Cognitive Computation (2021) 13:1070–1084
https://doi.org/10.1007/s12559-021-09894-x

Multi-task Pruning via Filter Index Sharing: A Many-Objective Optimization Approach

Hanjing Cheng[1] · Zidong Wang[2] · Lifeng Ma[3] · Xiaohui Liu[2] · Zhihui Wei[1]

Received: 5 April 2021 / Accepted: 7 June 2021 / Published online: 25 June 2021

Abstract
State-of-the-art deep neural network plays an increasingly important role in artificial intelligence, while the huge number of parameters in networks brings high memory cost and computational complexity. To solve this problem, filter pruning is widely used for neural network compression and acceleration. However, existing algorithms focus mainly on pruning single model, and few results are available to multi-task pruning that is capable of pruning multi-model and promoting the learning performance. By utilizing the filter sharing technique, this paper aimed to establish a multi-task pruning framework for simultaneously pruning and merging filters in multi-task networks. An optimization problem of selecting the important filters is solved by developing a many-objective optimization algorithm where three criteria are adopted as objectives for the many-objective optimization problem. With the purpose of keeping the network structure, an index matrix is introduced to regulate the information sharing during multi-task training. The proposed multi-task pruning algorithm is quite flexible that can be performed with either adaptive or pre-specified pruning rates. Extensive experiments are performed to verify the applicability and superiority of the proposed method on both single-task and multi-task pruning.

Keywords Multi-task learning · Evolutionary algorithm · Network pruning · Many-objective optimization

Introduction

With the success on applications in ImageNet challenge, the deep neural networks (DNNs) have been extensively utilized in a wide variety of applications [1–3] and showed superiorities over other approaches. However, as the structure becomes deeper and larger, the number of network parameters would increase considerably, resulting in a dramatic escalation in computing cost during the utilization of DNNs. Therefore, DNN with relatively low computing cost yet high accuracy is urgently needed nowadays, which gives rise to the development of network pruning technique to simplify the structure and reduce the parameters. Pruning deep CNNs is an important direction for accelerating the network. Recent developments on pruning can be generally divided into two categories, namely weight pruning [4–6] and filter pruning [7–10, 60–62].

It is the objective of model pruning to compress the model size and accelerate the inference of the network. Weight-pruning methods remove connections in the filters or across different layers, thereby reducing the cost of memory cost. However, the main weakness of weight pruning is the unstructural operation manner. The unstructured sparsity of the filters makes weight pruning hard to deploy existing basic linear algebra subprograms (BLASs) libraries. Hence, weight pruning is not effective in reducing computational cost. In contrast, filter pruning allows models to be structured with sparsity, reducing the storage usage on devices and achieving practical acceleration.

It should be noted that most of the existing results regarding the network pruning mainly concentrates on single network structure, and few effort has been devoted to investigating the multi-task pruning due primarily to the difficulty brought by the cross-coupling of different tasks as well as increasing parameters. Moreover, different from traditional multi-task learning algorithms, the methods proposed in this

✉ Zidong Wang
Zidong.Wang@brunel.ac.uk

[1] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

[2] Department of Computer Science, Brunel University London, Uxbridge, Middlesex UB8 3PH, UK

[3] School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China

paper of sharing information for different tasks should be designed with a trade-off between network size and performance. In addition, keeping the balance between different tasks is also a challenge for multi-task pruning.

Very recently, it has been declared in [11] that the key of multi-task pruning is to reduce both intra-redundancy in single task and inter-redundancy between multiple tasks. However, the proposed algorithm cannot be applied directly for pruning because of depending on the results of network merging. Motivated by this, we intend to propose a *unified* Multi-task Filter Index Sharing (MFIS) framework which could systematically deal with both single-task and multi-task pruning. As illustrated in Fig. 1, we first establish a filter candidate box which contains all filters in both tasks. Different from traditional methods for single-task pruning, within our proposed framework, important filters are selected under certain criteria imposed on multi-task, which is converted into a many-objective optimization problem (MOP) [12]. Then, filter sharing (FS) strategy is developed to force each filter choosing one and only one important candidate to support. Filters that support the same candidate might be pruned, merged or remained according to FS strategy. Moreover, an index matrix is introduced by the pruning and merging results of multi-networks to keep the network structure and to regulate the information sharing in subsequent multi-task learning.

The main contributions of this paper can be highlighted as follows:

i) Different from most traditional pruning techniques only applicable for single task, we propose a more general framework MFIS, which is able to deal with both multi-task pruning and single-network pruning.
ii) The proposed FIS in different networks is capable of selecting important filters and determining the most suitable operation (pruning, merging, or remaining) for each filter. Meanwhile, the index matrix in FIS can keep the network structure in multi-task training.
iii) Three criteria for selecting targeted important filters are put forward to further appropriately characterize the performance of MFIS that is subsequently optimized via the MOP method.
iv) The proposed algorithm is quite flexible where two pruning rate strategies, namely adaptive rates for different tasks and fixed rate for individual task, are adopted to balance the learning speed and accuracy.
v) Via extensive experiments, the effectiveness and efficiency of MFIS are demonstrated on different networks and databases. We prove that MFIS performs better compared with both single-task pruning methods and multi-task pruning methods.

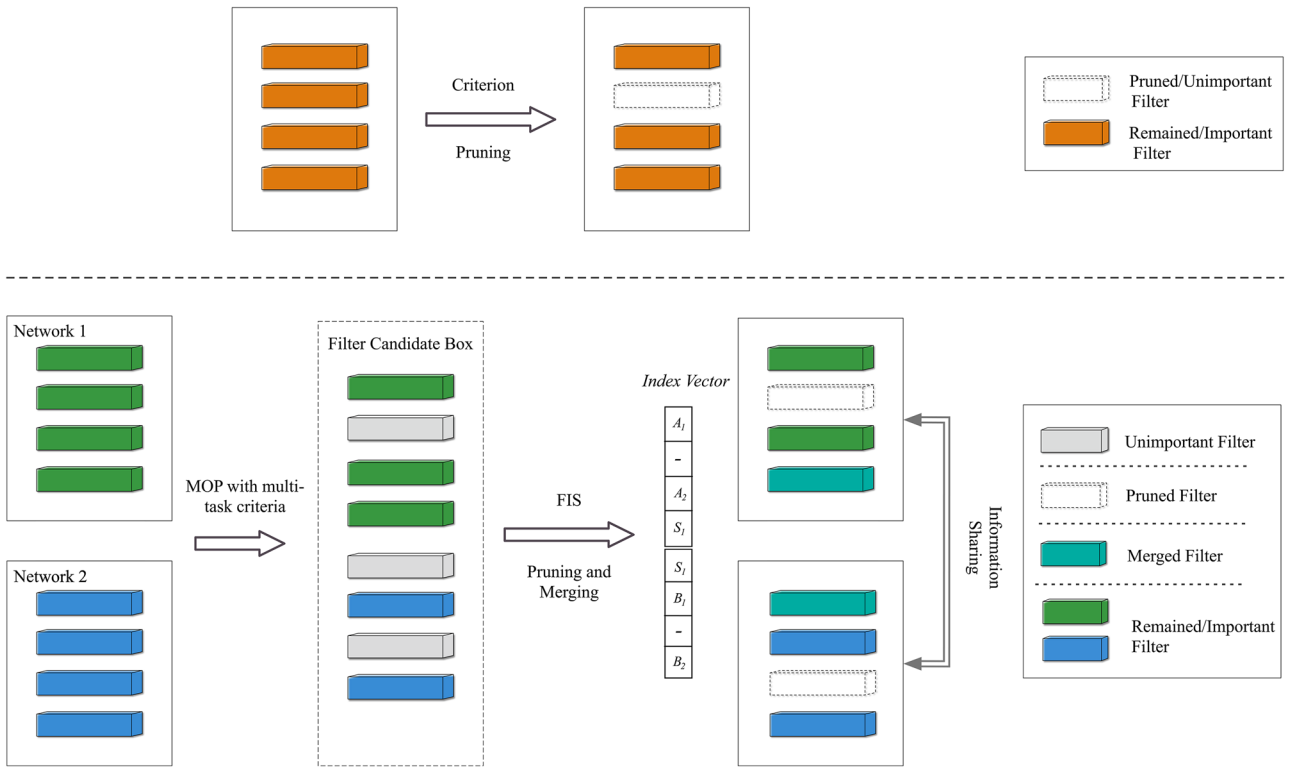

Fig. 1 An illustration of the difference between the traditional single-task pruning method and our multi-task pruning method

The remainder of this paper is organized as follows. In Section 2, we introduce the related work of model pruning, multi-task learning and many-objective optimization problem. In Section 3, we describe our framework in detail. The experimental study and the corresponding analysis are presented in Section 4. Section 5 gives the conclusions of this paper.

Related Work

Model Pruning

Filter pruning removes the entire filters, which could reduce the memory cost, and meanwhile, shows effectiveness of promotion on the inference speed. In recent publications, some works (e.g., [13, 14]) have utilized the training data and filter activations to determine the pruned filters, while other methods (e.g., [15, 16]) have determined the importance of the filters by weights. Commonly, data-independent algorithms are more effective than the former because the utilizing of training data is always computation consuming. Among data-independent algorithms, researchers design criterions to prune unimportant filters. In most existing publications regarding the model compression task, only single model compression has been considered which means that most criterions are designed for single mission. Although there have been some initial results on corresponding multi-task case, the developed algorithms have certain limitations. For instance, in [17], the proposed MTZ framework has only been able to deal with compression without pruning. In [11], an RDNet has been first established by two pre-trained models with a threshold parameter α, but the technique of pruning, i.e., Variational Information Bottleneck (VIB), is borrowed from [18].

Multi-task Learning

As is well known, multi-task Learning aims to improve generalization performance and reduces the risk of overfitting by jointly learning multiple related tasks [19–22]. ML of DNNs has been applied nowadays in various applications, from scene understanding [23] to face detection [1], to name but a few. ML enables combined model to generalize better for all tasks by sharing representations from related original networks. Recently, many researchers have devoted efforts on issues of 'what to share' and 'how to share' among tasks, especially those relevant to deep neural networks. Some representative works can be summarized as follows. Yang et al. [24] have proposed an algorithm of learning cross-task sharing structure at each layer in a deep network. Cross-stitching networks developed in [25] have introduced cross-stitch units to learn shared representations. In [26], soft ordering approach of shared layers has been applied in multi-task learning which shows more effective sharing ability. Lu et al. has proposed an automatic approach in [27] for designing multi-task deep learning architectures, which starts with a thin multi-layer network that is dynamically widened during training. All the mentioned multi-task learning methods do not consider the network size and only focus on tasks highly related. The parameters even grow during training process, leading to opposite outcome for pruning. Hence, a new multi-task pruning algorithm should be designed for combining model pruning and multi-task learning.

Many-objective Optimization Problem

On another reach frontier, MOP has been attracting particular attention in machine learning due to the fact that many relevant optimization problems can be converted to MOPs [12]. The description of an MOP problem is illustrated by the following (1) with m objectives:

$$\begin{aligned} &\min_{\boldsymbol{v}} \mathcal{F}(\boldsymbol{v}) \\ &\quad \mathcal{F} = \{\mathcal{F}_1(\boldsymbol{v}), \mathcal{F}_2(\boldsymbol{v}), \ldots, \mathcal{F}_m(\boldsymbol{v})\} \end{aligned} \tag{1}$$

where $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$ is a set of n decision variables; $\mathcal{F}(\cdot)$ is the objective function.

In our proposed work, by borrowing the idea from many-objective optimization problem, we first convert the issues of selecting the important filters into an MOP and then solve the problem by evolutionary algorithm, which will be discussed later in detail.

Methodology

Criteria for Selection

Taking two tasks as example, filter set V_A and V_B contain the original filters in model A and B, respectively. Filter *candidate* set C is initialize with V. We should first obtain an important filter set C^E from C. For the purpose of converting the selection of important filters C^E into an MOP, we introduce a binary-valued hyperparameter $\alpha_i \in \{0, 1\}$ for each filter $f_i \in C$. When $\alpha_i = 1$, it is said that f_i belongs to C^E. Then, $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ is the set of decision variables. The objectives of MOP can be designed according to the following criteria.

Loss of Replacing

This criterion aims to assure that each filter in V has appropriately small δE with a candidate in C^E:

$$\forall f_i \in V : \&\exists f_j \in C^E, \quad \text{s.t. } \delta E(i,j) < \epsilon \tag{2}$$

where $\delta E(i,j)$ denotes the loss of replacing f_i with f_j. Here, we calculate the loss of replacing $f_i \in V$ with $f_j \in C$ on task A by

$$\delta \hat{E}^A(i,j) = \begin{cases} +\infty & \alpha_j = 0 \wedge f_i \in V_A \\ \delta E^A(i,j) & \alpha_j = 1 \wedge f_i \in V_A \\ 0 & f_i \in V_B \end{cases} \tag{3}$$

Similarly, the loss of replacing f_i with f_j on task B is defined by

$$\delta \hat{E}^B(i,j) = \begin{cases} +\infty & \alpha_j = 0 \wedge f_i \in V_B \\ \delta E^B(i,j) & \alpha_j = 1 \wedge f_i \in V_B \\ 0 & f_i \in V_A \end{cases} \tag{4}$$

Then, the corresponding objective functions are defined as follows:

$$\mathcal{F}_1(\boldsymbol{\alpha}) = \sum_{f_i \in V} \min_{f_j \in C} \{\delta \hat{E}^A(i,j)\} \tag{5}$$

$$\mathcal{F}_2(\boldsymbol{\alpha}) = \sum_{f_i \in V} \min_{f_j \in C} \{\delta \hat{E}^B(i,j)\} \tag{6}$$

$\mathcal{F}_1(\cdot)$ and $\mathcal{F}_2(\cdot)$ describe the losses of replacing for network A and B, respectively. With limit on size of C^E, $\mathcal{F}_1(\cdot)$ and $\mathcal{F}_2(\cdot)$ will get rid of inappropriate candidates from C^E, thereby reducing the intra-redundancy in tasks and inter-redundancy between tasks at the minimum loss.

The Diversity of Selected Candidates

This criterion is to keep the diversity of selected candidates in C^E. To this end, we define the diversity of C^E as the sum of losses by replacing $f_i \in C^E$ with all the other filters in C^E, which is represented as follows:

$$\mathcal{D} = \sum_{i}^{\alpha_i=1} \sum_{j}^{\alpha_j=1} \delta E(i,j) \tag{7}$$

Then, the corresponding objective function $\mathcal{F}_3(\cdot)$ is defined by

$$\mathcal{F}_3(\boldsymbol{\alpha}) = -\mathcal{D} \tag{8}$$

The Entropy of Weights

A filter without variation may be failed on capturing the important information from input data [28]. Here, entropy of layer is introduced to evaluate the variation of filters' weights, which can be calculated as follows:

$$\mathcal{I} = -\sum_{k=1}^{n} p_k \log p_k \tag{9}$$

where n is the number of classes of sampling weights and p_k is the probability of classes k. High entropy of filters means high variation on weights. Then, the corresponding objective function $\mathcal{F}_4(\cdot)$ is defined by

$$\mathcal{F}_4(\boldsymbol{\alpha}) = -\mathcal{I} \tag{10}$$

Among aforementioned objectives, $\mathcal{F}_1(\cdot)$ and $\mathcal{F}_2(\cdot)$ control the intra-redundancy in tasks and inter-redundancy between tasks; $\mathcal{F}_3(\cdot)$ ensures diversity and searches candidates which are more similar to the clustering centres, close to supporters but irreplaceable with each other; Criterion $\mathcal{F}_4(\cdot)$ pays more attention to the contributions of candidates for preserving useful information. Each criterion conflicts with others and is chosen for different purpose. It is worth noting that though pruning with small L_2 norm is a mainly used criterion for filter pruning [15, 16], 'smaller-norm-less-important' has been reconsidered in some literature [29]. For multi-task pruning, searching filters with similar information that can be shared (in multi-network) or replaced (in single-network) are more important for designing criteria.

Filter Sharing

As illustrated in Fig. 2, all filters in V should choose one and only one candidate in the important filter set C^E. If $f_{j^*} \in C^E$ satisfies

$$j^* = \arg\min_j \{\delta E(i,j)\}. \tag{11}$$

we call f_i a *supporter* of candidate f_{j^*}.

The key point of the proposed filter sharing (FS) strategy is to collect filters which share the same candidate, with the same shape of colours in Fig. 2. On the one hand, pruning filters working on the same task (e.g., filters with green in V_A or filters with blue in V_B) can help to reduce the parameters of networks and accelerate the inference. On the other hand, merging filters with different tasks (e.g., filters with cyan in

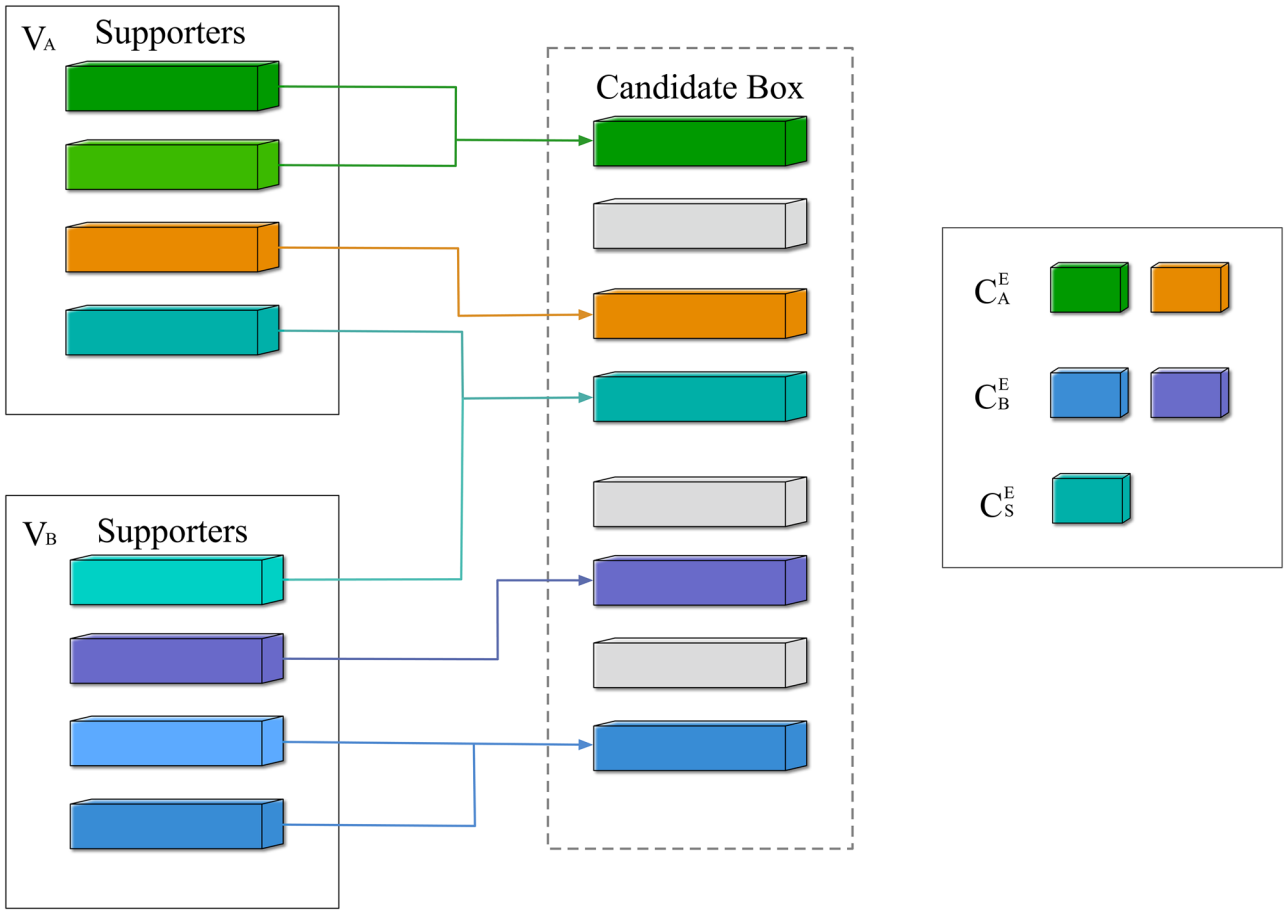

Fig. 2 Filter sharing in networks

V_A and V_B) can share information of two tasks. The orange and purple filters will be maintained in the new layers.

Then, we define $C^E = \{C_A^E, C_B^E, C_S^E\}$ with

$$\begin{cases} C_A^E = \{f_i | S_i \subseteq V_A\} \\ C_B^E = \{f_j | S_j \subseteq V_B\} \\ C_S^E = \{f_k | S_k \cap V_A \neq \emptyset \wedge S_k \cap V_B \neq \emptyset\} \end{cases} \tag{12}$$

where C_A^E, C_B^E and C_S^E consist of filters attributed to, respectively, task A, task B and both tasks; S_η is comprised of supporters with the same candidate f_η, which is defined by

$$S_\eta = \{f_i | \arg\min_j \ \delta\hat{E}(i,j) = \eta\}. \tag{13}$$

Note that $S_\eta \neq \emptyset$ because S_η contains at least f_η, namely, f_η will support itself.

Algorithm 1 The pruning and merging step

Input: elected candidates C^E; model M^A; model M^B
Output: updated model M^A; updated model M^B

for each $f_i \in C^E$ **do**
  Find S_i in (13).
  if $S_i \subseteq V_A$ or $S_i \subseteq V_B$ **then**
    Prune filters in $\{S_i \setminus f_i\}$.
  else
    if $f_i \in V_A$ **then**
      Find nearest f_k with f_i in $\{S_i \cap V_B\}$.
    else
      Find nearest f_k with f_i in $\{S_i \cap V_A\}$.
    end if
    Prune filters in $\{S_i \setminus \{f_k, f_i\}\}$.
    Merge and update f_i and f_k.
  end if
end for

Pruning and Merging

Suppose that layer l has N_l filters. If a pruning rate P_l is set for layer in network A and B, we would not get $P_l \times N_l$ filters pruned with $N^E = (1 - P_l) \times N_l$ for each network. Actually, the real sum of pruned filters is $2 \times P_l \times N_l - length(C_S^E)$ according to Algorithm 1.

Base on the aforementioned discussions, we have two strategies for pruning and merging step. **Strategy one** is to simply restrict the pruning rate P for the whole models and the real sum of reduced filters will be determined adaptively by FS strategy as well as the pruning rates on different tasks. **Strategy two** is to restrict the pruning rate for each network, under which we will prune additional $length(C_S^E)$ with norm-based criterion (pruning filters with small L_p norm) after filter sharing. Then the real pruning rate can be confirmed for each network. These two strategies can be chosen under different situations. Strategy one does not need to decide the specific rates on tasks. However, the real pruning rates cannot be confirmed. In contrast, although the rate on every task should be manually decided when using strategy two, pruning on networks are strictly executed under the preset rates.

MOP Model for Multi-task Pruning

Consequently, the MOP model is established as

$$\begin{aligned} &\min_{\alpha} \mathcal{F}(\boldsymbol{\alpha}) \\ &\quad \mathcal{F} = \{\mathcal{F}_1(\boldsymbol{\alpha}), \mathcal{F}_2(\boldsymbol{\alpha}), \mathcal{F}_3(\boldsymbol{\alpha}), \mathcal{F}_4(\boldsymbol{\alpha})\} \end{aligned} \tag{14}$$

In this paper, we shall use MOEA/D proposed in [30] to optimize the many-objective problem in (14). We provide the modified mutating strategy of MOEA/D as shown in Algorithm 2. Instead of randomly changing $\boldsymbol{\alpha}$, we impose a probabilistic constraint on the mutation of $\boldsymbol{\alpha}$ in the following way:

First, we remove the filter f_k from C_{cur}^E which is nearest to C_{cur}^E:

$$\begin{aligned} k = \arg\min_i &\delta E(i,j) \\ &\text{s.t. } i \neq j \end{aligned} \tag{15}$$

Then, we add a filter $f_k \in C \setminus C_{cur}^E$ into C_{cur}^E:

$$k = \arg\max_i \{\min_j \delta E(i,j)\}. \tag{16}$$

Since the PF $P = \{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_p\}$ obtained by MOP algorithm is not a single optimal solution, we should seek an optimal value for $\boldsymbol{\alpha}$. For instance, $\boldsymbol{\alpha}^*$ can be chosen as follows:

$$\begin{aligned} \boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} &\sum f_i(\boldsymbol{\alpha}) \\ &\text{s.t. } \sum \alpha_i = N^E \end{aligned} \tag{17}$$

where N^E is the limit size of C^E.

Algorithm 2 The revised mutating strategy

Input: current elected candidates C_{cur}^E; candidate set C, decision variable $\boldsymbol{\alpha}$

Output: updated $\boldsymbol{\alpha}$

```
Init random number r ∈ [0, 1].
if r > 0.5 then
    Set α_i = 0 for f_i, i is determined by (15).
    Set α_j = 1 for f_j, j is determined by (16).
else
    Randomly set α_i = 0 for f_i ∈ C^E_cur.
    Randomly set α_j = 1 for f_j ∈ {C \ C^E_cur}.
end if
```

How to Update Weights

Suppose $\delta E(i,j)$ is the difference between the original output results of f_i and the corresponding results replaced by f_j. We apply an alternative way introduced in [5, 31] to approximate the error function by Taylor series expansion as follows:

$$\begin{aligned} \delta E_l(i,j) &= E_l(j) - E_l(i) \\ &= (\frac{\partial E_l}{\partial \theta_l})^T \delta\theta_l + \frac{1}{2}\delta\theta_l{}^T \mathbf{H}_l \delta\theta_l + O(\|\delta\theta_l\|^3) \end{aligned} \tag{18}$$

where $l = 1, \ldots, L$ is the number of layers; $\delta\theta_l = \theta_{l,j} - \theta_{l,i}$ denotes the perturbation when f_i with parameter $\theta_{l,i}$ changes to f_j with parameter $\theta_{l,j}$; $\mathbf{H}_l = \partial^2 E_l / \partial \theta_l^2$ is the Hessian matrix which contains the whole second-order derivatives. The first term in (18) is vanished during training and higher-order terms $O(\|\delta\theta_l\|^3)$ can be regarded as 0 [5]. Hence, $\delta E_l^A(i,j)$ for task A can be redefined by

$$\delta E_l^A(i,j) = \frac{1}{2}(\delta\theta_l^A)^T \cdot \mathbf{H}_l^A \cdot \delta\theta_l^A \tag{19}$$

where $\mathbf{H}_l^A$ is calculated by

$$\mathbf{H}_l^A = \frac{1}{n} \sum x_l^A \cdot (x_l^A)^T \tag{20}$$

with x_l^A the input of layer l for task A.

Then, in order to merge filters with as less losses as possible, f_η can be updated by optimizing the following error function:

$$\min_{\delta\theta(1),\ldots,\delta\theta(n)} \sum_{f_i \in S_\eta} \delta E(i,\eta) \quad \text{s.t. } \delta\theta(1) + \theta(1) = \ldots = \delta\theta(n) + \theta(n) \tag{21}$$

where n is the number of filters in S_η. By applying the method of Lagrange multipliers, the optimal result $\theta(\eta)$ of f_η is obtained via the following formula:

$$\begin{aligned}\theta(\eta) =& \delta\theta(n) + \theta(n) \\ =& (\mathbf{H}(1) + \ldots + \mathbf{H}(n))^{-1} \cdot (\mathbf{H}(1)(\theta(1) - \theta(n)) + \\ & \ldots + \mathbf{H}(n-1)(\theta(n-1) - \theta(n))) + \theta(n)\end{aligned} \tag{22}$$

with $\mathbf{H}(n)$ denoting the Hessian matrix of f_n. Particularly, if all the filters have the same input, then $\theta(\eta)$ can be computed by

$$\theta(\eta) = \frac{1}{n} \sum_{f_i \in S_\eta} \theta(i) \tag{23}$$

It should be mentioned that computing $\mathbf{H}$ is still a tough work in some databases. Hence, we can calculate the distance instead (e.g., Euclidean distance) in the experiments:

$$\delta E(i,j) = dist(i,j) = \sum^{N_{l-1}} \sum^{K} \sum^{K} \|\theta_i - \theta_j\|^2 \tag{24}$$

where N_{l-1} is the input channels of layer l; $K \times K$ denotes the kernel size. When using Euclidean distance in (7), the selection under criterion $\mathcal{F}_3(\cdot)$ is similar to judging filters by geometric median in [7].

For SP-FIS, we only need to establish one network and prune the redundant filters by setting weights with 0. The MOP of SP-FIS is established according to (14) as follows,

$$\begin{aligned}&\min_{\alpha} \mathcal{F}(\boldsymbol{\alpha}) \\ &\mathcal{F} = \{\mathcal{F}_1(\boldsymbol{\alpha}), \mathcal{F}_3(\boldsymbol{\alpha}), \mathcal{F}_4(\boldsymbol{\alpha})\}\end{aligned} \tag{25}$$

For MP-FIS, distinguishing from [17] and [11], index sharing is proposed for multiple networks to keep network structure where sharing index will be preserved along with an index matrix I, shown in Fig. 3. To be specific, we record the indices of each single filter pointing to pruning, merging or remaining. Then, new networks can be easily recovered and trained according to the index matrix. The importance of keeping network structure has been proved in Section 4.4.1. Finally, the MFIS framework is described in Algorithm 3. It should be noticed that if a filter $f_i \in C_S^E$ is pruned in step 11 when using strategy two, it will be removed from C_S^E along with its shared filter in the other model.

In order to show the differences between MFIS framework and other methods more clearly, we compare with model compression methods, shown in Table 1.

Extending MFIS to Many-task Cases

MFIS can be easily extended to an M-task ($M \geq 3$) case which is illustrated in Fig. 4. Each network M_k can accept information from the other networks because FIS can determine each filter to be pruned, merged or remained for networks. In Fig. 4, original networks are presented with different color blocks in the bottom line. In the top block line, the grey blocks describe the unimportant filters selected by our algorithm and the color ones are the remained or merged filters for three tasks. Each filter may work for one to three tasks after pruning and merging.

Algorithm 3 The MFIS framework

Input: model M^A; model M^B; pruning rate P.
Output: index matrix I; updated M^A; updated M^B;

1: **for** $l = 1$ to L **do**
2: Calculate $\delta\hat{E}^A(i,j)$ and $\delta\hat{E}^B(i,j)$ for each f_i and f_j in models.
3: Optimize (14) with pruning rate P_l and obtain the optimal solution $\boldsymbol{\alpha}_l^*$ with the corresponding C_l^E.
4: Do pruning and merging step.
5: **if** using **strategy two** in III-C **then**
6: Prune additional $length(C_S^E)$ with small norms.
7: **end if**
8: Update index matrix I.
9: **end for**

Multi-task Filter Index Sharing Framework

We are now in a position to propose the MFIS framework. MFIS can be classified into the following two categories according to the number of tasks: single-task pruning FIS (SP-FIS) and multi-task pruning FIS (MP-FIS).

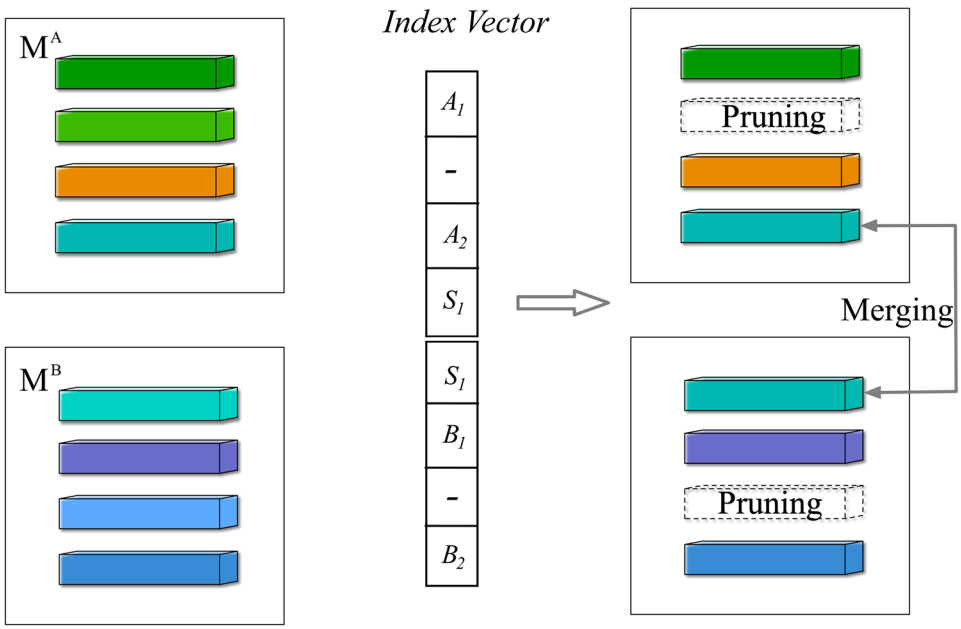

Fig. 3 The multi-task indexing steps

Experiments

In this section, we shall examine our MFIS framework on single-task pruning and multi-task pruning on various benchmarks with different networks. We will further explore the influence of keeping network structure and the balance on different tasks with MFIS. Finally, the MFIS will be extended to more task case.

Experimental Settings

Databases

In our experiments, the following databases are considered: MNIST database [32] is a small digital handwriting data set, which contains 60,000 images. CIFAR-10 and CIFAR-100 [33] contain 60,000 colour images in each database, in which 50,000 training images and 10,000 testing images are included. SVHN [34] is a real-world image dataset, which is obtained from house numbers in Google Street View images, with 73,257 digits for training and 26,032 digits for testing.

Table 1 The difference between MFIS and other methods

	prune	reduce Params.	reduce FLOPs	multi-task
PFEC [16]	✓	✓	✓	✗
MTZ [17]	✗	✓	✗	✓
SP-FIS	✓	✓	✓	✗
MP-FIS	✓	✓	✓	✓

ILSVRC-2012 [35] (ImageNet) is a large-scale dataset containing 1.28 million training images and 50,000 validation images of 1,000 classes.

Training Settings

Two types of network structure are used in the following experiments: VGG-net [37] and ResNet [38]. VGG used here is a slight modified version in [39] without Dropout [40], which contains only 2 FC layers. Compared with VGG-16 [37], the modified VGG has much fewer parameters in the fully connected layers. Hence, it could be more challenging to do compression work on it. We train the pre-trained networks with optimizer (stochastic gradient descent algorithm, SGD), initial learning rate (0.1) for baseline, momentum (0.9), batch size (128) and weight decay (0.0005). For CIFAR, the learning rate is divided by 5× at epoch 60, 120 and 160 and the network is trained for 200 epochs in total. For ImageNet, the learning rate is divided by 10× after every 30 epochs and the total epoch is 100. The training schedule of learning rate is set following the work in [7].

Pruning Settings

For pruning the scratch model, we utilize the regular training schedule without additional fine-tuning. For pruning, the pre-trained model with fine-tuning, the learning rate is reduced to one-tenth of the original learning rate [7]. For MOEA/D, we set the scale of the optimization iteration to be 200 for ResNet-20, 32, 56, 110 and 1000 for VGG-net and ResNet-50. The probabilities of mating and mutating are set to be 0.5 and 0.7.

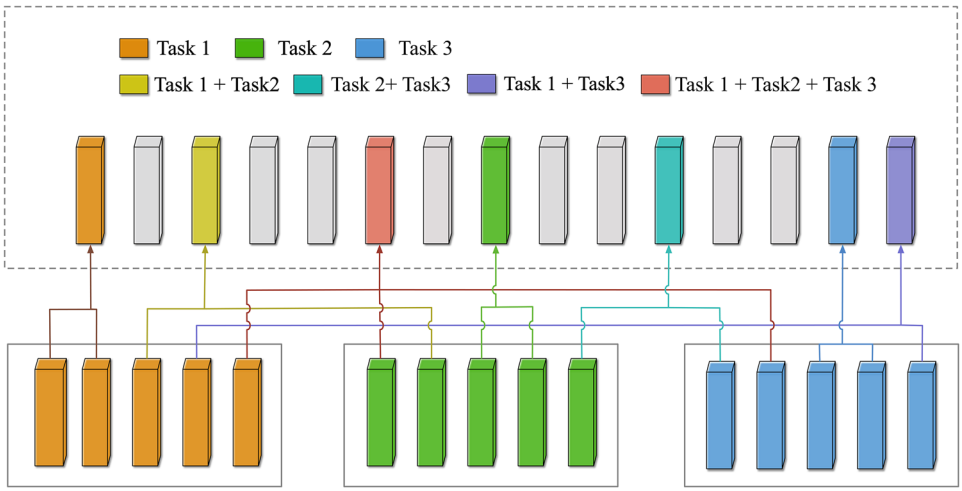

Fig. 4 The many-task case of MFIS

Performance Metric

To measure the network compression and testing performance, the following measurements are applied:

Acc. The accuracy of testing on database. Acc. ↓ (%) is the accuracy drop between pruned and the baseline model. The smaller, the better. For CIFAR-10, top-1 accuracy is provided, while for ILSVRC-2012, both top-1 and top-5 accuracies are reported.

Params. In this paper, Params. ↓ (%) indicates the percentage of reduced parameters in networks. Bigger Params. ↓ (%) means larger compression ratio.

FLOPs The overall floating point operations (FLOPs) denote the computation cost of networks. We use FLOPs ↓ (%) to describe the percentage of reduced FLOPs.

Single-task Pruning

For CIFAR-10 dataset, we test our SP-FIS on ResNet with depth 20, 32, 56, 110 and show the results in Table 2. We compare SP-FIS with existing single-task acceleration algorithms, namely, PFEC [16], CP [13], MIL [36], SFP [15], NISP [10], FPGM [7] and HRank [8]. For CIFAR, we run each setting three times and report the "mean ± std". In "Fine-tune?" column, "✓" and "✗" denote whether to use the pre-trained model as initialization or not, respectively. The experiment results validate the effectiveness of our method. Specifically, the best result of FPGM (FPGM-mix) on ResNet-110 accelerates the random initialized with an improvement of 0.17% on accuracy, but our SP-FIS achieves the results of 0.33% improvement with the same speedup ratio. Likewise, on ResNet-56, FPGM achieves the best result of 0.66% accuracy drop with FPGM-only, while our SP-FIS gains better result of only 0.32% accuracy drop. In comparison to SFP, our SP-FIS shows 0.49% accuracy drop which is better than 0.55% for SFP on ResNet-32, while the acceleration rate of 53.2% is also much better than 41.5% for SFP. To make the comparison more clearly, we also provide a result of 41.5% acceleration rate for SP-FIS, which gives an accuracy drop of only 0.18%. For pre-trained models, SP-FIS also achieve comparable results compared to other state-of-the-art methods. SP-FIS gives superior results because our method selects proper candidates for both pre-trained and randomly initialized networks by taking into account the three objectives mentioned earlier.

The proposed framework is also tested on ILSVRC-2012 with pre-trained ResNet-50. Following [7], we do not prune the projection shortcuts for simplification. The results are described in Table 3 and compared with six state-of-the-art methods.

Multi-task Pruning

In the experiments of MP-FIS, we first perform MP-FIS on VGG-net with pruning rate P on the whole models. Because few methods prune multi-task networks simultaneously, the following two methods are utilized for the comparison with our MP-FIS, namely, MTZ [17], a multi-task model compression method but without pruning, and PFEC [16], a pruning method for single task. All models use the same pre-trained networks. All models use the same pre-trained networks.

The experiment results are described in Table 4, where " / " denotes no result is provided. Considering that PFEC is a single-task method, we perform it on CIFAR-10 and CIFAR-100 separately with two types of compression ratio strategies: 49.90% parameters reduced on CIFAR-10 and 49.74% on CIFAR-100; 71.19% parameters reduced on CIFAR-10 and 27.41% on CIFAR-100. Note that MTZ does not accelerate the networks (no result on "FLOPs ↓ (%)" column) because of lack of pruning step. In "Params. ↓ (%)" column, the percentage of reduced parameters is given in

Table 2 Comparison of single-task pruning of ResNet on CIFAR-10

Depth	Method	Fine-tune?	Baseline Acc. (%)	Pruned Acc. (%)	Acc. ↓ (%)	FLOPs	FLOPs ↓ (%)
	SFP [15]	✗	92.20 (±0.18)	90.83 (±0.31)	1.37	2.43E7	42.2
20	FPGM-only [7]	✗	92.20 (±0.18)	90.44 (±0.20)	1.76	1.87E7	54.0
	FPGM-mix [7]	✗	92.20 (±0.18)	90.62 (±0.17)	1.58	1.87E7	54.0
	Ours (SP-FIS)	✗	92.20 (±0.18)	90.78 (±0.17)	1.42	1.87E7	54.0
	MIL [36]	✗	92.33	90.74	1.59	4.70E7	31.2
	SFP [15]	✗	92.63 (±0.70)	92.08 (±0.08)	0.55	4.03E7	41.5
	Ours (SP-FIS)	✗	92.63 (±0.70)	92.45 (±0.02)	0.18	4.03E7	41.5
32	FPGM-only [7]	✗	92.63 (±0.70)	91.93 (±0.03)	0.70	3.23E7	53.2
	FPGM-mix [7]	✗	92.63 (±0.70)	91.91 (±0.21)	0.72	3.23E7	53.2
	Ours (SP-FIS)	✗	92.63 (±0.70)	92.14 (±0.11)	0.49	3.23E7	53.2
	PFEC [16]	✗	93.04	91.31	1.75	9.09E7	27.6
	CP [13]	✗	92.80	90.90	1.90	-	50.0
	SFP [15]	✗	93.59 (±0.58)	92.26 (±0.31)	1.33	5.94E7	52.6
	FPGM-only [7]	✗	93.59 (±0.58)	92.93 (±0.49)	0.66	5.94E7	52.6
	FPGM-mix [7]	✗	93.59 (±0.58)	92.89 (±0.32)	0.70	5.94E7	52.6
	Ours (SP-FIS)	✗	93.59 (±0.58)	93.27 (±0.20)	0.32	5.94E7	52.6
56	PFEC [16]	✓	93.04	93.06	-0.02	9.09E7	27.6
	NISP [10]	✓	-	-	0.03	-	42.6
	CP [13]	✓	92.80	91.80	1.00	-	50.0
	HRank [8]	✓	93.26	93.17	0.09	6.27E7	50.0
	SFP [15]	✓	93.59 (±0.58)	93.35 (±0.31)	0.24	5.94E7	52.6
	FPGM-only [7]	✓	93.59 (±0.58)	93.49 (±0.13)	0.10	5.94E7	52.6
	FPGM-mix [7]	✓	93.59 (±0.58)	93.26 (±0.03)	0.33	5.94E7	52.6
	Ours (SP-FIS)	✓	93.76	93.57 (±0.04)	0.19	5.94E7	52.6
	MIL [36]	✗	93.63	93.44	0.19	-	34.2
	PFEC [16]	✗	93.53	92.94	0.61	1.55E8	38.6
	SFP [15]	✗	93.68 (±0.32)	93.38 (±0.30)	0.30	1.50E8	40.8
	FPGM-only [7]	✗	93.68 (±0.32)	93.73 (±0.23)	-0.05	1.21E8	52.3
	FPGM-mix [7]	✗	93.68 (±0.32)	93.85 (±0.11)	-0.17	1.21E8	52.3
110	Ours (SP-FIS)	✗	93.68 (±0.32)	94.01 (±0.08)	-0.33	1.21E8	52.3
	PFEC [16]	✓	93.53	93.30	0.20	1.55E8	38.6
	NISP [10]	✓	-	-	0.18	-	43.8
	FPGM-only [7]	✓	93.68 (±0.32)	93.74 (±0.10)	-0.06	1.21E8	52.3
	HRank [8]	✓	93.50	93.36	0.14	1.06E8	58.2
	Ours (SP-FIS)	✓	94.05	94.16 (±0.09)	-0.11	1.21E8	52.3

Table 3 Comparison of pruning ResNet-50 on ILSVRC-2012

Depth	Method	Baseline Top-1 Acc.(%)	Baseline Top-5 Acc.(%)	Pruned Top-1 Acc.(%)	Pruned Top-5 Acc.(%)	Top-1 Acc.↓ (%)	Top-5 Acc.↓(%)	FLOPs↓ (%)
	ThiNet [14]	72.88	91.14	72.04	90.67	0.84	0.47	36.7
	SFP [15]	76.15	92.87	62.14	84.60	14.01	8.27	41.8
	HRank [8]	76.15	92.87	74.98	92.33	1.17	0.54	43.8
50	NISP [10]	-	-	-	-	0.89	-	44.0
	CP [13]	-	92.20	-	90.80	-	1.40	50.0
	FPGM [7]	76.15	92.87	74.83	92.32	1.32	0.55	53.5
	Ours (SP-FIS)	76.15	92.87	75.23	92.50	0.92	0.37	53.5

Table 4 Comparison of multi-task pruning of VGG-net on CIFAR-10 and CIFAR-100

Method	Acc.(%)		Acc. ↓ (%)		Params.↓ (%)			FLOPs ↓ (%)	
	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100	whole	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
	93.41	/	-0.14	/	/	49.90	/	21.72	/
PFEC [16]	/	70.58	/	0.97	/	/	49.74	/	21.72
	93.45	/	-0.18	/	/	71.19	/	31.15	/
	/	71.19	/	0.36	/	/	27.41	/	11.93
MTZ [17]	93.31	71.16	-0.04	0.39	43.25	/	/	/	/
MP-FIS w/o H	93.52	71.42	-0.25	0.13	49.92	72.38	27.53	31.68	10.50
MP-FIS with H	93.48	71.80	-0.21	-0.25	49.45	70.61	28.36	30.00	11.22

three conditions: whole networks, network on CIFAR-10 and network on CIFAR-100, due to the existence of shared parameters in MTZ and MP-FIS. For MP-FIS, MP-FIS "with H" and "w/o H" describe computing δE by (19) or computing Euclidean distance instead. Obviously, it can be seen that the proposed MP-FIS performs much better in comparison to other methods.

The advantages of MP-FIS can be attributed to the fact that it allows access to relevant information in different tasks and limits redundant information in the same task, while traditional pruning methods only focus on one single task and do not take into account the facilitation of multi-task for learning. Although MTZ can deal with multiple tasks, it just simply forces one-to-one merging of filters between different tasks without pruning, which may lead to adverse outcome for tasks, especially when tasks are not strongly correlated, such as CIFAR-10 and CIFAR-100.

Then, we test our method with pruning rate P on each network of ResNet with depth 32, 56, and 110. The additional *length* (C_S^E) filters are pruned with norm-based criterion. As shown in shown in Table 5, we record the best accuracy results for two tasks, respectively, which means that filters in C_S^E are not forced to be shared for testing in this experiment. Likewise, as a single-task pruning method, FPGM is performed on CIFAR-10 and CIFAR-100 separately for comparison. The result also validates the effectiveness of our method.

More Explorations

Influence of Keeping Network Structure

For the purpose of proving the necessity of keeping network structure, we define TE to scale the tendency of sharing information from different tasks in the next layer as follows:

$$TE = \frac{1 - (\delta E^A(A,B) + \delta E^B(B,A))}{1 - (\delta E^A(A,A) + \delta E^B(B,B))} \tag{26}$$

where $\delta E^A(A,B)$ represents the average loss on model A of replacing filters in model A with those in model B. For pre-trained models, Lenet-300-100 network [32], which is a fully connected network with 2 FC hidden layers will be trained on MNIST database. We change the limit size in the first FC layer and then observe the variation of TE in the second FC layer. The results are shown in Fig. 5, from which we can observe that, without keeping network structure, the difference between filters becomes much bigger in two models. Such a phenomenon is mainly due to the fact that the filter similarity between two models has

Table 5 Comparison of multi-task pruning of ResNet on CIFAR-10 and CIFAR-100

Depth	Method	Baseline Acc. (%)		Pruned Acc. (%)		Acc. ↓ (%)		FLOPs ↓ (%)	
		CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
	FPGM [7]	93.18	/	92.75	/	0.43	/	41.5	/
32		/	70.28	/	69.44	/	0.84	/	41.5
	Ours (MP-FIS)	93.18	70.28	92.96	70.12	0.22	0.16	41.5	41.5
	FPGM [7]	93.76	/	93.55	/	0.21	/	52.6	/
56		/	71.79	/	70.51	/	1.28	/	52.6
	Ours (MP-FIS)	93.76	71.79	93.60	71.16	0.16	0.63	52.6	52.6
	FPGM [7]	94.05	/	94.22	/	-0.17	/	52.3	/
110		/	73.88	/	72.80	/	1.08	/	52.3
	Ours (MP-FIS)	94.05	73.88	94.22	73.04	-0.17	0.84	52.3	52.3

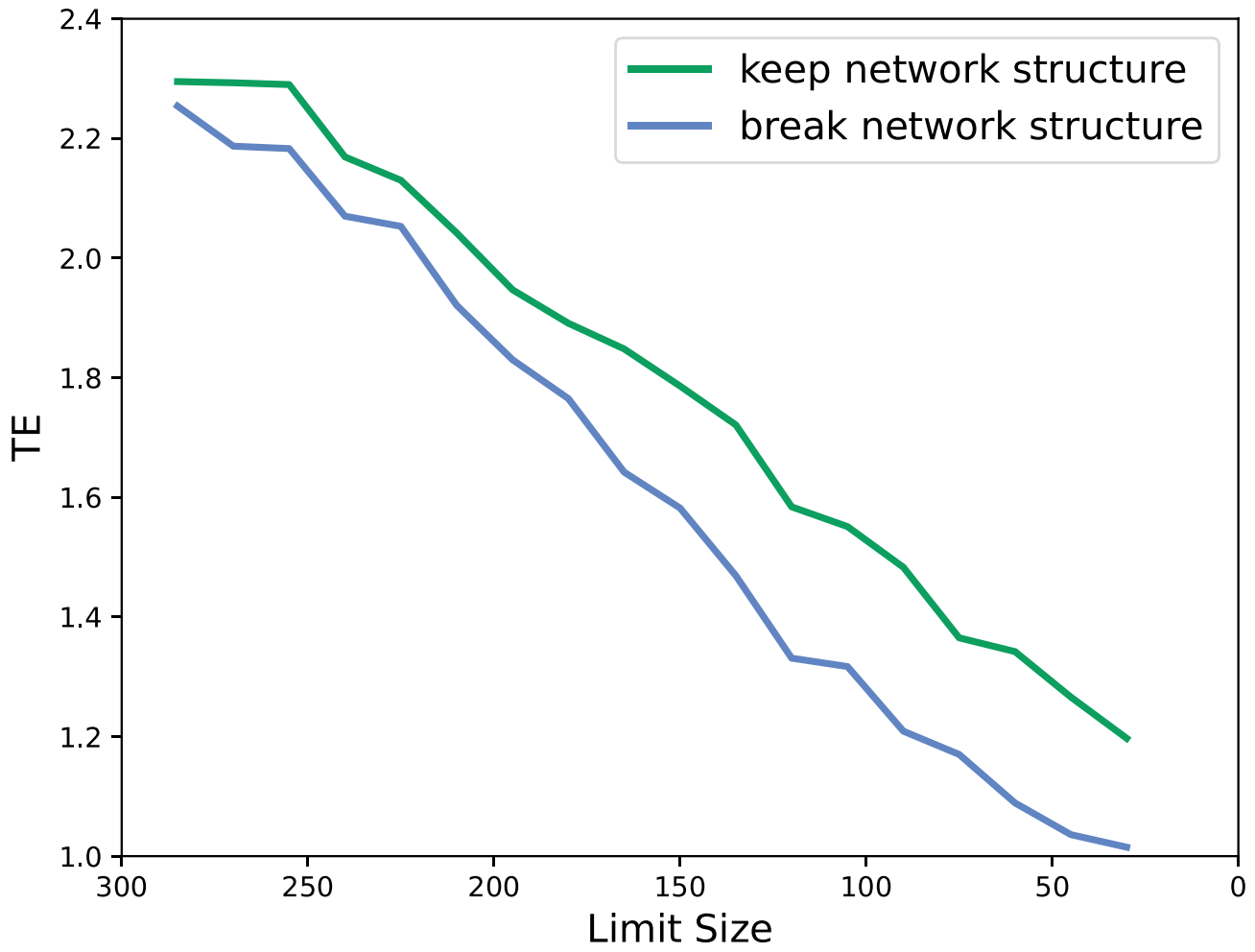

Fig. 5 The tendency *TE* of sharing filters from different tasks in the second FC layer with and without keeping layer structure

been influenced after breaking layer structure in the pruning step.

Balance on Tasks

With the hope of further showing that MFIS is able to prune the most appropriate filters between networks, we depict, in Fig. 6, the pruning rates of CIFAR-10 and CIFAR-100 in the last 6 layers of VGG-net. It is seen that the average pruning rate in CIFAR-100 is far smaller than that in CIFAR-10, as for VGG-net, network of CIFAR-10 has much more redundant filters compared with CIFAR-100. This has also been proved in Table 4 that PFEC achieves similar accuracies with 71.19% or 49.90% parameters reduced on CIFAR-10, while the performance is much worse with 49.74% or 27.41% parameters reduced on CIFAR-100.

Extending MFIS to Many-task Cases

We extend MFIS to a 3-task (CIFAR-10, CIFAR-100 and SVHN) case and give the results in Table 6. We can see that the results of the MP-FIS ($M = 3$) are further improved compared to MP-FIS ($M = 2$).

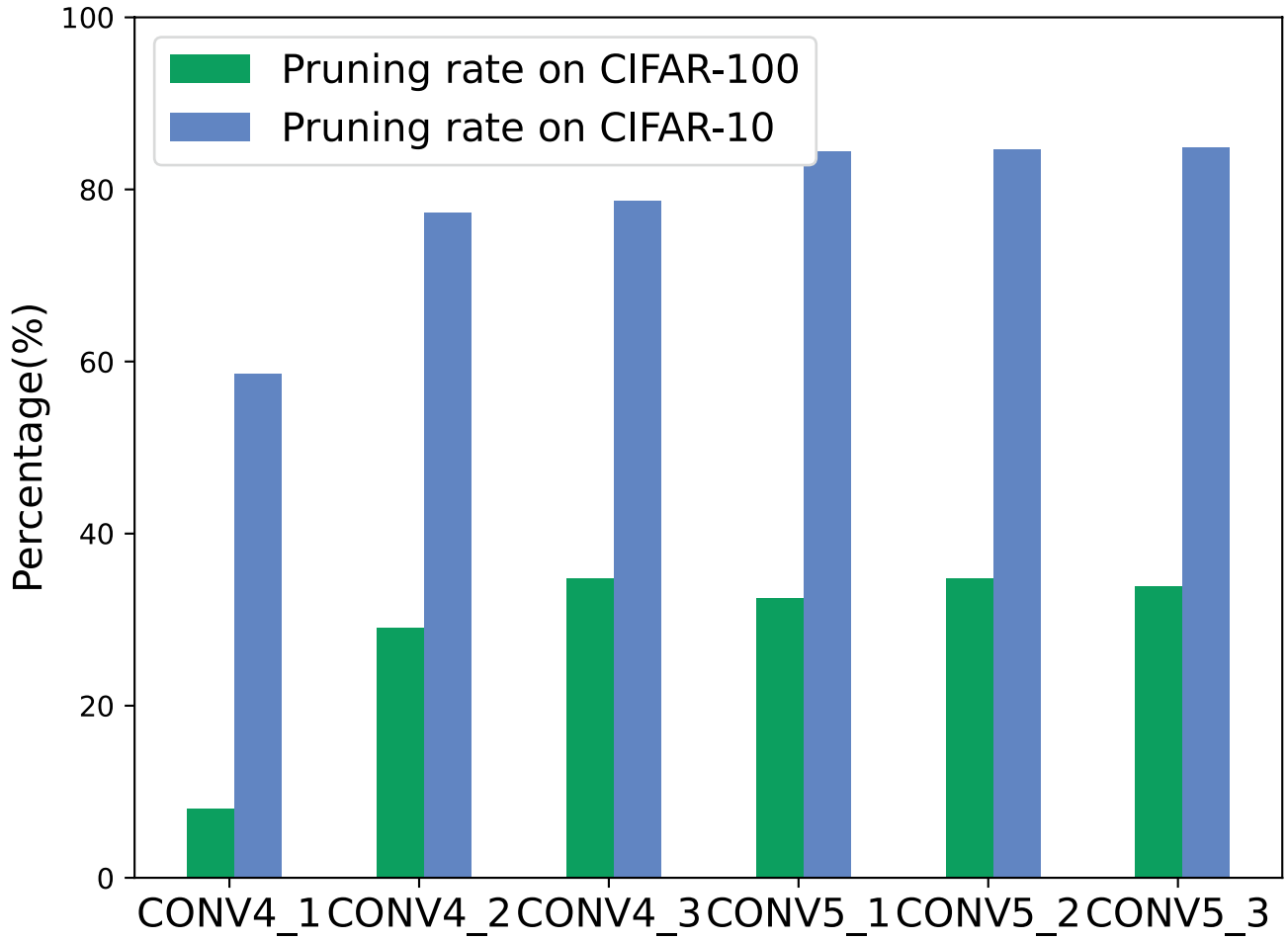

Fig. 6 The pruning rate of networks on different tasks in the last 6 layer

Table 6 Pruning ResNet on more than two tasks

Method	Task-1 Acc. (%)	Task-2 Acc. (%)	Task-3 Acc. (%)	FLOPs ↓ (%)
baseline	93.18	70.28	96.15	/
MP-FIS ($M = 2$)	92.96	70.12	/	41.5
MP-FIS ($M = 3$)	92.99	70.32	96.33	41.5

Conclusion

In this paper, a multi-task pruning algorithm MFIS has been provided by virtue of the filter index sharing approach. A filter sharing strategy has been proposed capable of automatically sharing filters of both inner networks (pruning) and external networks (merging). Through FIS, the proposed algorithm can determine the most suitable operation for filters. Three criteria have been developed to select important filters of multi-task networks. The proposed MFIS can deal with both multi-task pruning and single-network pruning by converting the optimization problem into MOP. In experiment parts, our proposed framework has shown much better performance in comparison to certain state-of-the-art methods via tests based on several benchmarks. It is has also been proved that MFIS works well on many-task case. However, there are still certain limitations of the developed MFIS. For instance, the criteria proposed in this paper for MFIS are only applicable for image classification task. In the future, some other type of tasks (e.g., semantic segmentation) should be considered in applications. One of the potential research topics might be taking more sorts of criteria into consideration and applying the framework into real-world applications [2, 3, 41–59]

Funding Information This work was supported in part by the National Natural Science Foundation of China under Grants 61701238, 11431015, 61773209, 61873148 and 61933007, the Natural Science Foundation of Jiangsu Province of China under Grant BK20190021, the Six Talent Peaks Project in Jiangsu Province of China under Grant XYDXX-033, the Royal Society of the U.K., and the Alexander von Humboldt Foundation of Germany.

Declarations

Conflict of Interest The authors declare that they have no conflicts of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Ranjan R, Patel VM, Chellappa R. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation and gender recognition. IEEE Trans Patt Anal Mach Intell. 2017;41(1):121–35.
2. Ieracitano C, Mammone N, Bramanti A, Hussain A, Morabito FC. A convolutional neural network approach for classification of dementia stages based on 2d-spectral representation of EEG recordings. Neurocomputing. 2019;323:96–107.
3. Ieracitano C, Paviglianiti A, Campolo M, Hussain A, Pasero E, Morabito FC. A novel automatic classification system based on hybrid unsupervised and supervised machine learning for electrospun nanofibers. IEEE/CAA J Automatica Sinica. 2021;8(1):64–76.
4. Carreira-Perpinán MA, Idelbayev Y. Learning-compression algorithms for neural net pruning, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018:8532–8541.
5. Dong X, Chen S, Pan S. Learning to prune deep neural networks via layer-wise optimal brain surgeon, in Advances in Neural Information Processing Systems. 2017:4857–4867.
6. Zhong G, Liu W, Yao H, Li T, Liu X. Merging similar neurons for deep networks compression. Cogn Comput. 2020;12(6):577–88.
7. He Y, Liu P, Wang Z, Hu Z, Yang Y. Filter pruning via geometric median for deep convolutional neural networks acceleration, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019:4340–4349.
8. Lin M, Ji R, Wang Y, Zhang Y, Zhang B, Tian Y, Shao L. Hrank: Filter pruning using high-rank feature map, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020:1529–1538.
9. Lin M, Ji R, Zhang Y, Zhang B, Wu Y, Tian Y. Channel pruning via automatic structure search, in International Joint Conference on Artificial Intelligence. 2020:673–679.
10. Yu R, Li A, Chen CF, Lai JH, Morariu VI, Han X, Gao M, Lin CY, Davis LS. Nisp: Pruning networks using neuron importance score propagation, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018:9194–9203.
11. He X, Gao D, Zhou Z, Tong Y, Thiele L. Disentangling redundancy for multi-task pruning. 2019. arXiv preprint arXiv:1905.09676.
12. Jin Y, Sendhoff B. Pareto-based multiobjective machine learning: An overview and case studies, IEEE Transactions on Systems, Man and Cybernetics. Part C (Appl Rev). 2008;38(3):397–415.
13. He Y, Zhang Z, Sun J. Channel pruning for accelerating very deep neural networks, in Proceedings of the IEEE International Conference on Computer Vision. 2017:1389–1397.
14. Luo JH, Wu J, Lin W. Thinet: A filter level pruning method for deep neural network compression, in Proceedings of the IEEE International Conference on Computer Vision. 2017:5058–5066.
15. He Y, Kang G, Dong X, Fu Y, Yang Y. Soft filter pruning for accelerating deep convolutional neural networks, in International Joint Conference on Artificial Intelligence. 2018:2234–2240.
16. Li H, Kadav A, Durdanovic I, Samet H, Graf HP. Pruning filters for efficient convnets. 2016. arXiv preprint arXiv:1608.08710.
17. He X, Zhou Z, Thiele L. Multi-task zipping via layer-wise neuron sharing, in Advances in Neural Information Processing Systems. 2018:6016–602.
18. Dai B, Zhu C, Guo B, Wipf D. Compressing neural networks using the variational information bottleneck, in International Conference on Machine Learning. 2018;1135–1144.
19. Li Z, Hoiem D. Learning without forgetting. IEEE Trans Patt Anal Machi Intell. 2017;40(12):2935–47.
20. Long M, Cao Z, Wang J, Philip SY. Learning multiple tasks with multilinear relationship networks, in Adv Neural Info Proc Syst. 2017:1594–1603.
21. Ruder S. An overview of multi-task learning in deep neural networks. 2017. arXiv preprint arXiv:1706.05098.
22. Zhang Y, Yang Q. A survey on multi-task learning. 2017. arXiv preprint arXiv:1707.08114.
23. Kendall A, Gal Y, Cipolla R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018:7482–7491.
24. Yang Y, Hospedales T. Deep multi-task representation learning: A tensor factorisation approach. 2016. arXiv preprint arXiv:1605.06391.
25. Misra I, Shrivastava A, Gupta A, Hebert M. Cross-stitch networks for multi-task learning, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016:3994–4003.
26. Meyerson E, Miikkulainen R. Beyond shared hierarchies: Deep multitask learning through soft layer ordering. 2017. arXiv preprint arXiv:1711.00108.
27. Lu Y, Kumar A, Zhai S, Cheng Y, Javidi T, Feris R. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017:5334–5343.
28. Meng F, Cheng H, Li K, Xu Z, Ji R, Sun X, Lu G. Filter grafting for deep neural networks, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020:6599–6607.
29. Ye J, Lu X, Lin Z, Wang JZ. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. 2018. arXiv preprint arXiv:1802.00124.
30. Zhang Q, Li H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evo Comput. 2007;11(6):712–31.
31. Hassibi B, Stork D. Second order derivatives for network pruning: Optimal brain surgeon. Advances in Neural Information Processing Systems. 1992;5:164–71.
32. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278–324.
33. Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images, 2009.

34. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY. Reading digits in natural images with unsupervised feature learning. 2011.
35. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Alexander CB, Li F-F. Imagenet large scale visual recognition challenge. Int J Comp Vis. 2015;115(3):211–52.
36. Dong X, Huang J, Yang Y, Yan S. More is less: A more complicated network with less inference complexity, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017:5840–5848.
37. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014. arXiv preprint arXiv:1409.1556.
38. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016:770–778.
39. Zagoruyko S. 92.45% on cifar-10 in torch, [EB/OL]. 2015. Available: http://torch.ch/blog/2015/07/30/cifar.html.
40. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15(1):1929–58.
41. Cao J, Bu Z, Wang Y, Yang H, Jiang J, Li H-J. Detecting prosumer-community group in smart grids from the multiagent perspective. IEEE Trans Syst Man Cybernet: Syst. 2019;49(8):1652–64.
42. Cao J, Wang B, Brown D. Similarity based leaf image retrieval using multiscale R-angle description. Info Sci. 2016;374:51–64.
43. Chen Y, Wang Z, Wang L, Sheng W. Mixed H_2/H_∞ state estimation for discrete-time switched complex networks with random coupling strengths through redundant channels. IEEE Trans Neural Net Learn Sys. 2020;31(10):4130–42.
44. Cheng H, Wang Z, Wei Z, Ma L, Liu X. On adaptive learning framework for deep weighted sparse autoencoder: A multiobjective evolutionary algorithm, IEEE Transactions on Cybernetics, in press, https://doi.org/10.1109/TCYB.2020.3009582.
45. Li Q, Wang Z, Li N, Sheng W. A dynamic event-triggered approach to recursive filtering for complex networks with switching topologies subject to random sensor failures. IEEE Trans Neural Net Learn Syst. 2020;31(10):4381–8.
46. Liu D, Wang Z, Liu Y, Alsaadi FE. Extended Kalman filtering subject to random transmission delays: Dealing with packet disorders. Info Fusion. 2020;60:80–6.
47. H. Liu, Z. Wang, B. Shen and H. Dong, Delay-distribution-dependent H_∞ state estimation for discrete-time memristive neural networks with mixed time-delays and fading measurements, IEEE Transactions on Cybernetics, vol. 50, no. 2, pp. 440–451, 2020.
48. Liu W, Wang Z, Zeng N, Yuan Y, Alsaadi FE, Liu X. A novel randomised particle swarm optimizer. Int J Mach Learning Cybernet. 2021;12(2):529–40.
49. Liu Y, Chen S, Guan B, Xu P. Layout optimization of large-scale oil-gas gathering system based on combined optimization strategy. Neurocomputing. 2019;332:159–83.
50. Liu Y, Cheng Q, Gan Y, Wang Y, Li Z, Zhao J. Multi-objective optimization of energy consumption in crude oil pipeline transportation system operation based on exergy loss analysis. Neurocomputing. 2019;332:100–10.
51. Qian W, Li Y, Chen Y, Liu W. Filtering for stochastic delayed systems with randomly occurring nonlinearities and sensor saturation. Int J Syst Sci. 2020;51(13):2360–77.
52. Qian W, Li Y, Zhao Y, Chen Y. New optimal method for L_2-L_∞ state estimation of delayed neural networks. Neurocomputing. 2020;415:258–65.
53. Yang H, Wang Z, Shen Y, Alsaadi FE, Alsaadi FE. Event-triggered state estimation for Markovian jumping neural networks: On mode-dependent delays and uncertain transition probabilities. Neurocomputing. 2021;424:226–35.
54. Yue W, Wang Z, Liu W, Tian B, Lauria S, Liu X. An optimally weighted user- and item-based collaborative filtering approach to predicting baseline data for Friedreich's Ataxia patients. Neurocomputing. 2021;419:287–94.
55. Zhao D, Wang Z, Wei G, Han QL. A dynamic event-triggered approach to observer-based PID security control subject to deception attacks, Automatica. 2020;120 (109128).
56. Zhao Z, Wang Z, Zou L, Guo J. Set-Membership filtering for time-varying complex networks with uniform quantisations over randomly delayed redundant channels. Int J Syst Sci. 2020;51(16):3364–77.
57. Zou L, Wang Z, Hu J, Liu Y, Liu X. Communication-protocol-based analysis and synthesis of networked systems: Progress, prospects and challenges, Int J Syst Sci. in press, https://doi.org/10.1080/00207721.2021.1917721.
58. Zou L, Wang Z, Hu J, Zhou DH. Moving horizon estimation with unknown inputs under dynamic quantization effects. IEEE Trans Auto Control. 2020;65(12):5368–75.
59. Zou L, Wang Z, Zhou DH. Moving horizon estimation with non-uniform sampling under component-based dynamic event-triggered transmission, Automatica. 2020;120(109154).
60. Han S, Pool J, Tran J, Dally W. Learning both weights and connections for efficient neural network. Adv Neural Info Proc Syst. 2015;28:1135–43.
61. He Y, Dong X, Kang G, Fu Y, Yan C, Yang Y. Asymptotic soft filter pruning for deep convolutional neural networks. IEEE Trans Cybernet. 2019;50(8):3594–604.
62. Liu Z, Mu H, Zhang X, Guo Z, Yang X, Cheng KT, Sun J. Metapruning: Meta learning for automatic neural network channel pruning, in Proceedings of the IEEE International Conference on Computer Vision. 2019:3296–3305.