Neural Network Pruning by Cooperative Coevolution

Haopu Shang[1] , **Jia-Liang Wu**[1] , **Wenjing Hong**[2] and **Chao Qian**[1*]

[1]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[2]Department of Computer Science and Engineering
Southern University of Science and Technology, Shenzhen 518055, China

{shanghp, wujl, qianc}@lamda.nju.edu.cn, hongwj@sustech.edu.cn

Abstract

Neural network pruning is a popular model compression method which can significantly reduce the computing cost with negligible loss of accuracy. Recently, filters are often pruned directly by designing proper criteria or using auxiliary modules to measure their importance, which, however, requires expertise and trial-and-error. Due to the advantage of automation, pruning by evolutionary algorithms (EAs) has attracted much attention, but the performance is limited for deep neural networks as the search space can be quite large. In this paper, we propose a new filter pruning algorithm CCEP by cooperative coevolution, which prunes the filters in each layer by EAs separately. That is, CCEP reduces the pruning space by a divide-and-conquer strategy. The experiments show that CCEP can achieve a competitive performance with the state-of-the-art pruning methods, e.g., prune ResNet56 for 63.42% FLOPs on CIFAR10 with -0.24% accuracy drop, and ResNet50 for 44.56% FLOPs on ImageNet with 0.07% accuracy drop.

1 Introduction

Convolution neural networks (CNNs) have achieved great success in computer vision tasks, such as image classification [He *et al.*, 2016] and object recognition [Liu *et al.*, 2020]. However, modern CNNs are typically too computationally intensive to be deployed on resource-constrained applications. To address this issue, a variety of methods have been proposed to reduce their computational costs [Hinton *et al.*, 2015; Han *et al.*, 2016]. Among them, neural network pruning has achieved impressive results in various applications by removing redundant parameters while keeping a high accuracy, as modern CNNs are usually overparameterized.

Despite its broad application prospects, the large number of parameters of CNNs poses a great challenge to neural network pruning. Performing the pruning process on the filter

level can alleviate this issue to some extent. However, most previous filter pruning methods rely heavily on some criteria or auxiliary modules, which are used to measure the importance of each filter, but are usually derived from the experience and knowledge of experts. Thus, an automatic framework is essential to facilitate the application of filter pruning on various architectures and data sets.

Filter pruning can be generally formulated as a complex optimization problem with the aim of searching for a good subset of the original filters to be retained such that the resultant pruned network has a low computational cost while keeping a high accuracy. Evolutionary algorithms (EAs) [Bäck, 1996] are a kind of general-purpose heuristic randomized optimization algorithms, inspired by natural evolution. They have been naturally applied to solve the optimization problem of pruning, making the process automatic [Yao, 1999]. However, unlike the multilayer perceptrons of earlier years, the large number of filters of modern CNNs implies a large search space, making EAs difficult to obtain a satisfactory solution within a limited computational overhead. As a consequence, existing EA-based pruning methods mainly take compromises when applied to modern CNNs. One way is to apply EAs to optimize the hyper-parameters (e.g., the pruning ratio of each layer [Li *et al.*, 2018]) of existing pruning methods with expert knowledge. Others often use indirect encoding methods to compress the search space partly [Zhou *et al.*, 2021; Fernandes and Yen, 2021], which may, however, degrade the performance due to the loss of information.

To solve the essential large-scale neural network pruning problem, this paper proposes a novel Cooperative CoEvolution algorithm for Pruning (CCEP). Cooperative coevolution uses the idea of divide-and-conquer, and has been shown to be effective for solving large-scale optimization problems [Hong *et al.*, 2019]. Although cooperative coevolutionary EAs have been proposed before [Ma *et al.*, 2019], none of them has been designed with the explicit purpose to solve neural network pruning problems. The main idea of CCEP is to use the strength of cooperative coevolution to overcome the large-scale issue in neural network pruning, and meanwhile utilize the property of neural networks for the non-trivial separation issue of cooperative coevolution. Specifically, considering the characteristics of forward propagation that features are processed layer by layer, CCEP divides the original search space by layer because the impact of removing a filter

is mainly related to whether the other filters in the same layer can retain the representation ability.

Experiments using ResNet and VGG on CIFAR-10 and ImageNet show that the proposed CCEP can achieve a competitive performance with a number of state-of-the-art pruning methods which require experts to design good criteria or auxiliary modules. Furthermore, compared with previous automatic EA-based pruning methods, CCEP can achieve both larger pruning ratio and smaller accuracy drop. Note that CCEP can be flexibly applied to different network architectures and data sets, since no complex rules or modules are employed and the optimization process is highly decoupled with the architecture. Due to the advantage of cooperative coevolution, CCEP can also be easily parallelized, which is friendly to the application in practice.

2 Related Work

2.1 Neural Network Pruning

Neural network pruning on CNNs has been widely studied in the past decade. The methods can be generally divided into non-structured and structured pruning, where the former is directly conducted on the weights and the latter is usually on the basic structured cells such as convolution filters. Nowadays, filter pruning has prevailed since it is friendly to most existing hardware and software frameworks.

Based on how to indicate the filters to be pruned, the mainstream filter pruning methods can be further categorized into criteria-based and learning-based methods. Criteria-based methods design some criteria artificially to identify unimportant filters and prune them. For example, L1 [Li *et al.*, 2017] prunes the filters with smaller l_1-norm weights. FPGM [He *et al.*, 2019] calculates the geometric median of the filters in each layer and prunes the neighbouring filters. HRank [Lin *et al.*, 2020] locates the low-rank features and prunes the corresponding filters. ThiNet [Luo *et al.*, 2019] prunes the filters by minimizing the reconstruction error. However, it is not easy to decide a proper criterion as well as the pruning ratio of each layer, which usually require rich experience and plenty of trials of experts. Learning-based methods use auxiliary modules to model the importance of filters in each layer and prune the neural network accordingly. For example, AutoPruner [Luo and Wu, 2020] uses learnable indicators on filters as the reference to prune. BCNet [Su *et al.*, 2021] introduces a bilaterally coupled network to efficiently evaluate the performance of a pruned network, and performs the pruning based on this surrogate model. Such methods can liberate the trials on designing proper criteria, but usually lead to a complex pruning framework and deteriorate the extendibility.

2.2 Pruning by Evolution

Since the 1990s, EAs have been applied to prune artificial neural networks automatically [Yao, 1999], with the aim of removing unimportant neural cells and connections while keeping a high accuracy. As the scales of neural networks were small, EAs could be easily applied to achieve a good performance. However, in recent years, the architectures of neural networks have become more and more complex and large, hindering the direct application of EAs, since the large search space makes EAs hard to produce a good pruned network in acceptable running time.

The current EA-based pruning methods often search in a compressed filter space, e.g., DeepPruningES [Fernandes and Yen, 2021] shares an encoding in different layers, and ESNB [Zhou *et al.*, 2021] uses block-level search. Instead of pruning neural networks directly, EAs are also applied to tune the hyper-parameters of existing pruning methods, e.g., OLMP [Li *et al.*, 2018] searches a proper pruning threshold of each layer for layer-wise magnitude-based pruning methods. However, all these compromised strategies may lose useful information, leading to a less-satisfactory performance.

2.3 Cooperative Coevolution

Cooperative coevolution [Potter and Jong, 2000] is a famous evolutionary framework that has been shown suitable for large-scale optimization. It uses a divide-and-conquer approach to decompose a large-scale problem into several small-scale subcomponents and evolve these subcomponents cooperatively. A number of approaches have been proposed to incorporate the cooperative coevolution framework to improve the performance of EAs [Ma *et al.*, 2019]. However, none of them is designed with the explicit goal of being able to solve neural network pruning problems. Furthermore, due to different properties of problems, it will often lead to a bad performance if applying a decomposition strategy designed for one problem directly to another. In this work, we adapt cooperative coevolution to neural network pruning for the first time, and employ a decomposition strategy based on the characteristics of forward propagation of CNNs.

3 The CCEP Algorithm

Let Φ denote a well-trained CNN with n convolution layers $\{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n\}$, where $\mathcal{L}_i$ indicates the ith layer and contains l_i filters. The aim of neural network pruning is to determine a subset of filters in Φ for removal so that the resulting pruned network has maximum accuracy and minimum computational cost. Let $\boldsymbol{\sigma} = \{\sigma_{ij} \mid \sigma_{ij} \in \{0,1\}, i \in \{1,2,...,n\}, j \in \{1,2,...,l_i\}\}$, where σ_{ij} denotes whether the jth filter (denoted as $\mathcal{L}_{ij}$) in the ith layer of the network Φ is retained. That is, if $\sigma_{ij} = 1$, the filter $\mathcal{L}_{ij}$ is retained, otherwise it is removed. Thus, a vector $\boldsymbol{\sigma}$ naturally represents a pruned network $\Phi_{\boldsymbol{\sigma}} = \bigcup_{i=1}^{n} \bigcup_{j=1}^{l_i} \sigma_{ij} \mathcal{L}_{ij}$. When σ_{ij} equals to 1 for any i and j, $\Phi_{\boldsymbol{\sigma}}$ is actually the original network Φ.

Under the solution representation based on vector $\boldsymbol{\sigma}$, neural network pruning can be formulated as an optimization problem that maximizes the accuracy and minimizes the computational cost of a pruned network $\Phi_{\boldsymbol{\sigma}}$ simultaneously. Using the number of FLoat point OPerations (FLOPs), which is a common metric, to measure the computational cost, the pruning problem can be formally described as

$$\underset{\boldsymbol{\sigma} \in \{0,1\}^{\sum_{i=1}^{n} l_i}}{\arg\max} \left(\text{Accuracy}(\Phi_{\boldsymbol{\sigma}}), -\text{FLOPs}(\Phi_{\boldsymbol{\sigma}}) \right). \quad (1)$$

This is essentially a large-scale optimization problem, as a deep CNN usually contains many filters, i.e., $\sum_{i=1}^{n} l_i$ is large.

To solve the problem, a novel algorithm based on cooperative coevolution named CCEP, is proposed. The main idea is

to make use of the divide-and-conquer technique applied in the cooperative coevolution framework, but transferring it to the context of neural network pruning. Despite cooperative coevolution has achieved impressive success in large-scale optimization, it has been reported that it would have a poor performance in non-separable problems if a proper decomposition strategy is not given [Ma *et al.*, 2019]. As the filters in CNNs are usually highly related, the problem of neural network pruning is inherently non-separable. To solve this issue, a decomposition strategy based on the specific characteristics of typical CNNs is used in CCEP. Specifically, considering that the features are commonly processed layer by layer during the forward propagation of a CNN, the impact of removing a filter mainly depends on whether the other filters in the same layer can retain the representation ability. Thus, a natural strategy is to decompose the search space by layer. That is, the vector $\boldsymbol{\sigma}$ of decision variables is split into n groups, where $\forall i \in \{1, 2, ..., n\}$, the ith group corresponds to the ith layer and contains l_i variables $\{\sigma_{i1}, \ldots, \sigma_{il_i}\}$. After that, CCEP coevolves n subpopulations representing these groups, each of which is optimized with a separate EA. It is worthy noting that CCEP is highly decoupled with the architecture of the original network, and thus can be applied to different networks flexibly and extended easily. Details of the framework of CCEP and the EA in each group are described below.

3.1 The Framework of CCEP

The framework of CCEP is shown in Algorithm 1. It employs an iterative prune-and-finetune strategy and finally outcomes a set of pruned networks with different pruning ratios for user selection. During each iteration, the algorithm works as follows. At the beginning, it splits the candidate filters to be pruned into n groups by layer in line 4 of Algorithm 1. Once the groups are created, the algorithm optimizes these groups in parallel in line 5. A separate EA is employed for each group, and it will continue until the stop condition specified in the EA is reached and outcome an individual representing the specific pruned layer. After that, a collaboration among these groups takes place in line 6, i.e., a complete pruned network will be created by splicing the n resultant pruned layers in their intrinsic order. At last, the algorithm will perform a finetune process to recover the accuracy of the obtained complete pruned network in line 7. An illustration of one complete coevolution process is shown in Figure 1. The finetuned pruned network Φ' will be preserved into the archive A in line 8, and used as the base network Φ_0 in line 9 for the next iteration. After running T iterations, the pruned networks in the archive A are output in line 12.

3.2 Details of the EA in Each Group

Generally, the EA in each group follows the typical evolutionary framework. It starts by initializing a subpopulation for the current group, and then evolves the subpopulation iteratively by generating and evaluating new individuals, and selecting better individuals to remain in the next generation. When terminating, the EA selects an individual from the final subpopulation and outcomes the corresponding pruned layer. The whole process is shown in Algorithm 2 and Figure 2.
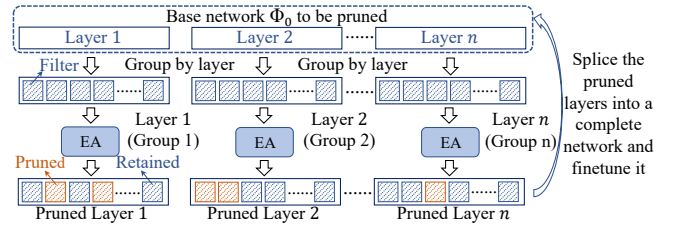

Figure 1: Illustration of the cooperative coevolution process of CCEP. A network with n layers is iteratively evolved by being divided into n groups, each of which represents a single layer, applying an EA in each group in parallel to obtain specific pruned layers, generating a complete pruned network by splicing the n pruned layers, and finetuning the generated network.

Algorithm 1 CCEP framework

Input: A well-trained CNN Φ, maximum iteration T
Output: A set of pruned networks with different sizes
Process:
1: Set the base network $\Phi_0 = \Phi$;
2: Let $A = \emptyset$, and $i = 0$;
3: **while** $i < T$ **do**
4:     Group the network Φ_0 by layer;
5:     Apply an EA in each group in parallel to obtain specific pruned layers, as shown in Algorithm 2;
6:     Generate a complete pruned network by splicing all pruned layers in their intrinsic order;
7:     Finetune the complete pruned network;
8:     Add the finetuned network Φ' into A;
9:     Set $\Phi_0 = \Phi'$;
10:     $i = i + 1$
11: **end while**
12: **return** the pruned networks in A

Concretely, the subpopulation is initialized with m individuals in line 1 of Algorithm 2. The first individual is created by the vector with all 1s, denoted as I_0, to encourage a conservative pruning. The other $m - 1$ individuals are generated by applying the following bit-wise mutation operator with a mutation rate p_1 on I_0. Specifically, given an individual, the bit-wise mutation operator flips each bit of the individual with some probability (called mutation rate) independently until all bits have been tried or an upper bound r on the pruning ratio that limits the maximum number of pruned filters has been met. That is, the number of 0s (i.e, the number of pruned filters) of the generated individual should be no larger than $|I_0| \times r$, where $|I_0|$ is the number of filters contained by the ith layer of the current base network, preventing the pruning from being too violent. A detailed explanation of the bit-wise mutation operator is provided in the supplementary material.

In each generation of Algorithm 2, an offspring individual is generated by applying the bit-wise mutation operator with a mutation rate p_2 on a parent individual randomly selected from P, and this process is repeated independently to generate m offspring individuals in line 4. When evaluating an offspring individual in line 5, which corresponds to a single
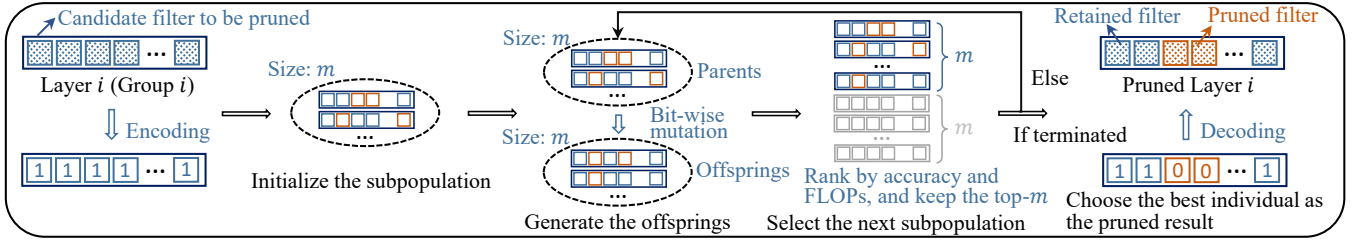
Figure 2: Illustration of the EA in each group of CCEP. Given the candidate filters of the ith layer to be pruned, it employs an iterative evolutionary optimization process to search for an optimal pruned layer in terms of accuracy and FLOPs. Six important steps are involved: encoding, initialization, offspring generation, environmental selection, final individual selection, and decoding.

Algorithm 2 EA in each group

Input: The ith layer, population size m, data set D_s, mutation rates p_1, p_2, ratio bound r, maximum generation G
Output: A pruned layer
Process:

1: Initialize a subpopulation P with I_0 and $m-1$ individuals generated from I_0 by bit-wise mutation with p_1 and r;
2: Let $j = 0$;
3: **while** $j < G$ **do**
4:   Generate m offspring individuals, each of which is generated by randomly selecting a parent individual from P and conducting bit-wise mutation with p_2 and r on the parent individual;
5:   Calculate the accuracy of the m individuals on D_s as well as their FLOPs;
6:   Set Q as the union of P and the m individuals;
7:   Rank the individuals in Q by accuracy and FLOPs;
8:   Select the top m individuals and use them to update P;
9:   $j = j + 1$
10: **end while**
11: Conduct the final individual selection;
12: **return** the pruned layer w.r.t. the selected individual

---

pruned layer, a complete network is first formed by obtaining other layers from the base network of the current iteration and splicing them with the current pruned layer in their intrinsic order, as shown in Figure 3. The resultant complete network is evaluated in terms of accuracy, which is conducted on a small randomly sampled data set D_s to improve efficiency, and FLOPs. The results are assigned back to the individual under evaluation. After that, the m offspring individuals and the individuals in the current subpopulation P will be merged and ranked in lines 6–7, and the top m individuals will be selected to form the new subpopulation for the next generation in line 8. For the ranking, all individuals are ranked by their accuracy descendingly, and for the individuals with the same accuracy, the one with fewer filters is ranked higher. Note that to reproduce offspring individuals, only the simple bit-wise mutation operator is used, but the experimental results in Section 4 have shown that it is good enough. Using more sophisticated operators (e.g., crossover operators) is expected to further improve the performance.

After running G generations, the final individual selection will be carried out. Two strategies are considered. One is to
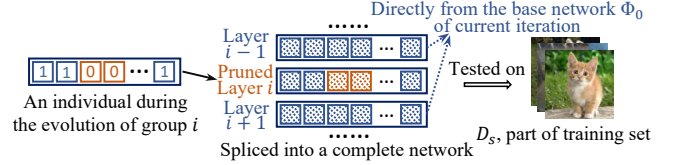

Figure 3: Illustration of individual splicing for evaluation in the EA of group i, where an individual corresponding to a pruned ith layer is spliced with other layers from the current base network to form a complete network, and then evaluation can be conducted as usual.

always select the top-ranked individual in the final subpopulation, called strategy Sel_a. The other strategy called Sel_b, is to select the top-ranked one except the individual with all 1s. That is, if the top-ranked individual in the final subpopulation has all bits equal to 1 (i.e., does not prune any filter), the strategy Sel_b will select the runner-up one. Sel_b can be used to speed up the whole pruning process while mildly sacrificing accuracy, which is useful in complex problems. The influence of these two strategies will be examined in the experiments.

4 Experiments

The experiments are conducted from two aspects. The first is to compare CCEP with existing pruning methods, including not only state-of-the-art pruning methods which require experts to design good criteria or auxiliary modules, but also automatic EA-based pruning methods. The second is to visualize the architecture of the pruned networks, and examine the effectiveness of introducing the pruning ratio bound r in mutation and the influence of hyper-parameters of CCEP.

Two classic data sets CIFAR10 [Krizhevsky and Hinton, 2009] and ImageNet [Russakovsky $et~al.$, 2015] for image classification are used for the examination. Two typical networks with different architectures are examined, i.e., VGG [Simonyan and Zisserman, 2015] and ResNet [He $et~al.$, 2016]. Following the common settings, the pruning is on all convolution layers for VGG, while it is on the first convolution layer of the normal residual blocks and the first and second convolution layers of the bottleneck blocks for ResNet.

The methods for comparison include six typical criteria-based pruning methods: L1 [Li $et~al.$, 2017], SFP [He $et~al.$, 2018a], FPGM [He $et~al.$, 2019], ThiNet [Luo $et~al.$, 2019], HRank [Lin $et~al.$, 2020] and ManiDP [Tang $et~al.$, 2021]; seven learning-based methods: AMC [He $et~al.$, 2018b], GAL [Lin $et~al.$, 2019], MetaPruning [Liu $et~al.$, 2019],

LFPC [He *et al.*, 2020], Pscratch [Wang *et al.*, 2020], BC-Net [Su *et al.*, 2021] and GReg [Wang *et al.*, 2021]; and two EA-based methods: DeepPruningES [Fernandes and Yen, 2021] and ESNB [Zhou *et al.*, 2021]. All the results of the pruning ratio of FLOPs and accuracy are obtained directly from their original reports.

4.1 Experiments on CIFAR-10

CIFAR-10 has 50K images in the training set and 10K images in the test set, with the size of 32×32 for 10 categories. ResNet-56, ResNet-110 and VGG-16 are tested, where the common variant of VGG-16 [Li *et al.*, 2017] for CIFAR-10 is used here. The settings of CCEP are described as follows. It runs for 12 iterations, i.e., $T = 12$ in Algorithm 1. For the EA (i.e., Algorithm 2) in each group, the population size m is 5, the mutation rates $p_1 = 0.05$ and $p_2 = 0.1$, the ratio bound r is 0.1, the maximum generation G is 10, and 20% of the training set is used for accuracy evaluation. For the final individual selection in line 11 of Algorithm 2, the strategy Sel_a is used. The setting of finetuning in line 7 of Algorithm 1 is provided in the supplementary material. We run CCEP independently for 10 times with different random seeds.

The comparison results in terms of accuracy drop and pruning ratio of FLOPs are shown in Table 1. For CCEP, we only present the solution with a pruning ratio similar to the comparison methods in Table 1 from the generated archive A. Generally, CCEP not only outperforms state-of-the-art pruning methods which require experts to design good criteria or auxiliary modules, but also automatic EA-based pruning methods. In most cases, CCEP can achieve both larger pruning ratio and smaller accuracy drop. As for the comparison with DeepPruningES, CCEP shows an obvious superiority on accuracy drop with similar pruning ratios. Additionally, we present the curves of pruning process of CCEP during the 10 independent runs in Figure 4. The solid line is the mean value of the 10 runs, and the shadow area represents the 95% confidence interval. The results imply a good stability of CCEP.
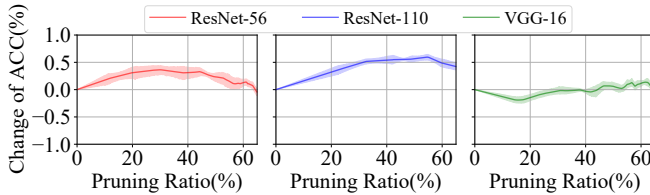

Figure 4: Repetition test of CCEP on CIFAR-10.

4.2 Experiments on ImageNet

ImageNet contains 1.28M images in the training set and 50K in the validation set, for 1K classes. The size of each image is 224×224. ResNet-34 and ResNet-50 are examined. Since ImageNet is much complex than CIFAR-10, a slightly larger mutation rate and pruning ratio bound (i.e., $p_1 = 0.1$ and $r = 0.15$) are used to improve the pruning speed, and also only 1% of the training set is used for accuracy evaluation. The other settings of CCEP are same as that on CIFAR-10.

The results are shown in Table 2, where the algorithms are sorted by their pruning ratios. For CCEP, we select three so-

Method	Architecture	Base ACC(%)	ACC↓ (%)	FLOPs↓ (%)
Hrank		93.26	0.09	50.00
AMC		92.80	0.90	50.00
Pscratch		93.23	0.18	50.00
FPGM		93.59	0.33	52.30
SFP		93.59	0.24	52.60
ESNB	ResNet-56	93.59	0.62	52.60
LFPC		94.37	0.35	52.90
GAL-0.8		93.26	1.68	60.20
ManiDP		93.70	0.06	62.40
CCEP		93.48	**-0.24**	**63.42**
DeepPruningES		93.37	2.65	66.23
ESNB		93.25	-0.12	25.19
SFP		93.68	-0.18	40.80
GAL-0.5		93.50	0.76	48.50
FPGM	ResNet-110	93.68	-0.17	52.30
Hrank		93.50	0.14	58.20
DeepPruningES		93.80	2.46	64.84
CCEP		93.68	**-0.22**	**67.09**
L1		93.25	-0.15	34.20
GAL-0.05		93.96	0.19	39.60
BCNet	VGG-16	93.99	-0.37	50.63
HRank		93.96	0.53	53.50
CCEP		93.71	**-0.19**	**63.20**
DeepPruningES		93.94	2.98	65.49

Table 1: Comparison in terms of accuracy drop and pruning ratio on CIFAR-10. The algorithms are listed in ascending order of the pruning ratio. The results of our algorithm CCEP are shown in bold.
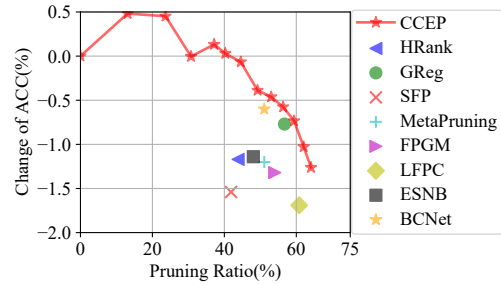

Figure 5: Pruning comparison for ResNet-50 on ImageNet.

lutions, which have pruning ratios similar to that in Table 2, from the final archive A generated in line 12 of Algorithm 1. Generally, CCEP surpasses its counterparts at different levels of pruning ratio with less drop of accuracy. To the best of our knowledge, this is the first time that an EA-based filter pruning method has been examined on ImageNet. The superiority of CCEP on such a large data set provides a more comprehensive demonstration of its effectiveness for neural network pruning. Besides, all the pruned networks (i.e., the archive A in Algorithm 1) generated during the pruning process of CCEP on ResNet-50 are shown in Figure 5. On one hand, CCEP can achieve a set of pruned networks with different pruning ratios in a single run, providing a better flexibility for users. On the other hand, the curve implies the ability of CCEP in finding a good tradeoff between accuracy and pruning ratio. That is, at the beginning, the accuracy of the

Architecture	Method	Base Top-1(%)	Pruned Top-1(%)	Base Top-5(%)	Pruned Top-5(%)	Top-1 ↓	Top-5 ↓	FLOPs ↓(%)
ResNet-34	GReg	73.31	73.61	-	-	-0.30	-	24.24
	CCEP-1	**73.30**	**73.64**	**91.42**	**91.51**	**-0.34**	**0.11**	**25.44**
	FPGM	73.92	72.63	91.62	91.08	1.29	0.54	30.00
	SFP	73.92	71.83	91.62	90.33	2.09	1.29	41.10
	CCEP-2	**73.30**	**72.67**	**91.42**	**90.90**	**0.63**	**0.72**	**42.10**
ResNet-50	SFP	76.15	74.61	92.87	92.06	1.54	0.81	41.86
	CCEP-1	**76.13**	**76.06**	**92.86**	**92.81**	**0.07**	**0.05**	**44.56**
	ESNB	77.27	76.13	-	-	1.14	-	48.06
	MetaPruning	76.60	75.40	-	-	1.20	-	51.10
	BCNet	77.50	76.90	-	93.30	0.60	-	51.10
	FPGM	76.15	74.83	92.87	92.32	1.32	0.55	54.00
	ThiNet	75.30	72.03	92.20	90.99	3.27	1.21	55.75
	CCEP-2	**76.13**	**75.55**	**92.86**	**92.63**	**0.58**	**0.23**	**56.35**
	GReg	76.13	75.36	-	-	0.77	-	56.71
	LFPC	76.13	74.46	92.87	92.32	1.67	0.55	60.80
	Hrank	76.15	71.98	92.87	91.01	4.17	1.86	61.12
	CCEP-3	**76.13**	**74.87**	**92.86**	**92.35**	**1.26**	**0.51**	**64.09**

Table 2: Comparison in terms of accuracy drop and pruning ratio on ImageNet. The algorithms are listed in ascending order of the pruning ratio. The results of our algorithm CCEP are shown in bold. The '-' means that the corresponding result is not provided in its original paper.
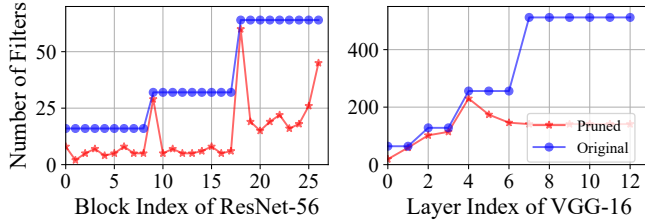

Figure 6: Visualization of the original networks and the pruned networks obtained by CCEP on CIFAR-10.

Figure 7: Comparison of CCEP with and without the ratio bound r for pruning ResNet-56 on CIFAR-10.

pruned network can increase as the pruning goes on, implying that the pruned redundant filters might bring noise. In the later stages, the accuracy starts to fall as too many filters are pruned and the representation ability might be damaged.

4.3 Further Studies and Discussion

Figure 6 visualizes the architectures (i.e., the number of filters in each block or layer) of the pruned networks on CIFAR-10, obtained by CCEP. We can observe that on ResNet-56, the filters around the expansion of channels are less pruned than others; while on VGG-16, the filters at the tail part tend to be pruned. These results imply that CCEP can achieve different degrees of redundancy in different layers automatically, which may be difficult to be captured manually.

We have introduced the ratio bound r in mutation to limit the ratio of pruned filters in each iteration of CCEP. Its effect is examined by comparing CCEP with and without r by 5 independent runs for pruning ResNet-56 on CIFAR-10. Figure 7 shows the average accuracy change and pruning ratio of their generated pruned networks. As expected, using r brings significantly better accuracy by preventing overly aggressive pruning, which would make it hard to recover the accuracy.

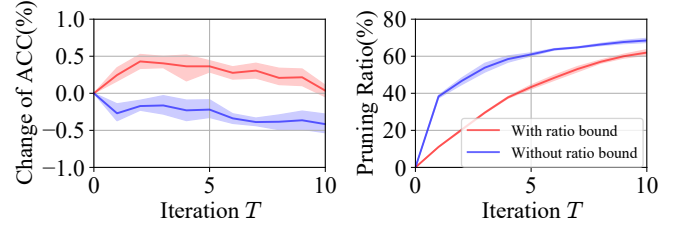The influence of hyper-parameters (including the population size m, maximum generation G, mutation rate p_1, ratio bound r and final individual selection strategy) of CCEP is examined by pruning ResNet-56 on CIFAR-10, and provided in the supplementary material. Generally, a larger G, or a larger p_1 and r would lead to a faster pruning process but at an expense of accuracy. For the final individual selection strategy, Sel_a achieves better accuracy while Sel_b shows superiority on efficiency especially on the complex data set. This is intuitive, as a faster pruning process is usually more violent. These results can guide the settings of CCEP to some extent in practice, and the settings used in our experiments are to make a trade-off between the pruning speed and accuracy.

5 Conclusion

This paper proposes an automatic neural network pruning algorithm CCEP. It applies cooperative coevolution to tackle the large search space of neural network pruning, where the layered characteristic of typical neural networks is utilized to tackle the non-trivial separation issue in cooperative coevolution. Experiments show the competitive performance of CCEP, compared with a number of state-of-the-art pruning methods. Besides, the framework of cooperative coevolution makes CCEP highly flexible and easily parallelizable. In the future, we will try to incorporate more advanced EA operators to further improve the performance of CCEP.

References

[Bäck, 1996] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford University Press, 1996.

[Fernandes and Yen, 2021] F. E. Fernandes and G. G. Yen. Pruning deep convolutional neural networks architectures with evolution strategy. *Information Sciences*, 552:29–47, 2021.

[Han *et al.*, 2016] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, 2016.

[He *et al.*, 2016] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

[He *et al.*, 2018a] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang. Soft filter pruning for accelerating deep convolutional neural networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2234–2240, Stockholm, Sweden, 2018.

[He *et al.*, 2018b] Y. He, J. Lin, Z. Liu, H. Wang, L. Li, and S. Han. AMC: Automl for model compression and acceleration on mobile devices. In *Proceedings of the 15th European Conference on Computer Vision (ECCV)*, pages 815–832, Munich, Germany, 2018.

[He *et al.*, 2019] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4340–4349, Long Beach, CA, 2019.

[He *et al.*, 2020] Y. He, Y. Ding, P. Liu, L. Zhu, H. Zhang, and Y. Yang. Learning filter pruning criteria for deep convolutional neural networks acceleration. In *Proceedings of the 2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2006–2015, Seattle, WA, 2020.

[Hinton *et al.*, 2015] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Hong *et al.*, 2019] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, and X. Yao. A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 23(3):525–537, 2019.

[Krizhevsky and Hinton, 2009] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

[Li *et al.*, 2017] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[Li *et al.*, 2018] G. Li, C. Qian, C. Jiang, X. Lu, and K. Tang. Optimization based layer-wise magnitude-based pruning for DNN compression. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2383–2389, Stockholm, Sweden, 2018.

[Lin *et al.*, 2019] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. S. Doermann. Towards optimal structured CNN pruning via generative adversarial learning. In *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2790–2799, Long Beach, CA, 2019.

[Lin *et al.*, 2020] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the 2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1526–1535, Los Alamitos, CA, 2020.

[Liu *et al.*, 2019] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K. Cheng, and J. Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the 2019 IEEE International Conference on Computer Vision (ICCV)*, pages 3295–3304, Seoul, Korea (South), 2019.

[Liu *et al.*, 2020] L. Liu, W. Ouyang, X. Wang, P. W. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. Deep learning for generic object detection: A survey. *International Journal of Computer Vision*, 128(2):261–318, 2020.

[Luo and Wu, 2020] J. Luo and J. Wu. Autopruner: An end-to-end trainable filter pruning method for efficient deep model inference. *Pattern Recognition*, 107:107461, 2020.

[Luo *et al.*, 2019] J. Luo, H. Zhang, H. Zhou, C. Xie, J. Wu, and W. Lin. Thinet: Pruning CNN filters for a thinner net. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(10):2525–2538, 2019.

[Ma *et al.*, 2019] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu. A survey on cooperative co-evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 23(3):421–441, 2019.

[Potter and Jong, 2000] M. A. Potter and K. A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[Russakovsky *et al.*, 2015] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[Simonyan and Zisserman, 2015] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[Su *et al.*, 2021] X. Su, S. You, F. Wang, C. Qian, C. Zhang, and C. Xu. Bcnet: Searching for network width with bilaterally coupled network. In *Proceedings of the 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2175–2184, Virtual Event, 2021.

[Tang *et al.*, 2021] Y. Tang, Y. Wang, Y. Xu, Y. Deng, C. Xu, D. Tao, and C. Xu. Manifold regularized dynamic network pruning. In *Proceedings of the 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5018–5028, Virtual Event, 2021.

[Wang *et al.*, 2020] Y. Wang, X. Zhang, L. Xie, J. Zhou, H. Su, B. Zhang, and X. Hu. Pruning from scratch. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, pages 12273–12280, New York, NY, 2020.

[Wang *et al.*, 2021] H. Wang, C. Qin, Y. Zhang, and Y. Fu. Neural pruning via growing regularization. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, 2021.

[Yao, 1999] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[Zhou *et al.*, 2021] Y. Zhou, G. G. Yen, and Z. Yi. Evolutionary shallowing deep neural networks at block levels. *IEEE Transactions on Neural Networks and Learning Systems*, in press, 2021.