achieved the best results when using various CNNs as supernets. It is worth noting that when only about half of the parameters and calculations are included, the subnets trained by STCA all achieve significantly higher performance than the supernet. When training on the CIFAR-10 dataset, the accuracy of the subnets obtained by pruning VGGNet-16, ResNet-56, and ResNet-110 are increased by 0.58%, 1.22%, and 1.61% respectively compared with the supernet. When training on the CIFAR-100 dataset, the accuracy of the subnets obtained by pruning VGGNet-16, ResNet-56, and ResNet-110 are increased by 2.20%, 0.93%, and 1.69% respectively compared with the supernet. The subnet performance of STCA training on the CIFAR-100 dataset is more superior. When training on the STL-10 dataset, the subnet trained by STCA still achieved the best performance, about 86.90%, which is 1.97% higher than the 84.93% accuracy of the supernet. CIFAR-100 and STL-10 datasets contain fewer pictures in each category than the CIFAR-10 dataset. It is more difficult for a model trained on the CIFAR-100 and STL-10 datasets to achieve better performance. This proves that the subnet trained by STCA can learn more effective information from the dataset and supernet.

C. AUTOMATIC STRUCTURED PRUNING

The automatic structured pruning Network Sliming [16] can performs structured pruning automatically by adding scale factors to each channel. Section III-A describes the specific implementation steps. The experiment sets are the same as section IV-A. When using the 19-layers VGGNet-19 as supernet, we will prune 70% channels of the VGGNet-19 to get the subnet. When taking the 164-layers pre-activation PreResNet-164 as supernet, there are 40% parameters be pruned. And when the 40-layers DenseNet is set as supernet, we will cut off 60% channels of it.

Table 5 shows the experimental results of taking automatic structure pruning to search the subnet. The accuracy of these small subnets trained by STCA is significantly higher than that of training subnets by fine-tuning. Even compared with the original large supernet, the subnet trained by our method still has more superior performance. When the CIFAR-10 is used as dataset, the accuracy of the subnets obtained by pruning VGGNet-19, PreResNet-164, and DenseNet-40 are increased by 0.84%, 0.65%, and 0.23% respectively compared with the supernet. When training on the CIFAR-100 dataset, the accuracy of the subnets is increased by 2.13%, 0.44%, and 0.16% respectively compared with the supernet. The effect of STCA on

TABLE 6. Results for using various pruned ration to search subnet. The FLOPs mean floating-point operations per second, which is used to represent the amount of calculation. The pruned ratio is 0% means the network unpruned and without the accuracy of fine-tuned and ST.

Dataset	Model	Original	Pruned Ratio	Parameter	FLOPs	Fine-tuned	ST
CIFAR-10	VGGNet-19	93.53%	0%	20.8M	7.97×10^8	-	-
			50%	5.01M	5.01×10^8	93.55%	**94.01%**
			60%	3.70M	3.98×10^8	93.48%	**94.13%**
			70%	2.31M	3.91×10^8	93.52%	**94.37%**
CIFAR-100	VGGNet-19	72.63%	0%	20.8M	7.97×10^8	-	-
			50%	5.01M	5.01×10^8	72.37%	**74.30%**
			60%	3.70M	3.98×10^8	72.21%	**74.58%**
			70%	2.31M	3.91×10^8	72.17%	**74.76%**

TABLE 7. Results for the subnet trained by ST compare to other method. Scratch represents the accuracy of the pruned network train from scratch.

Dataset	Model	Original	Pruned Ratio	Scratch [17]	LECNP [16]	LCCL [39]	KD [11]	STCA
CIFAR-10	VGGNet-19	93.53%	70%	93.30%	93.52%	93.31%	91.94%	**94.37%**
	PreResNet-164	95.04%	40%	94.70%	94.68%	94.09%	94.06%	**95.69%**
CIFAR-100	VGGNet-19	72.63%	70%	71.86%	72.17%	72.05%	69.21%	**74.76%**
	PreResNet-164	76.80%	40%	76.36%	76.10%	75.26%	72.15%	**77.24%**

the CIFAR-100 dataset is more obvious than that of the CIFAR-10 dataset, which is consistent with the experimental results in Section IV-B. When performing a experiment on the STL-10 dataset, the subnets obtained by STCA training the VGGNet-19 achieved 86.21%, 85.69%, and 85.24% accuracy when the pruning rate was 30%, 50%, and 70%, respectively. Both are higher than the accuracy of the subnet fine-tuned on the dataset. And when the pruning rate is 30% and 50%, the accuracy of the subnet trained by STCA is still higher than that of the supernet. When the pruning rate is 70%, since the subnet has only a few parameters and calculations, its performance is slightly lower than that of the supernet, about 0.17%. These experimental results show that our proposed STCA can automatically search and train an optimal network, and achieve better performance than the supernet with only fewer memory and computational burden.

D. VARIOUS PRUNED RATIO TO SEARCH SUBNET

To prove the effectiveness of our proposed STCA more extensively, in this section, we will employ various pruned ratios for experiments. The 19-layers VGGNet-19 is selected as supernet, and the automatic structured pruning Network Slimming is used to search the subnet. The pruned ratio are set as 30%, 50% and 70%. The training settings are same as setcionIII-A, and the CIFAR datasets are used as training data.

Table 6 displays the experiment results. The subnet trained by our STCA achieved better results than the subnet fine-tuning on the dataset when using all pruning ratios, and also achieved better performance than the supernet. When training on the CIFAR-10 dataset, the subnets trained by STCA obtained the accuracy of 94.01%, 94.13%, and 94.37% respectively at 50%, 60%, and 70% pruning rates. When using the CIFAR-100 dataset, the subnets trained by STCA achieved the precision of 74.30%, 74.58%, and 74.76% respectively at 50%, 60%, and 70% pruning rates. Especially when the pruning rate is 70%, the subnet trained by STCA only has about 1/10 parameters and 1/2 calculations of the supernet, achieve the significant accuracy on the CIFAR-10 and CIFAR-100 datasets, which are significantly higher than the accuracy of supernet, about 0.84% and 2.13%.

E. COMPARISON WITH SOTA METHODS

In this section, the subnet trained by STCA will be compared with other SOTA methods. Include subnet trained from scratch [17], the subnet trained by the LCCL [39] and LECNP [16]. The subnet trained by knowledge distillation [11]. When the subnet trained from scratch, the pruned will be initialized and without the supernet to transfer knowledge. When using knowledge distillation for training, the subnet is obtained by directly reducing the number of channels in the supernet according to the corresponding pruning ratio. For example, when the pruning ratio is set as 50%, there are half of channels in each layer will be pruned. We chose the 19-layers VGGNet-19, the 164-layers pre-activation PreResNet-164, and the 40-layers DenseNet-40 as supernets.

The experimental results is exhibits in table 7. The KD means knowledge distillation. When using different structures as supernets, the corresponding subnets trained by our STCA all obtain the highest accuracy. Although the subnets trained by knowledge distillation will also learn knowledge from the supernet, the effect of these subnets is not ideal. It proves that an optimized structure is crucial for

TABLE 8. Results for the subnet trained by ST which uses various term in $\mathcal{L}_{\text{Total}}$.

Dataset	Model	Prune Ratio	Original	$\mathcal{L}_{\text{cs}}$	$\mathcal{L}_{\text{cs}}$+$\mathcal{L}_{\text{st}}$	$\mathcal{L}_{\text{cs}}$+$\mathcal{L}_{\text{kd}}$	$\mathcal{L}_{\text{Total}}$
CIFAR-10	VGGNet-19	30%	93.53%	93.01%	93.90%	94.20%	**94.44%**
CIFAR-100	VGGNet-19	30%	72.63%	71.88%	74.81%	75.09%	**75.13%**

a CNNs to achieve better ability. And the performance of the subnet trained from scratch and the subnet purned by LCCL and LECNP fine-tuned on the data is lower than that of the subnet trained by STCA, it is because these CNNs trained from scratch and fine-tune can only learn knowledge from the dataset. It proves that the knowledge from the supernet is very useful for improving the performance of the subnet. The experimental results prove that the search and train stage of STCA jointly promote the subnet to achieve better results.

F. ABLATION EXPERIMENTS

In this section, we will demonstrate the importance of each loss function in $\mathcal{L}_{\text{Total}}$ through ablation experiments. We will select the automatic structure pruning Network Sliming [16] to search the optimal structure, and the subnet will trian on CIFAR dataset, the prune ratio is set as 30%, the trian settings are same as sectionIV-A.

Tabel 8 shows the experiment results of ablation experiments. The subent will trainied uses $\mathcal{L}_{\text{cs}}$, $\mathcal{L}_{\text{cs}}+\mathcal{L}_{\text{st}}$, $\mathcal{L}_{\text{cs}}+\mathcal{L}_{\text{kd}}$, $\mathcal{L}_{\text{Total}}$ respectively. When using $\mathcal{L}_{\text{cs}}$ to train the subnet, the subnet only trained on the dataset, and the accuracy of the subnet is lower than the supernet due to 30% parameters are pruned. When $\mathcal{L}_{\text{st}}$ and $\mathcal{L}_{\text{kd}}$ are used for training, the performance of the subnet has been greatly improved. Compare to the subnet only uses $\mathcal{L}_{\text{cs}}$ for training, on the CIFAR-10 and CIFAR-100 dataset, the subnet trained by $\mathcal{L}_{\text{cs}}+\mathcal{L}_{\text{st}}$ achieves 93.93% and 74.81% respectively, the subnet trained by $\mathcal{L}_{\text{cs}}+\mathcal{L}_{\text{kd}}$ achieves 94.20% and 75.09% respectively. The subnet trianied by $\mathcal{L}_{\text{Total}}$ achieves the best accuracy, 94.44% on CIFAR-10 and 75.13% on CIFAR-100 respectively. This experiment results demonstrated that each term in our proposed overall objective function is effective to improve the performance of the subnet.

V. CONCLUSION

The huge memory and computational burden of Convolutional Neural Networks(CNNs) are unaffordable for exiting resource-limited artificial intelligence(AI) devices. In this paper, we propose a two-stage model compression and acceleration approach STCA to deal with the problem. The structure of STCA consists of the large supernet and the small subnet which pruned by supernet. At the search stage, STCA searches the subnet with optimized structure from the supernet based on the network pruning method. At the training stage, the subnet is trained from scratch, and the knowledge is transformed from supernet to the subnet to promote the performance of the subnet.

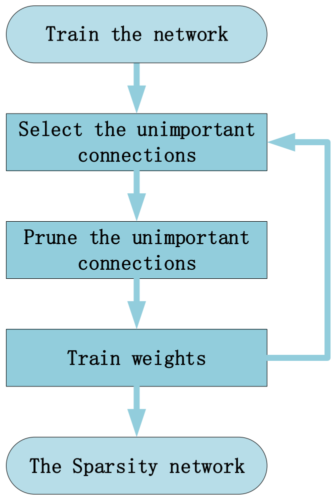

Train the network
Select the unimportant connections
Prune the unimportant connections
Train weights
The Sparsity network

FIGURE 4. The diagram of train step of the unstructured network pruning method.

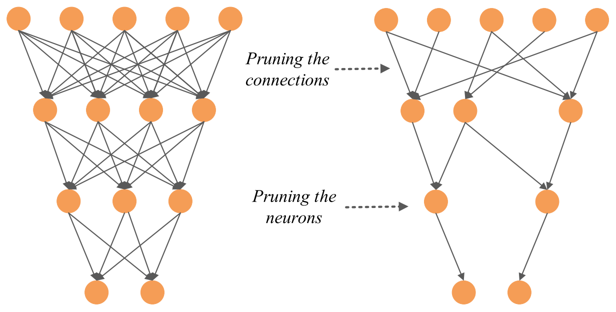


FIGURE 5. The diagram of train step of the unstructured network pruning method.

Extensive experiments on benchmark dataset prove that our proposed method can be widely applied to existing advanced CNNs to compress the size of the network and improving the performance of the network. When the VGGNet19, ResNet50, DenseNet40, etc. are used as the supernet, the corresponding subnet only with few parameters and calculations all achieve significantly better performance than the supernet. In future work, we will further improve STCA according to practical application requirements and train the suitable compact model to apply to AI equipment.

APPENDIX

A. UNSTRUCTURED PRUNING

Figure 4 shows the entire pruning step of the unstructured network pruning method(USNP). USNP first trains a neural network to find unimportant connections. Unlike conventional training of the neural networks, the purpose of

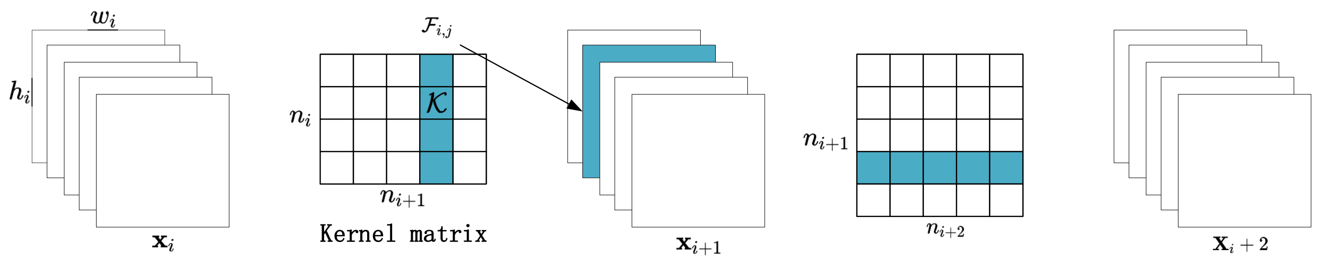


FIGURE 6. The x_i is the input feature, the x_{i+1} is the output feature, and the x_{i+2} is the output feature of the next layer. The $\mathcal{F}_{i,j}$ is a 3-dimension convolution kernel and $\mathcal{K}$ is the 2-dimension convolution kernel.

the USNP training neural network is to determine which connections are important, rather than training to get the final value of these connections. After the neural network is trained, the USNP will sort these connections according to the corresponding weight values, and then obtain the pruning threshold through the preset pruning rate. By comparing the pruning threshold with the weights of these connections, connections with weights lower than the threshold will be pruned. Finally, the network that has been pruned is retrained to make the network retain as much performance as possible. These three steps of the USNP will repeat until the neural network convergence.

1) REGULARIZATION

During the network training process, the regularization method used will affect network pruning and retraining. $L1$ regularization can effectively penalize smaller non-zero parameters and cause more parameters to approach zero, enabling the network to find unimportant connections more accurately and obtain better accuracy. But for other connections with larger parameters, the effect of $L1$ regularization is not as good as that of $L2$ regularization. In general, $L2$ regularization can get better pruning results.

2) ITERATIVE PRUNING

To learn the correct connection and keep the accuracy of the pruned network as much as possible, USNP will iteratively perform network pruning. In the entire pruning process of the neural network, the pruning will be repeated-in the prune and retrain step, and the number of connections in the neural network will decrease gradually. After many rounds of such iterations, the smallest number of connections can be found.

3) PRUNING NEURONS

After trimming the connections, in order to further reduce the size of the network, the USNP will prune these neurons with zero input connections or zero output connections, as shown in figure 5. Neurons with zero-input connections (or zero-output connections) have no contribution to the final performance of the neural network, so cutting out these neurons will not affect the performance of the network.

Algorithm 2 Searching Process of PSNP

```
1: Calculate the sum of its absolute kernel weights $s_j = \sum_{l=1}^{n_i} \sum |\mathcal{K}_l|$;
2: Get the sort list $S = \{s_1, s_2, \cdots, s_{n_i}\}$, $_i$ is the channel number of layer $i$;
3: Select filters with the smallest sum values, and pruned the filters and corresponding feature maps. In the next convolutional layer, the m corresponding feature maps are also removed;
```

B. PREDEFINED STRUCTURED PRUNING

It is different from the magnitude-based pruning method that can only reduce the parameters of the fully connected layer. Predefined structured pruning(PSNP) is an accelerated pruning method. PSNP will prune those convolution kernels that are considered to have little effect on the output accuracy. In the pruning process, PSNP deletes the entire convolution kernel and its feature maps to reduce computational cost. Unlike the method of pruning weights, PSNP does not cause sparse connection patterns.

As shown in figure 6, PSNP will prune the entire convolution kernel when performing network pruning. The h_i and w_i is the height and width of the input feature $x_i \in \mathbb{R}^{n_i \times h_i \times w_i}$. And the x_i will input to in 3D convolution kernel $\mathcal{F}_{i,j} \in \mathbb{R}^{n_i \times k \times k}$ to get the output feature $x_{i+1} \in \mathbb{R}^{n_{i+1} \times h_{i+1} \times w_{i+1}}$, then, the x_{i+1} will input to next layer to get the x_{i+2}. The kernel matrix is composed of n_{i+1} number of $\mathcal{F}_{i,j}$, and the $\mathcal{F}_{i,j}$ is composed of n_i number of 2D kernels $\mathcal{K} \in \mathbb{R}^{k \times k}$. During the pruning process, PSNP will search the ineffective $\mathcal{F}_{i,j}$ to prune, after pruned the $\mathcal{F}_{i,j}$, the corresponding feature map in x_{i+1} is removed, which reduces $n_i k^2 h_{i+1} w_{i+1}$ operations. After pruning, if there were m filters of layer i is pruned, it will reduces m/n_i computations for layer i and $i+1$.

1) SEARCH THE UNIMPORTANT KERNEL

PSNR uses $l1 - norm$ to determine the importance of the convolution kernel. First, PSNR calculate the absolute values of the parameters in the 3D convolution kernel $\mathcal{F}_{i,j}$ and add them. Then prune the smallest or smallest m 3D Convolution kernel. The detailed steps are shown in the algorithm 2.

REFERENCES

[1] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2654–2662.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014, *arXiv:1412.7062*. [Online]. Available: http://arxiv.org/abs/1412.7062

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Dec. 1998.

[4] X. Dong and Y. Yang, "Network pruning via transformable architecture search," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 760–771.

[5] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," 2018, *arXiv:1803.03635*. [Online]. Available: http://arxiv.org/abs/1803.03635

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[7] S. Han, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. systems.*, vol. 2015, pp. 1135–1143.

[8] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Proc. Adv. Neural Inf. Process. Syst.*, 1993, pp. 164–171.

[9] 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.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[11] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*. [Online]. Available: http://arxiv.org/abs/1503.02531

[12] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[14] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 598–605.

[15] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. Peter Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*. [Online]. Available: http://arxiv.org/abs/1608.08710

[16] 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.

[17] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," 2018, *arXiv:1810.05270*. [Online]. Available: http://arxiv.org/abs/1810.05270

[18] 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.

[19] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," 2016, *arXiv:1611.06440*. [Online]. Available: http://arxiv.org/abs/1611.06440

[20] B. Peng, X. Jin, D. Li, S. Zhou, Y. Wu, J. Liu, Z. Zhang, and Y. Liu, "Correlation congruence for knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5007–5016.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[22] A. Romero, N. Ballas, S. Ebrahimi Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*. [Online]. Available: http://arxiv.org/abs/1412.6550

[23] 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

[24] C. Yang, L. Xie, S. Qiao, and A. Yuille, "Knowledge distillation in generations: More tolerant teachers educate better students," 2018, *arXiv:1805.05551*. [Online]. Available: http://arxiv.org/abs/1805.05551

[25] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," 2016, *arXiv:1612.03928*. [Online]. Available: https://arxiv.org/abs/1612.03928

[26] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 421–436.

[27] 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

[28] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[29] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[30] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[31] H. Chen, Y. Wang, C. Xu, B. Shi, C. Xu, Q. Tian, and C. Xu, "AdderNet: Do we really need multiplications in deep learning?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1468–1477.

[32] Z. Xu, Y.-C. Hsu, and J. Huang, "Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks," 2017, *arXiv:1709.00513*. [Online]. Available: http://arxiv.org/abs/1709.00513

[33] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar, "Born again neural networks," 2018, *arXiv:1805.04770*. [Online]. Available: http://arxiv.org/abs/1805.04770

[34] H. Chen, Y. Wang, C. Xu, Z. Yang, C. Liu, B. Shi, C. Xu, C. Xu, and Q. Tian, "Data-free learning of student networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3514–3522.

[35] Y. Xu, "Positive-unlabeled compression on the cloud," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 2565–2574.

[36] 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

[37] *Learning Multiple Layers of Features from Tiny Images*, Alex Krizhevsky, Toronto, ON, Canada, 2009.

[38] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 215–223.

[39] X. Dong, J. Huang, Y. Yang, and S. Yan, "More is less: A more complicated network with less inference complexity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5840–5848.

[40] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[42] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," 2019, *arXiv:1905.11946*. [Online]. Available: http://arxiv.org/abs/1905.11946

[43] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2016, *arXiv:1611.01578*. [Online]. Available: http://arxiv.org/abs/1611.01578

[44] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," 2018, *arXiv:1802.03268*. [Online]. Available: http://arxiv.org/abs/1802.03268

[45] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," 2018, *arXiv:1806.09055*. [Online]. Available: http://arxiv.org/abs/1806.09055

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[47] Z. Wang, S. Lin, J. Xie, and Y. Lin, "Pruning blocks for CNN compression and acceleration via online ensemble distillation," *IEEE Access*, vol. 7, pp. 175703–175716, 2019, doi: 10.1109/ACCESS.2019.2957203.

[48] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han, "AMC: AutoML for model compression and acceleration on mobile devices," 2018, *arXiv:1802.03494*. [Online]. Available: http://arxiv.org/abs/1802.03494

[49] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 126–136, Jan. 2018, doi: 10.1109/MSP.2017.2765695.

[50] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artif. Intell. Rev.*, vol. 53, no. 7, pp. 5113–5155, Oct. 2020, doi: 10.1007/s10462-020-09816-7.

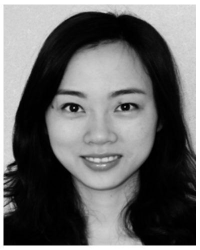
[51] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[52] K. Clark and C. D. Manning, "Deep reinforcement learning for mention-ranking coreference models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1–4, doi: 10.18653/v1/D16-1245.

JIALIANG TANG (Graduate Student Member, IEEE) was born in Mianyang, Sichuan, China, in 1997. He received the B.S. degree from Chengdu Technological University, Chengdu, China, in 2019. He is currently pursuing the M.S. degree with the Southwest University of Science and Technology, Mianyang. His research interests include image recognition, multimodal information transformation and fusion, neural network model compression and acceleration, computer vision, and deep learning.

NING JIANG (Member, IEEE) received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, in 2006, and the M.S. and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2010 and 2013, respectively. He is currently an Associate Professor and a Ph.D. Supervisor with the School of Computer Science and Technology, Southwest University of Science and Technology. His main research interests include feature extraction algorithm-based on target detection and recognition, network structure optimization algorithm-based on artificial deep neural networks, fast feature matching and accurate target positioning algorithm-based on binocular vision, and model compression algorithm-based on deep neural networks.

WENXIN YU (Member, IEEE) was born in Mianyang, Sichuan, China, in 1984. He received the B.S. degree from Shanghai Jiao Tong University, Shanghai, China, in 2006, and the M.S. and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2010 and 2013, respectively. From 2015 to 2017, he was an Associate Research Fellow with the School of Computer Science and Technology, Southwest University of Science and Technology. Since 2018, he has been the Vice President of the School of Computer Science and Technology, Southwest University of Science and Technology. He has been exploring the cutting-edge direction of video and image processing and focused on 3-D image synthesis, image stereo matching, and other issues. His main research interests include 3-D multiview synthesis filling technology, image stereo matching technology, multiview compatible fast coding algorithm, neural networks, pattern recognition, low-power consumption video decoding algorithm, and image error concealment technology.



JINJIA ZHOU (Member, IEEE) received the B.E. degree from Shanghai Jiao Tong University, China, in 2007, and the M.E. and Ph.D. degrees from Waseda University, Fukuoka, Japan, in 2010 and 2013, respectively. From 2013 to 2016, she was a Junior Researcher with Waseda University. She was selected as a JST PRESTO Researcher for the period of 2017–2021. She is currently an Associate Professor and a Co-Director of the English-Based Graduate Program, Hosei University. She is also a Senior Visiting Scholar with the State Key Laboratory of ASIC and System, Fudan University, China. Her research interests include algorithms and VLSI architectures for multimedia signal processing. She received the Research Fellowship of the Japan Society for the Promotion of Science, from 2010 to 2013, and the Hibikino Best Thesis Award, in 2011. She was a recipient of the Chinese Government Award for Outstanding Students Abroad of 2012. She was a co-recipient of the ISSCC 2016 Takuo Sugano Award for Outstanding Far-East Paper, the Best Student Paper Award of VLSI Circuits Symposium 2010, and the Design Contest Award of ACM ISLPED 2010. She participated in the design of the world first 8K UHDTV video decoder chip, which was granted the 2012 Semiconductor of the Year Award of Japan.

LIUWEI MAI was born in Guangzhou, Guangdong, in 1985. He received the bachelor's degree in electronic information technology and technology from Zhaoqing University, in June 2008. He currently works with the Shanghai Branch, Sichuan Languang Development Company Ltd., the company. He has more than ten years of work experience related to artificial intelligence. He participated in and presided over a number of artificial intelligence projects. Since 2008, he has been engaged in the research of artificial intelligence equipment.